

Методика составления школьных расписаний с использованием персонального компьютера

Пилюков Николай Петрович
www.mnogosmenka.ru

27 ноября 2023 г.

Аннотация

Данная публикация знакомит читателя с уникальным, сравнительно недавно появившимся, алгоритмом составления школьного расписания. Сообщаются результаты тестирования единственной в мире программы, которая может не составлять, а составить школьное расписание в полностью автоматическом режиме. Опираясь на результаты десятков миллионов тестов (построенных школьных расписаний) развенчивается миф о невозможности составления школьного расписания без участия человека. Приводится описание программы «АФМ: Аннушка» (версия 1.025.21), предназначенной для тестирования алгоритма любым желающим.

Для понимания основного содержания публикации, как правило, не требуется какой-либо специальной математической подготовки, таким образом, она адресована широкому кругу заинтересованных читателей, в первую очередь завучам и администрации средних школ, а так же начинающим разработчикам программ составления школьных расписаний.

Содержание

1	Введение	4
2	Поколения ПО составления учебных расписаний	5
2.1	Программы первого поколения	6
2.2	Программы второго поколения	7
2.3	Программы третьего поколения	8
3	Нужны ли программы составления школьных расписаний?	9
4	Немного о сложности решаемой задачи	11
5	Парадоксальность рынка ПО составления учебных расписаний	14

6 «Древнеегипетский» алгоритм	17
6.1 Общий взгляд на алгоритм решения задачи	17
6.2 Тестирование алгоритма	19
6.3 Полнота модели учебного процесса	22
6.4 Основные достоинства алгоритма и программы	23
7 О существовании решения математических задач	23
7.1 Примеры задач	23
7.2 О существовании решения задач адекватно описывающих учебный процесс	27
8 Особенности пользовательского интерфейса программы «АФМ: Аннушка»	27
9 Начало работы с программой «АФМ: Аннушка»	28
9.1 Установка программы и ее иконка	28
9.2 Главное окно программы	29
9.3 Кнопка панели управления Выход	31
10 Нулевой этап алгоритма	32
10.1 Окно Список решаемых задач	32
10.2 Окно Базовые параметры школы	33
10.3 Окошки сообщений об ошибках	35
11 Первый этап алгоритма	35
11.1 Окошко Количество параллелей в школе	36
11.2 Окно Список классов с параллелями	37
11.3 Окно Учебная нагрузка параллелей	38
11.4 Окошко Формирование санитарных ограничений для классов	39
12 Второй этап алгоритма	41
12.1 Окно Список специальных помещений	41
12.2 Окно Распределение нагрузки классов в специальных помещениях	42
12.3 Окошко Расчет расписания для специальных помещений	45
13 Третий этап алгоритма	46
13.1 Окно Формирование списка совместителей и их нагрузок	47
13.2 Окно Формирование таблицы рабочих дней совместителей	48
13.3 Окно Формирование запретов на первый урок для совместителей	49
13.4 Окно Формирование таблицы распределения педагогической нагрузки совместителей	50
13.5 Окошко Построение расписания для совместителей	52
14 Четвертый этап алгоритма	54
14.1 Окошко Количество потоков и старших параллелей объединяемых в потоки	54
14.2 Окно Формирование потоков	56
14.3 Окно Формирование нагрузок потоков	58

14.4	Окошко Построение расписания для потоков	59
15	Пятый этап алгоритма	59
15.1	Окно Формирование учебного плана для предметов с разделением на группы	60
15.2	Окно Расчет количества преподавателей для предметов с разделением на группы	61
15.3	Окно Список преподавателей и запретов на первый урок	62
15.4	Окно Распределение педагогической нагрузки в классах с разделением на группы	63
15.5	Окошко Построение расписания для классов с разделением на группы	65
16	Шестой этап алгоритма	66
16.1	Окно Список учебных предметов	66
16.2	Окно Формирование учебного плана для предметов без разделения на группы	69
16.3	Окно Список преподавателей	71
16.4	Окно Формирование таблицы компетентности преподавателей	72
16.5	Окно Распределение педагогической нагрузки	74
16.6	Окно Запреты на работу в первый урок	76
16.7	Окошко Построение расписания для классов без разделения на группы	76
17	Заключительный этап составления расписания	78
17.1	Назначение преподавателей в специальных помещениях	80
17.1.1	Окно Список преподавателей для спецпомещений	81
17.1.2	Окно Таблица компетентности преподавателей для спецпомещений	81
17.1.3	Окно Распределение нагрузки среди спецпреподавателей	82
17.2	Вкладка Расписание кабинетов	85
18	Дополнительные возможности программы	
	«АФМ: Аннушка»	86
18.1	Кнопки окон для ввода данных Закончить и кнопка панели управления Продолжить начатую задачу	86
18.2	Кнопка панели управления Возврат к пройденному шагу...	87
18.3	Пункты главного меню Запомнить/Вспомнить текущее расписание	89
18.4	Кнопка панели управления Автогенерация данных	90
18.5	Кнопка Вывод в файл Excel	94
18.6	Кнопка Найти подстроку	94
18.7	Просмотр длинного содержания ячеек	95
18.8	Архивирование и восстановление данных	95
18.8.1	Пункт главного меню Архивирование расписания	96
18.8.2	Пункт главного меню Восстановление расписания	97
19	Расчет расписания для школ, работающих в две смены	99

20 Заключение	103
---------------	-----

Список литературы	104
-------------------	-----

1 Введение

Кто из людей умеющих читать и писать, а значит из тех кто учился в школе, не знаком с таблицей которая висит в вестибюле каждой школы и которая называется — расписание. Однако подавляющее большинство из них, да пожалуй почти все, никогда не задумывались от том: легко ли составлять такие таблицы?

Вместе с тем, проблема составления школьного расписания имеет продолжительную, интересную и порой весьма загадочную историю. Для читателя, что называется «не в теме» пожалуй весьма неожиданно будет узнать о том, что до настоящего времени не удалось построить (придумать) стройную и логичную теорию решения этой задачи.

За последнее полтора – два десятка лет в Российской Федерации было защищено не менее дюжины диссертаций по темам связанным с задачей составления учебных расписаний. За десятилетие, перед этим периодом, количество защищенных диссертаций не меньше, а может быть и еще больше. Хотя в основном диссертации защищаются на звание кандидата технических наук и рассматриваются задачи составления расписания занятий для высших учебных заведений, тем не менее данный факт свидетельствует о том, что к задачам составления школьного расписания проявляет внимание все больше и больше исследователей. Возможно этот поток работ связан с постоянным прогрессом и всеобщей доступностью вычислительной техники. Действительно на наших глазах произошли по истине удивительные процессы. Еще каких-нибудь тридцать пять лет назад, такую электронно-вычислительную машину, как ЕС1066, могло себе позволить приобрести только крупное, как правило оборонное, предприятие. Располагался такой компьютер в помещении площадью до нескольких сот квадратных метров, оборудованном мощной системой бесперебойного электропитания и системой поддержки микроклимата. Такие электронно-вычислительные машины в первую очередь предназначались для решения уникальных научно-технических задач, оказывающих влияние на обороноспособность страны.¹ Сегодня у многих дома на письменных столах стоят персональные компьютеры. Но только вдумайтесь. Оперативная память такого персонального компьютера по сравнению с выше упомянутым гигантом больше в 1000 – 2000 раз. Быстродействие выше, более чем в 1000 раз. И это не описка. Более чем в тысячу раз.

Сегодня для образованного человека, который никогда не сталкивался (или еще не сталкивался) с проблемой автоматического составления расписания в школе, включая людей которые называют себя IT-специалистами уровня эксперта, утверждение о том, что ранее не существовало программного обеспечения которое могло бы решить данную задачу (составить расписание в автоматическом режиме без участия человека) кажется просто абсурдным. Логика их рассуждений очень проста: — «При таком развитии техники, этого просто не может быть». Вторым аргументом, как им кажется бьющим «наповал», является обращение к любой поисковой системе имеющейся в сети Интернет. По поисковому запросу: — «Программа для составления расписания уроков», открываются десятки, если не сотни ссылок. Достаточно быстро находятся программы обещающие качественное решение данной задачи в автоматическом режиме. Точнее говоря, нахо-

¹Поэтому весь этаж где располагалась такая ЭВМ обтягивали мелкой металлической сеткой, дабы исключить возможность «электронного подглядывания» со стороны заклятых врагов советской власти.

дятся авторы декларирующие то, что их приложение может *составлять* расписание в автоматическом режиме.

При этом заметим, авторы декларируют то, что их программа может именно *составлять* школьное расписание в автоматическом режиме, но не говорят о том, может ли их программа такое расписание *составить*. А это, как должно быть понятно любому человеку знающему русский язык, совсем и далеко не одно и то же².

Таким образом, мы сегодня имеем:

Во-первых. Массовое заблуждение неспециалистов (а точнее людей, не посвятивших несколько десятков часов, работе с данными программами) о том, что эту задачу легко решить и о том, что рынок до отказа забит такими программами.

Во-вторых. Огромную армию завучей, попытавшихся применить такие программы в своей работе, и понявших, что их обманули, унизили, оскорбили, надругались и т.п.

В-третьих. Широко распространенный миф среди людей, которые считают себя специалистами в области разработки таких программ, о том, что задачу составления школьного расписания в автоматическом режиме невозможно решить в принципе. И это (невозможность решения) строго доказанный математический факт³.

Данная публикация посвящена, с одной стороны, развеиванию первого массового заблуждения о легкости решаемой задачи, и, с другой стороны, опровержению мифа о том, что такая задача не поддается решению в принципе.

2 Поколения ПО составления учебных расписаний

Первые публикации на тему применения вычислительной техники с целью автоматизировать составление расписания занятий появились в начале 60-х годов предыдущего столетия. Таким образом, задача составления учебного расписания с помощью вычислительной техники имеет достаточно продолжительную историю.⁴ За последние 60 с лишним лет интенсивных исследований была проведена огромная интеллектуальная работа многих тысяч специалистов по всему миру.

Совершенно не удивительно то, что программы для составления школьного расписания появлялись и совершенствовались по мере развития вычислительной техники. Поэтому обратимся (естественно в телеграфном стиле) к весьма условным периодам этого развития.

Не вдаваясь сильно в исторические изыскания и не рискуя сильно ошибиться, появление действующего компьютера (электронно-вычислительной машины — ЭВМ) и приносящего реальную пользу, можно отнести к 1945 году. Это появление (опять же не рискуя сильно ошибиться) можно обосновать огромной потребностью в вычислениях для военных нужд. Одной из первых задач которые решались на первых ЭВМ была задача составления баллистических таблиц для артиллерии и авиации. Не последнюю роль в потребности военных играла задача изучения атомного и термоядерного взрыва. В силу

²Подчеркнем — не составлять, а составить. То есть, не расставить 97% учебной нагрузки (найти тупик), а расставить всю нагрузку без исключения, и кроме того, всегда, то есть во всех случаях при условии допустимых исходных данных.

³Очень оригинальный довод во время обсуждения вопроса о возможности решения задачи составления школьного расписания в автоматическом режиме привел один из руководителей компании, занимающейся перепродажей программы составления школьного расписания: — «Эту задачу решить невозможно, я сам пробовал».

⁴Самой же задаче составления учебного расписания (без помощи вычислительной техники) по всей видимости, уж никак не менее трехсот лет.

выше указанных причин, сам факт существования ЭВМ и принципы ее работы по началу оставались засекреченными.

Примерно десять лет понадобилось на то, что бы довести сведения о «тактико-технических характеристиках» первых ЭВМ до широкого круга узких специалистов — математиков занимающихся численными методами. Результат не заставил себя долго ждать. С 1955 года, вместе с появлением языка высокого уровня Fortran, наблюдался взрывообразный рост такой отрасли научных знаний, как прикладная математика. Сотни и тысячи практически важных задач стали предметом исследования математиков с применением электронно-вычислительной техники, что повлекло за собой разработку совершенно новых численных методов решения этих задач. По той причине, что стоимость компьютеров была совершенно не сопоставима с тем экономическим эффектом которые они могли принести для гражданского промышленного предприятия, единственными пользователями этой техники были военные и весьма узкий круг ученых. Другими словами те люди которые не знали слов — дорого, затраты или словосочетания — экономический эффект.

Но время шло. Технологии производства и проектирования вычислительной техники развивались стремительными темпами. В результате, производительность компьютеров росла не бывальными шагами, а их стоимость быстро снижалась. Цены на компьютеры от астрономических неуклонно приближались к земным (хотя и еще заоблачным). К 1965 году весьма заметно вырос круг ученых которым для исследований была доступна вычислительная техника. К этому времени (начало шестидесятых годов), как было отмечено выше, и относятся первые публикации на тему составления на больших ЭВМ школьного расписания. Совершенно естественно, что работы поначалу имели постановочный характер, а позже теоретический.

2.1 Программы первого поколения

Около пятнадцати лет понадобилось на то, что бы придумать все, что можно было легко придумать в отношении задачи составления школьного расписания. Этот период (с 1965 до 1980 года) вызывает резкие смешанные чувства. С одной стороны, были предложены красивые и оригинальные математические модели задачи составления школьного расписания (вершинная раскраска графов, реберная раскраска графов), а с другой стороны, вне всякого сомнения эти модели следует отнести к весьма упрощенному варианту организации учебного процесса. Другими словами, полностью задача не была решена и даже не сформулирована в деталях. Более того, в 1976 году появилась работа израильских математиков [18] где, по их мнению, была доказана принципиальная трудность решения задачи составления школьного расписания, в том случае если преподавателям самостоятельно позволить задавать для себя, так называемые, методические дни.

Итак, к 1980 году не смотря на то, что производительность ЭВМ постоянно повышалась, а их стоимость постоянно снижалась, в результате чего уже и некоторые гражданские промышленные предприятия перешли в категорию активных пользователей вычислительной техники, наша задача по прежнему оставалась до конца не решенной, а вычислительная техника для главного пользователя — школ, оставалась недоступной. Вот пожалуй к этому периоду можно было бы отнести программы первого поколения для составления расписания учебных занятий.

2.2 Программы второго поколения

В силу выше указанных двух причин (труднорешаемость задачи и недоступность вычислительной техники для конечного потребителя) интерес, в начале 80-х годов, к автоматическому составлению расписания занятий заметно ослабел (а может быть даже и полностью угас). Высшие учебные заведения, использующие данное программное обеспечение, взяли крен от собственно составления расписаний занятий к учету и контролю успеваемости студентов. Еще раз подчеркнем, что администрации школ в подавляющем большинстве даже и не догадывались о существовании программ составления учебных расписаний.

Однако, к этому времени (естественно за рубежом) среди некоторых «яйцеголовых» студентов возникает мода на конструкторы из радиодеталей. Забрехала эра персональных компьютеров. Мода оказалась весьма прилипчивой и круг «яйцеголовых» неуклонно расширялся. Весьма вероятно, что конструкторы из радиодеталей так и остались бы уделом кучки «ненормальных» если бы самый крупный в то время производитель пишущих машинок, а заодно и самых распространенных в то время больших ЭВМ, американская корпорация IBM, приблизительно в 1985 году, прозорливо не смекнула бы, что эти конструкторы, если им придать форму пишущей машинки, могли бы эти пишущие машинки заменить. Причем не просто заменить, а сделать из пишущей машинки сверхинтеллектуальную пишущую машинку составив конкуренцию «свинцовым технологиям» в издательском деле. Разумеется в то время никто, разве что кроме самых прозорливых, и предположить не мог, что конструкторы из радиодеталей когда-нибудь смогут составить конкуренцию настоящим вычислительным устройствам.

Тем не менее, жребий был брошен, началось массовое производство убийц пишущих машинок. Не долго себя заставляли ждать и идеи производства сначала «два в одном» (пишущая машинка плюс помощник бизнесмена — электронная таблица), потом «три в одном» (еще плюс бухгалтерская программа), потом «четыре в одном» (еще плюс программа учета товаров на складе), и так далее, и так далее, и так далее. Вчерашние студенты по мановению волшебной палочки стали превращаться в миллионеров и даже в миллиардеров, а бывшие конструкторы из радиодеталей все больше и больше стали походить на настоящие электронно-вычислительные машины. В технический и деловой язык вошла уважительная аббревиатура «Пи-Си» (PC), что означало персональный компьютер и уже в конце 80-х годов XX столетия ни у кого не вызывало сомнений, что у них на столе располагалась не игрушечная, а совершенно настоящая электронно-вычислительная машина которая может приносить реальную пользу.

Противоположные тенденции — взрывообразный рост производительности бывших игрушек, с одной стороны, и стремительное падение их цены, с другой стороны, сделали свое дело. В некоторых продвинутых школах на столе у руководителей появились, по сегодняшним меркам, здоровущие мониторы, которые как живой укор вопили: — «Наполни меня нужным программным обеспечением». Не мудрено, что вспомнилась уже казалось бы совсем забытая идея составления расписания учебных занятий. Тысячи любителей легкой наживы бросились писать программы для школ, гарантируя полную автоматизацию и невиданное улучшение всего, что только попадет под руку. К этому периоду, пожалуй можно отнести программы второго поколения, автоматизирующие процесс составления школьных расписаний.

В девяностых годах прошлого столетия индустрия персональных компьютеров переживала невероятный рост. Производительность персоналок увеличивалась вдвое чуть ли не каждый год и каждый год приносил инновационные программные продукты. У работающих в этой сфере «на ботинках рвались подметки». А программы по составлению

школьных расписаний что-то никак не хотели составлять эти расписания . . .

Сейчас конечно трудно сказать, знали или нет производители программ для составления школьного расписания о том наследстве которое им оставили их предшественники в 1965 – 1980-х годах прошлого столетия и о предупреждении израильских математиков 1976 года о том, что эта задача труднорешаемая, но факт остается фактом — администрация учебных заведений потихоньку списывала старые добрые пишущие машинки заменяя их персональными компьютерами. При этом, расписание по прежнему, за небольшими исключениями, составлялось в ручную.

2.3 Программы третьего поколения

К началу XXI века вместе с окончательным доминированием операционных систем с графическим пользовательским интерфейсом, приходит и конец программам второго поколения по составлению школьного расписания, которые использовали псевдографический интерфейс ушедшей операционной системы MS-DOS и, по сегодняшним меркам, мизерные возможности персоналок конца двадцатого столетия. Индустрия персональных компьютеров благополучно прекратила свое бурное развитие и перешла к пресловутой «стабильности». Персональная вычислительная техника преодолела рубеж производительности больших ЭВМ середины 80-х годов прошлого века, стоимость персоналок снизилась со стоимости легкового автомобиля до месячной зарплаты рядового инженера, таким образом, все было готово для разработки программ третьего поколения.

И действительно, в самом конце прошлого столетия несметное количество производителей, в очередной раз но уже, как им казалось, на новом техническом и технологическом уровне взялись за разработку программ составления школьного расписания. На фоне прекращения заметного (хотя и плавного) роста производительности персональных компьютеров, стабилизации идей в области программного обеспечения развивались программы, которые можно было бы отнести к программам третьего поколения. Основной особенностью этих программ, как нам кажется, является то, что они могли бы разрабатываться учитывая как ошибки, так и оригинальные находки предшественников. Здесь в первую очередь имеются в виду разработчики девяностых. С математическими результатами шестидесятых, семидесятых и восьмидесятых дело обстоит проще. Если ты о них знаешь, то используешь, если не знаешь, то «придумываешь велосипед заново». Другой особенностью является то, что эти программы разрабатывались с использованием нового на то время — графического интерфейса пользователя. Нет сомнений в том, что графический интерфейс предоставляет разработчику принципиально бóльшие возможности по сравнению с псевдографическим (текстовым). Но в этом, в тоже время, кроется и опасность. Если мы начнем сравнивать имеющиеся на рынке (в использовании) программы составления школьного расписания, то обнаружим совершенно потрясающее разнообразие способов формирования (ввода) исходных данных, необходимых для расчета, хотя с математической точки зрения большинство программ, вроде бы, делают (или по крайней мере, должны делать) совершенно одно и тоже. Таким образом, на качество программ составления школьного расписания стало оказывать существенное влияние логичность и удобство пользовательского интерфейса.

Сегодня, с высоты 2023 года, стоит отметить, что по сравнению с программами девяностых, программы третьего поколения (нулевых) весьма «поумнели», но оптимизма у разработчиков заметно поубавилось. Обещать полную автоматизацию всего, что только попало под руку, уже никто (или почти никто, разумеется за исключением откровенных мошенников) не берется.

Многие из проектов начатых на излете девяностых к настоящему времени прекратили свое существование в силу их невостребованности. Другие же продолжают развиваться и совершенствоваться. Третьи на протяжении последних двадцати лет застыли в своем развитии. Четвертые, появившиеся сравнительно недавно, и присосавшись к, так называемым, «современным платформам», предназначенным для решения совершенно других задач, обещают 97% расстановки учебной нагрузки, т.е. нахождение тупика, когда не существует ни одного места в сетке расписания куда можно было бы «приткнуть» оставшиеся 20 – 30 занятий. Таким образом, как ранее отмечалось, об окончательном и бесповоротном решении задачи составления учебного (школьного) расписания говорить было еще преждевременно.

3 Нужны ли программы составления школьных расписаний?

Обычно, говоря о пользе (необходимости) использования программы по автоматизированному расчету учебного расписания, указывают такой фактор, как — на порядок сокращение трудозатрат (времени) завуча при его составлении. Часто указывают на то, что расписание с помощью компьютера может быть получено лучшего качества. Хотя этот довод, учитывая сказанное чуть ниже, является не бесспорным.

По нашему мнению следует согласиться с тем, что расчет расписания с помощью компьютера позволит, кроме экономии времени и получения лучшего качества расписания, с одной стороны, исключить субъективные оценки и личные симпатии по отношению к учителю (части учителей), при составлении расписания, в том числе и при распределении педагогической нагрузки, а с другой стороны, позволит полностью исключить незаслуженные обвинения в адрес завуча или администрации школы со стороны учителей, в таких субъективных оценках и симпатиях, поскольку очевидно, что компьютер «лицо незаинтересованное» (во всем «виноват» компьютер). Тем самым, расчет распределения педагогической нагрузки и генерация расписания на компьютере, может улучшить психологический климат в учительском коллективе (соблюсти принципы справедливости и равноправия), точно так же как улучшает настроение игроков футбольной команды судья матча после того как разыгрывает право первого удара по мячу с помощью жребия.

В 2001 году, ныне покойной, компанией «Хронобус» было проведено анкетирование почти 1000 московских школ по вопросу необходимости создания и внедрения АРМ(а) «Расписание» [15]. Результаты анкетирования показали, что у всех школ имеется искреннее желание пользоваться такой программой, но никто этого не делает. Причем причиной дружного игнорирования подобных средств автоматизации является вовсе не отсутствие необходимой техники или денег, а качество предлагаемых на рынке программ. Фраза: — «Если бы мне предложили увеличить зарплату в полтора или два раза, за то что я пользуюсь такой программой составления школьного расписания, то я бы от этого предложения отказалась» была не редкостью. Другими словами по мнению завучей программы по составлению школьного расписания являются программным обеспечением с отрицательной стоимостью.

Сегодня по прошествии двадцати с лишним лет с момента выше указанного анкетирования у потенциальных пользователей программ по составлению учебных расписаний — завучей школ, к таким программам, еще в большей степени и не обоснованно сформировалось стойкое негативное, а часто и агрессивное отношение.⁵

⁵Зафиксированы случаи когда завучи — в общем то, культурные и воспитанные люди, услышав слово-

Вводящая в заблуждение реклама о навязываемом «информационном пространстве школы» и «цифровизации образования» формирует представление об авторах этого пространства (вместе с рьяными цифровизаторами), как о мошенниках сбывающих тухлый товар или ворах пилящих государственный бюджет. По мнению завучей школ с большим стажем работы, *практика показывает, что данные программы могут быть использованы только в качестве инструмента для первоначальной расстановки предметов с последующей ее ручной доводкой, а так же сохранения информации и вывода ее на печать. После автоматизированного распределения предметов (программа, как правило, расставляет от 40 до 70 %) учитывать гигиенические требования к расписанию уроков практически невозможно, так как приходится не только доставлять оставшиеся не расставленные предметы, но и существенно изменять (до 60 %) автоматизированную расстановку предметов по принципу «лишь бы расставить».*

В подтверждении сказанного можно привести цитату из работы 2016 года сотрудников Белорусско-Российского университета (Могилев) [16]. *Опрос завучей школ (проводился опрос завучей могилевских школ) показал, что попытки использования программ предпринимались неоднократно, но в целом безуспешно.*

Умудренные опытом мастера своего дела, рекомендуют начинающим при составлении расписания учебных занятий пользоваться десятком – другим советом, проверенных многолетним опытом и практикой, используя при этом, вместо компьютера, макеты таблицы расписания уроков из листов картона, цветной бумаги, широкого прозрачного скотча, клея, кармашков и т.п. И они безусловно правы. Использование компьютера в режиме обычного редактора (как, всем знакомого текстового редактора) или использование программ, заводящих процесс расстановки занятий в тупиковые ситуации, когда ни одного занятия невозможно пристроить в сетку расписания теоретически, ничего кроме неоправданных затруднений, неудобств и озлобленности принести не может.

Ожидание пользователей таких программ (завучей) сомнений не вызывает. По их мнению программы составления школьного расписания после ввода всех исходных данных, должны в полностью автоматическом режиме, составлять расписание по своему качеству превосходящее расписание составленное в ручную (или уж по крайней мере не хуже). Неадекватность ожиданий завучей и получаемого ими результата от таких программ, порождает агрессивное и даже враждебное отношение пользователей к этим программам и вместе с ними к автоматизаторам «раздвигающим за небольшие (или большие) деньги информационное пространство школы» и дёбующим государственный бюджет с молитвами на устах о цифровизации образования.

Следует отметить, что разработчики программ по составлению школьного расписания в ходе «естественного отбора» поделились на три группы. Первая группа публично отстаивает точку зрения на то, что задачу автоматического расчета школьного расписания невозможно решить в принципе. И поэтому они «не будь бестолочами» даже и не пытаются этого сделать. А те кто пытается, по их мнению полные невежды. *«У нас не программа расчета школьного расписания, а редактор школьного расписания. Мы не строим расписание вместо человека, а помогаем человеку построить самостоятельно (в ручном режиме) расписание»* — с гордостью заявляют они.⁶

Вторая группа разработчиков декларирует в качестве цели — полную автоматизацию построения школьного расписания, но в своих рекламных материалах и руководствах пользователя дипломатично умалчивает о **фактическом достижении** поставленной

сочетание: — «Программа для составления школьного расписания» мгновенно переходили на матерщину.

⁶ «Не достигнув желаемого, они сделали вид, что желали достигнутого» — Мишель де Монтень. «Как получилось, так и хотели» — поговорка.

цели. «Наша программа может строить расписание в автоматическом режиме, в ручном режиме и в смешанном (полуавтоматическом) режиме» — не обманывая пользователей констатируют они. Внимание потенциальных пользователей на том, что конь может пить воду из реки, но не может ее выпить, а программа может строить (составлять) расписание в автоматическом режиме, но не может его построить (составить), эти разработчики не акцентируют. По нашему мнению это весьма взвешенная и достойная позиция, которая, несмотря на маленькую (или большую?) хитрость, может вызвать только уважение. Или, по крайней мере, не вызывает агрессивного отношения к разработчикам со стороны пользователей, честно их (пользователей) предупреждая: — «Результат не гарантирован и более того — маловероятен».

И наконец, третья группа разработчиков. «Введите исходные данные, нажмите кнопку расчет, и через несколько минут вы гарантированно получите расписание с расстановкой всех без исключения занятий. Ограничений на размерность задачи никаких. Классов хоть — 999. Преподавателей хоть — 2999. Совместителей хоть половина. Разделим класс на группы хоть до 256-ти групп. Ограничения для преподавателей и предметов любые. Каждый преподаватель выбирает для себя удобные для него рабочие дни и часы. Никаких окон у преподавателей. Занятия по предметам проводятся только в разрешенные для этих предметов часы. Неукоснительное соблюдение параллелей. Каждому предмету присваиваются баллы сложности. Точное соответствие с санитарными нормами распределения суммарной сложности предметов во времени гарантированно. Чередование в течении дня трудных и легких предметов.» — без стеснения заявляют они. Осталось только пообещать каждой «училке» по лохматому мужику, а каждому мужику по бутылке водки. Кстати говоря, на такой незатейливый ход идут разработчики наиболее беспомощных программ в плане автоматического составления расписания и к тому же неряшливых на вид (хотя есть одна, на вид весьма привлекательная, но очень беспомощная и бессовестная). Такие программы в Microsoft метко окрестили — food dog — «собачья еда».

Трудно сказать, что именно движет людьми, идущими на прямой и бесхитростный обман потребителей. Этот обман всегда становится очевидным при первом же вводе учебного плана школы (вводе исходных данных задачи) в программу. По Российскому законодательству, в соответствии со ст. 179 Гражданского кодекса РФ, сделки совершенные под влиянием обмана могут быть признаны судом недействительными, при этом обманщик возвращает обманутому все полученные деньги, возмещает обманутому реальный ущерб и в добавок еще должен перевести в доход государства такую же сумму, что им была получена от продажи программы.⁷

4 Немного о сложности решаемой задачи

Стоит сказать два слова о сложности решения задачи составления школьного расписания.⁸ Квалифицированным пользователям персонального компьютера, уверовавшим в

⁷При этом стоит отметить, что легенды и слухи окружающие российское правосудие (а точнее отсутствия правосудия, что естественно на руку мошенникам) мы здесь не обсуждаем.

⁸Здесь мы не будем останавливаться на теории NP-трудных задач, поскольку обсуждение этого вопроса увело бы читателя далеко в сторону от интересующей нас темы, а так же носило бы явно преждевременный и поверхностный характер. Заинтересованному же читателю, можно порекомендовать обратиться к пожалуй самому цитируемому в нашей стране изданию по данной теме [6]. Для полного понимания представляемой здесь публикации, под NP-трудными задачами можно понимать — практически нерешаемые задачи, хотя это не совсем точный (или совсем не точный) «перевод».

его всемогущество, не без помощи оголтелой пропаганды СМИ и фантастических историй о «неуловимых хакерах» за которыми скрывается банальное воровство денег с применением компьютера (при этом, применением весьма простеньким), кажется, что задача составления школьного расписания ничуть не сложнее задачи создания, например, качественного редактора текстов, или хорошей программы для монтаже видеофильмов, или звукового редактора. Однако, как упоминалось ранее, число исследователей⁹ так или иначе изучавших эту задачу трудно поддается подсчету. Среди них десятки докторов технических и физико-математических наук,¹⁰ сотни кандидатов наук, не только технических, но и физико-математических, не говоря уже о тысячах рядовых любителях математических головоломок, безусловно включающих в себя многочисленную армию студентов технического и физико-математического профиля обучения. Среди исследователей задачи составления школьного расписания можно упомянуть и двух академиков — В.С.Танаева¹¹ и В.С.Михалевича,¹² можно было бы назвать и зарубежных ученых с мировым именем. Кроме ученых не обходили своим вниманием задачу составления школьного расписания и выдающиеся бизнесмены¹³.

И тем не менее, не смотря на, без преувеличения, титанические усилия исследователей, говорить о полном и всеобъемлющем (или хотя бы удовлетворительном) решении задачи составления учебного расписания — до недавнего времени, не приходилось.

В качестве подтверждения сказанного приведем цитату известного отечественного математика [10].

... Так как задача составления расписаний хорошо всем известна по школьному быту, то на каждом курсе находится один или несколько студентов, обуреваемых идеей алгоритмизации составления расписания занятий. Поэтому я вынужден предупредить, что это очень сложная задача. ... Существует особая наука — теория расписаний, изучающая и систематизирующая задачи такого рода, а так же различные приближенные методы их решения (на точные методы надежды почти нет). Особое место среди них занимают эвристические методы, в которых делаются попытки описать логику и технику действий диспетчера. ...

Интересно одно наблюдение. Но прежде приведем еще одну цитату [17].

Гипотезу четырех красок можно с полным основанием назвать «болезнью четырех красок», так как она во многом похожа на заболевание. Она в высшей степени заразна. Иногда она протекает сравнительно легко, но в некоторых случаях приобретает затяжной или даже угрожающий характер. Никаких прививок против нее не существует; правда, люди с достаточно здоровым организмом после короткой вспыш-

⁹Имеются ввиду русскоязычные публикации, которых по сравнению с англоязычными не так много. Скорее всего их число не превышает общего вклада Российской Федерации в сфере высоких технологий (до 24 февраля 2022 года), который оценивался в пределах 0.4 – 0.6 % (от нуля целых четырех десятых процента до нуля целых шести десятых процента) от общемирового.

¹⁰Правда, физико-математических наук на порядок меньше.

¹¹Танаев Вячеслав Сергеевич (1940 – 2002) — белорусский математик, директор НИО «Кибернетика» НАН РБ, доктор физико-математических наук (1978), профессор (1980), действительный член Национальной академии наук Беларуси (2000). Область научных интересов: исследование операций, теория расписаний, методы оптимизации.

¹²Михалевич Владимир Сергеевич (1930 – 1994) — украинский математик и кибернетик, академик АН Украины, академик РАН (1991; академик АН СССР с 1984). Труды по теории оптимальных статистических решений, системному анализу, теоретической и экономической кибернетике. Государственная премия СССР (1981).

¹³Существует легенда [7] о том как, теперь один из самых богатых людей на планете, а тогда юный (шестнадцати летний) Билл Гейтс вместе со своим старшим другом Полом Алленом «решили» эту задачу, взамен получили за программирование машинное время и заработали 2400 долларов карманных денег.

ки приобретают пожизненный иммунитет. Этой болезнью человек может болеть несколько раз, и она подчас сопровождается острой болью, но ни одного летального исхода зарегистрировано не было. Известен по крайней мере один случай передачи болезни от отца к сыну, так что, может быть, она наследственна.

Здесь выдающийся американский математик иронизирует над старинной задачей о раскраске политической карты в четыре цвета, где страны имеющие общую границу должны быть окрашены в разные цвета. Представляется, что все сказанное им, можно отнести и на счет задачи составления школьного расписания. Так, автору этих строк вздумалось, в меру своей возможности, отследить дальнейшую карьеру людей защитивших диссертацию по соответствующей теме. Казалось бы, новосостоявшемуся ученому «сам Бог» велел обратить свои научные достижения в деньги. То есть, каким-то образом вывести свое детище на рынок, поскольку практически всегда после защиты диссертации остается некая программа или часть автоматизированной системы по составлению расписания занятий. Так вот — нет. Все известные автору случаи защиты диссертации по этой теме заканчиваются одним — после защиты, диссертант бросает эту задачу и, как правило, начинает (или продолжает) спокойную и размеренную карьеру преподавателя в вузе. Другими словами, приобретает к задаче составления учебного расписания пожизненный, стойкий иммунитет.

Заканчивая общие рассуждения о сложности решения задачи составления школьного расписания сошлемся еще на два мнения. Но прежде обратим внимание на то, кто это мнение высказывает.

Ни для кого не секрет, что некоторые школьные преподаватели информатики в припадках дидактических экспериментов поручают школьникам в качестве «домашнего задания» разработать программу составления расписания занятий для любимой школы. Школьники, естественно, засучив рукава с энтузиазмом берутся за решение этой задачи. В качестве выхлопа от данной идеи в сети Интернет можно встретить многочисленные рассуждения и теоретизирования по этому поводу выше указанного контингента. Чего только не придумывают и каких только мнений не высказывают пион Эры . . . Не меньший ажиотаж данная тема вызывает и у людей с техническим образованием в попытках автоматизировать деятельность диспетчеров любимого вуза. Но эти мнения, мягко говоря, большого интереса не представляют. Профессиональные же математики, специалисты по теории расписаний, высказываются о проблеме составления учебного расписания чрезвычайно редко. Поэтому (или тем более) их мнение на этот счет представляется весьма интересным.

Итак. Сотсков Юрий Назарович, доктор физ.-мат. наук, профессор, главный научный сотрудник Объединенного института проблем информатики НАН Беларуси, Минск, один из виднейших специалистов в области теории расписаний, автор ряда монографий по теории расписаний. В своей статье [2] в частности пишет:

. . . С математической точки зрения задача построения оптимального расписания учебных занятий является достаточно сложной, поскольку она принадлежит классу так называемых NP-трудных задач. . . . В данной статье показано, как раскраска вершин графа может применяться для построения расписания учебных занятий. . . .

. . . Задача раскраски вершин графа является NP-трудной, и, следовательно, ее обобщение, описанное в разд. 2, также является NP-трудной задачей. . . .

Далее. Лазарев Александр Алексеевич, доктор физ.-мат. наук, профессор, главный научный сотрудник Института проблем управления им. В.А.Трапезникова РАН, Москва, один из виднейших специалистов в области теории расписаний, автор ряда монографий по теории расписаний. В своей статье [5] в частности пишет:

... Задача составления учебного расписания является хорошо известной задачей комбинаторной оптимизации «Составление временных таблиц» (*timetabling*). Даже нахождение допустимого расписания является *NP-трудной* в сильном смысле проблемой. Поэтому при ее решении необходимо использовать математические методы решения задач комбинаторной оптимизации. ...

Короче: — «Сливай воду, суши весла, туши свет ...»

5 Парадоксальность рынка ПО составления учебных расписаний

Рынок ПО составления учебных расписаний, который складывался вместе с рынком любого ПО для персональных компьютеров, представляется просто уникальным,¹⁴ или уж по крайней мере удивительным, или на худой конец весьма странным. Так в чем же его уникальность или странность?

Вам когда-нибудь приходилось видеть такую рекламу: — «Купите наш пылесос, который не может всасывать пыль». Или такую: — «Все кастрюли которые мы вам можем предложить — дырявые». Или такую: — «Наш телевизор уникален — он никогда, ничего не показывает». Или такую: — «Купите у нас битые стекла, они могут пригодиться вам во время выноса из квартиры мусора».

А вот рекламу: — «Купите нашу программу для составления школьного расписания, которая не может его составить, но зато может составлять», нам приходилось видеть сколько угодно. «Ну купите, купите, купите. Наша программа может же составлять расписание (не составить). Она вам расставит *почти* все занятия, ну а уж остальные, как-нибудь сами. Выбираться из тушика, это же так интересно. Ну хоть за 15 долларов. Это же небольшие деньги, мы ведь столько работали ...».

Так сколько же стоит пылесос, который не всасывает пыль, дырявая кастрюля или телевизор, который никогда, ничего не показывает? Прежде чем ответить на этот нелегкий вопрос попробуем оценить количество потенциальных покупателей и сравнить его с количеством школ (завучей) которые уже совершили свою покупку.

Демографами установлено, что около 16% населения развитых стран являются школьниками. Именно эта цифра используется при строительстве новых школ в новых районах застройки [13]. Далее будем производить арифметические вычисления применительно к

¹⁴Утверждение про уникальность рынка ПО составления школьного расписания, пожалуй все же не верно. Очень похожая ситуация сложилась на рынке ПО автоматического перевода текстов (в частности технических, написанных человеком для человека, тексты на языках программирования, естественно, не в счет) с одного языка на другой, например, с английского на русский. Точно так же, эта задача весьма внятно была сформулирована не менее шестидесяти лет назад. Точно так же, решалась усилиями многих тысяч исследователей. До настоящего времени в попытках решить задачу перевода прилагаются фантастические усилия. Приблизительно те же результаты — близкие к нулевым. Если программе автоматического перевода не предлагать специально подобранный текст (как порой делают разработчики подобного ПО в своих демоверсиях) или примитивный текст, понять из перевода: — «Это вообще о чем?», практически не возможно.

В качестве другого примера, можно было бы привести ПО распознавания речи. Судя по программам обучения иностранному языку, кое-какие результаты в этом направлении все же имеются (отличить правильное произношение от неправильного — с трудом удастся), но до окончательного решения проблемы еще очень и очень далеко. Мечта о вводе в компьютер данных или команд с голоса (возможно за исключением примитивных случаев), все еще остается мечтой.

Третьем примером могло бы быть ПО распознавания рукописного текста (не путать с распознаванием типографского текста, это принципиально различные задачи). Многолетние усилия огромной армии исследователей пока-что увенчались практически нулевым результатом.

Российской Федерации (родина все-таки).¹⁵ Итак, население составляет примерно 140 млн. человек. Таким образом, школьников примерно 20 млн.¹⁶ Школ около 41 тыс.¹⁷. Значит среднее количество учеников в школе 490 человек. Но это среднее количество. Известно [13], что за последние 60 – 70 лет типовыми проектами школ, считались школы на 1000 – 1400 учеников. Отсюда вывод — имеется огромное количество школ с количеством учеников на много меньшим, чем наша средняя цифра — 490 человек. Очевидно, что это школы в сельской местности или в очень небольших городах (населенных пунктах городского типа). Отсюда же, более сильный вывод — огромному количеству школ, программы для составления расписания учебных занятий не нужны в принципе.

Оценить количество школ, которым такие программы не нужны в принципе, конечно же весьма трудно. Тем не менее, внимательно посмотрев на потолок увидим там цифру — 70%. Из чего следует, что 30% школ имеют количество учеников от 900 и более, а таким школам программа, которая, может не составлять школьное расписание, а может его составить, пригодилась бы. Получаем окончательную цифру — 12 тыс. школ. Вот это, пожалуй, потенциальная емкость рынка для Российской Федерации.

А что же мы имеем на сегодня в реальности? Вопрос не простой. Достоверная статистика отсутствует. Прежде всего на ум приходит одна программа, которая в середине нулевых годов для всех школ РФ была этим школам «впендюрина бесплатно». Началось разработки этой программы относится к 1998 году, а окончание (последняя версия) к 2003 году. По внешнему виду, тем более для своего времени, программа конечно же неплохая. По сравнению с другими аналогичными программами, имеет весьма логичный и продуманный интерфейс пользователя. На наше субъективное мнение, лучший пользовательский интерфейс. Однако, хотя там и имеется кнопка **Составить расписание**, программа на редкость беспомощна в плане автоматического (без участия человека) составления расписания. Она не способна решить даже тех простых подзадач, с которыми другие программы легко справляются. Судя по отзывам в сети Интернет, этой программой, несмотря на всю ее псевдобесплатность, почти никто не пользуется. Так что, будем считать ее «радиационным фоном или реликтовым излучением» не влияющим на общую рыночную ситуацию.

Поехали дальше. Поставим такой вопрос. Имеются ли на рынке программы которые могут оказать завучу хоть какую-то помощь при составлении расписания? Например, многие завучи в ручную составляют расписание в два этапа. На первом этапе по их выражению: — «Разбираются с иностранцами». Другими словами, составляют расписание для преподавателей и классов, которые делятся на группы, при изучении иностранного языка. Вторым этапом идет — все остальное. По крайней мере две программы, представленные на рынке, с выше указанным, первым этапом, справляются на зависть прекрасно. Здесь же можно спланировать время проведения элективных курсов. При этом расстав-

¹⁵Следует отметить, что расчеты могут очень сильно варьироваться выбирая в качестве примера другие страны. Так, например, Франция имеет население в два раза меньше чем РФ, однако школ там в два раза больше. Таким образом, «школо-оснащенность на душу населения» отличается в этих странах в четыре раза. Правда, такой расчет произведен без учета того, что во Франции совсем другая система образования чем в России.

¹⁶По данному показателю в последние 18–19 лет наблюдалась противоречивая динамика. На начало 2000/2001 учебного года в стране было 20553,5 тысяч школьников. На начало 2011/2012 учебного года их было 13737,8. После этого начинается их рост. На начало 2013/2014 учебного года в школах училось 13877,4 тысяч человек. В 2017 году количество школьников составило 15705,9 тысяч человек. В 2018 году школьников было уже 16137,3 тысяч.

¹⁷По данным Счетной палаты, начиная с 2001 года, сельские школы сократились почти вдвое: с 46000 до 24000, городские — на четверть, с 23000 до 18000. По данным Росстата, в 2018 году всего в России было 41349 школ.

ляется от 10 до 40 процентов занятий от общего объема учебной нагрузки всей школы. Так что, кое-какая польза от применения компьютера, оснащенного такими программами, конечно же есть. Причем одна из этих программ весьма агрессивно и настойчиво пытается достроить расписание, т.е. справиться со вторым этапом задачи. В некоторых случаях, пускай и редких, это ей удается. Другие же, при достраивании расписания до полного (финального), абсолютно беспомощны.

Так сколько же народу на сегодня пользуется ПО для составления расписания учебных занятий в Российской Федерации? Некоторые производители такого ПО, на своих сайтах публикуют информацию о своих клиентах. Правда, к этой информации стоит относиться весьма осторожно. Как было отмечено выше, некоторые производители в «припадах маркетинга» идут на весьма бесхитростный обман потенциальных клиентов. И тем не менее, отделив зерна от плевел получим цифру — порядка 1500 школ. Что составляет около 12% емкости потенциального рынка. Следовательно 88% потенциальных клиентов еще не окупены.

Теперь остается перевести количество в качество. То есть, умножить количество потенциальных клиентов на цену одной лицензии ПО. Но для этого нужно знать цену такой лицензии.

Интересно, приходилось ли читателю держать в руках толстую книжку с примерно таким названием: — «Определение стоимости программного обеспечения». А нам приходилось. На самом деле, формула очень проста. Программное обеспечение, какой бы сложности и объема оно ни было, стоит ровно столько, сколько за него платит клиент (пользователь). Ярчайшим примером тому, является операционная система Windows от Microsoft. Наверное мало кто задумывался о том, что по количеству труда, таланта, знаний и т.п., высадка человека на Луну, по сравнению с этой операционной системой, является детскими шалостями. И тем не менее, сто пятьдесят баксов на бочку, и ты легальный пользователь.

Проблема только в том, что количество потенциальных клиентов — пользователей операционной системы и программы для составления школьного расписания несопоставимо, ни в первом, ни во втором, ни в третьем приближении. Отсюда вывод: — «Не смотря на то, что некоторые просят по 15 долларов за дырявые кастрюли, программа которая действительно могла бы решить большинство проблем завучей, должна стоить дорого». Остается только ответить на вопрос: — «Что такое дорого?»

Конечно же, у каждого представления о «Дорого» свои. Но наверное, для завуча (или аналогичной должности, дорого это его месячная зарплата. Что собственно, мы и наблюдаем, или по крайней мере ранее наблюдали, в реальности. Первое время, на мировом рынке эти программы именно столько и стоили. Падение цен, как нам представляется, произошло именно из-за того, что вдруг обнаружилось — за 5000 долларов была куплена дырявая кастрюля. Теперь, те же дырявые кастрюли, предлагают по 500 долларов США¹⁸.

И наконец, перемножив количество на цену, получим примерную емкость рынка ПО для составления школьного расписания. То есть, рынок не менее денежно-емкий, чем, скажем, рынок различных систем автоматизированного проектирования в промышленности или строительстве. И к стати говоря, не менее наукоемкий. С другой стороны — не так уж это и много, если учесть, что, как сообщали некоторые СМИ в январе 2020 года, у заместителя начальника таможни РФ по энергетике изъяли 1 тонну сто долларовых купюр (100 миллионов долларов США наличными) и 18 тонн золота (слитками, хранившимися в жилом помещении).

¹⁸В Эрефии по 80 долларов.

6 «Древнеегипетский» алгоритм

Эта полукриминальная история произошла весной 2012 года. Один ученый-археолог, проработавший долгое время в Египте, обратился к своим знакомым программистам со странной просьбой. С его слов, одна старая египтянка передала ему рукопись с описанием алгоритма составления школьного расписания, который достался ей от прабабушки.

При этом, египтянка уверяла, что ее прабабушка якобы украла рукопись с описанием этого алгоритма в одной из гробниц которую раскопали археологи.

Авторство алгоритма приписывалось египетской жрице по имени Анушер. Собственно его просьба заключалась в том, что бы проверить на современном компьютере действительно ли этот алгоритм способен построить школьное расписание.



Рис. 1: Древнеегипетские пирамиды на закате солнца

Поначалу друзья его подняли на смех, расценив повествование страной женщины о якобы украденной из гробницы рукописи, как бред умолищенного. Но внимательно ознакомившись со странными записями и найдя их вполне логичными, все же решили проверить эти записи, несмотря на всю фантастичность поведанной им истории.

Предварительно скажем, что сама терминология «Древнеегипетского» алгоритма и организация древнеегипетской школы представляет отдельный исторический интерес, но поскольку данная публикация предназначена не для историков, а для современных пользователей ПК, в дальнейшем, рассказывая об этом алгоритме, будем придерживаться ныне принятой терминологии, привычной для живущего сейчас читателя.

6.1 Общий взгляд на алгоритм решения задачи

Основное отличие «Древнеегипетского» алгоритма (далее слово — древнеегипетский, будем опускать) составления школьного расписания от современных (часто используемых разработчиками программ составления школьного расписания) подходов заключается в том, что задача разбивается на части (этапы), или точнее, на ряд последовательно решаемых задач, при этом каждая решенная задача на предыдущим этапе является ограничением (исходными данными) для задачи, подлежащей решению на следующем этапе. Другими словами, применен метод декомпозиции общей решаемой задачи, часто именуемый в математической литературе, как метод — «Разделяй и властвуй».

Следует отметить, что каждая в отдельности из задач, которая последовательно решается в ходе расчета расписания, не является труднорешаемой. Это и позволяет с помощью последовательного решения серии легко решаемых задач, решить всю задачу составления школьного расписания целиком¹⁹.

Для большей наглядности всего выше сказанного проиллюстрируем алгоритм с помощью рисунка 2 на котором показаны все его основные (главные) этапы, которых, как видно из рисунка — шесть.

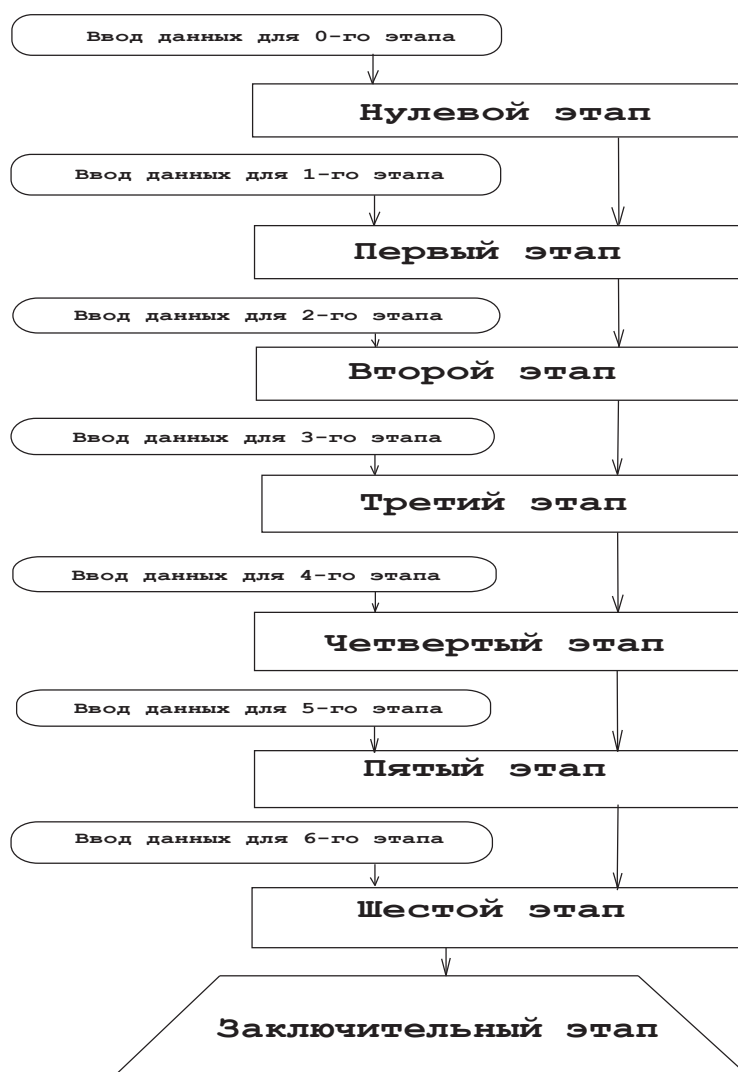


Рис. 2: Блок-схема «Древнеегипетского» алгоритма составления расписаний

Обратим внимание на то, что для решения каждой задачи по отдельности, естественно нужно ввести в компьютер (сформировать) все исходные данные для ее решения. Эти шаги (формирование исходных данных) алгоритма и содержание самих задач (этапов), будут нами подробно описаны далее.

¹⁹Слава тебе, Господи, что ты создал все нужное не трудным, а все трудное — не нужным. (Сковорода Григорий Саввич, 1722 – 1792) Сковороду Г.С. называют «первым философом Российской империи».

6.2 Тестирование алгоритма

Как будет видно из дальнейших разделов этой публикации, ничего сложного для понимания в работе, тестируемого алгоритма построения школьного расписания нет. Одна за другой решаются, связанные между собой последовательностью (порядком следования), отдельные легко решаемые (не NP-трудные) задачи до тех пор пока все они не будут исчерпаны. Тем не менее с уверенностью утверждать, что каждая из этих задач может быть решена, до проведения тестирования, не имелось никаких оснований. В отсутствие каких-либо теоретических обоснований алгоритма, что характерно для древнего Египта, проверить его работоспособность можно было только экспериментально, тем более, что именно такая задача и была поставлена ученым-археологом, наткнувшимся на странную историю об украденной рукописи. Если говорить совсем точно — об украденном из гробницы папирусе.

Вполне естественно то, что первая мысль которая пришла в голову программистам — это создать обычное приложение для операционной системы Windows. Но что такое обычное win-приложение? Будучи активированным (запущенным на выполнение) оно ожидает поступления событий от пользователя, например, ввода исходных данных, нажатия мыши и т.п. А каким образом можно эти исходные данные получить, а позже занести в программу?

Слава Богу, а точнее слава США, в то время (2012 год), мало-мальски уважающая себя школа открыла свой сайт в сети Интернет и первое что появлялось на этом сайте, не считая фотографий с различных праздничных мероприятий, так это учебный план школы. Оставалось только его скопировать и ввести в программу в качестве исходных данных для расчета расписания. Вопрос. Сколько нужно для этого времени? Практика использования ныне предлагаемых рынком программ составления школьного расписания показала, что для ввода учебного плана вместе с формированием таблицы распределения педагогической нагрузки, требуется от 8-ми до 10-ти часов, мягко говоря, кропотливого труда. Предположим, что этот учебный план введен, а таблица распределения педагогической нагрузки сформирована, и о чудо . . . , расписание построено. О чем это говорит. Ровным счетом ни о чем. Никакой гарантии того, что следующая задача будет решена — нет. Вот если бы расписание не было построено, то это говорило бы о многом, а именно, о том, что алгоритм не решает поставленной задачи. Другими словами, алгоритм в виде обычного win-приложения, в некотором смысле, практически не поддается тестированию. Как же быть?

Опять же — слава Богу, а точнее слава Microsoft, в современных версиях операционных систем Windows поддерживается так называемый режим консольного приложения. Кстати говоря, для некоторых молодых людей это является полным откровением, им никогда не доводилось видеть черных окошек с бегущими внутри этих окошек строчками текста. Действительно, это стиль больших ЭВМ из далекого прошлого и давно покинувшей сцену — MS-DOS. Многие из пользователей ПК о таких окошках просто не знают или никогда не обращали на них внимания. Для тех кто таких окошек никогда не видел или не обращал на них внимания, представим их вид на рисунке 3.

Действительно, человеку со стороны, понять, что там внутри происходит не так-то просто. Но у этих окошек есть одно преимущество. Они могут висеть на экране компьютера, производя нужные вычисления, без какого либо участия пользователя и день, и месяц, и . . . не беремся утверждать сколько²⁰. Именно это и требовалось для проверки (тщательного тестирования) алгоритма.

²⁰Скорее всего — пока не сгорит блок питания.

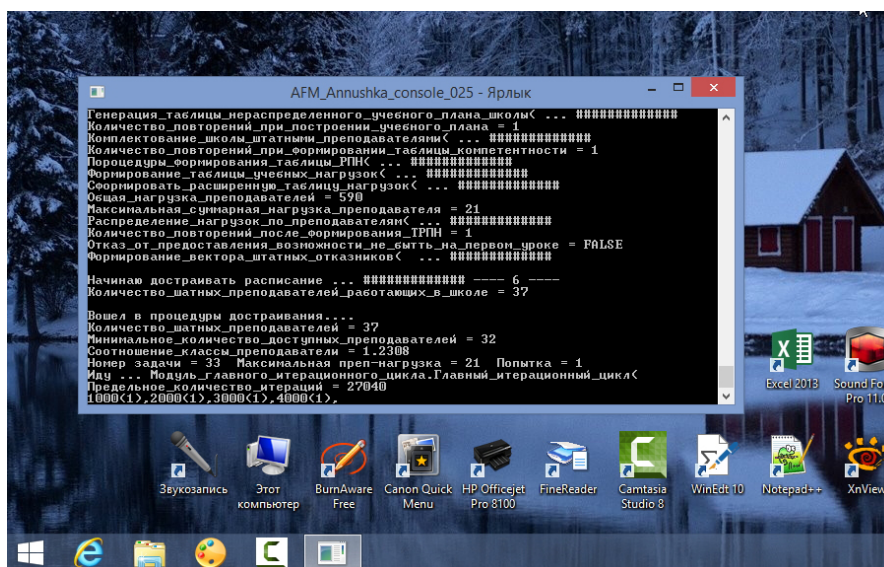


Рис. 3: Часть монитора ПК с окошком консольного приложения

Далее ход рассуждений был следующий. Написание генератора исходных данных, построенных в определенных рамках случайно, (грубо говоря, учебного плана типовой школы и таблицы распределения педагогической нагрузки) конечно же займет некоторое время, но зато, будучи один раз написанным такой генератор позволит получить неограниченное количество тестовых заданий для проверки алгоритма. Достаточно будет только лишь после решения очередной задачи передать управление этому генератору для построения нового (следующего) задания. Появится возможность получить статистически достоверные данные о качестве тестируемого алгоритма. Например, 80 процентов задач решено, а 20 нет, или наоборот. Нужно только лишь, число решаемых задач сделать достаточно большим. Именно так и предстояло поступить — консольное приложение, таким виделся выход из создавшейся ситуации.

Как говорится, быстро сказка сказывается, да не быстро дело делается. Придумать генератор исходных данных, адекватно отражающий все практические ситуации, пусть даже и типовой школы, оказалось не таким простым занятием. Но однажды сбылись мечты сумашедшие... рано или поздно... сколько веревочка не вейся... Генератор исходных данных закончен, алгоритм запрограммирован, «все ошибки исправлены»,²¹ ловушки на ошибки расставлены, проверки результатов вычислений установлены.

Вначале программе предлагалось для составления расписания небольшое количество классов — от 9 до 14 (маленькая школа). Решения выскакивали как из пулемета. При увеличении количества классов — от 15 до 21 (средняя школа) решения выстреливались быстро, но уже не как из пулемета... скорее как из пистолета. Далее. Вот она... большая школа, до четырех классов в параллели, общее количество классов от 22 до 28. Явно включились тормоза... Процесс стал напоминать ленивую утку переваливающуюся с ноги на ногу. Но одно радовало — строчка: «Количество не решенных задач = » постоянно показывала ноль.

Стало ясно. Для получения статистически достоверных данных, подтверждающих возможность решения любой разумной задачи в полностью автоматическом режиме, одного компьютера не достаточно. Небольшие арифметические расчеты показывали — что

²¹Ха, ха...

бы оперировать числами от шести и более знаков о количестве решенных задач, требуется не менее десятка компьютеров или одного, но 40-ка ядерного, которого на момент тестирования алгоритма в природе не существовало. А для десятка компьютеров (можно прикинуть количество выделяющегося от этих компьютеров тепла и издаваемый постоянный шум от вентиляторов) требуется отдельная комната. Но ничего, нас не оставишь... Десяток не десяток, а семь четырех-ядерных компьютеров, специально для проведения эксперимента, вскоре были введены в эксплуатацию.

В результате по прошествии года «насилованных действий» древнеегипетского алгоритма по отношению к достопочтенной четырех-ядерной семерки, и после десятков миллионов решенных задач, можно с уверенностью утверждать: — «Любая, без исключений, разумная, корректно заданная задача²² по расчету школьного расписания, может быть решена без участия человека в полностью автоматическом режиме». При этом, суммарное время расчета 1000 задач примерно следующее: для группы задач от 9 до 14 классов = 20 минут, для группы задач от 15 до 21 класса = 40 минут, для группы задач от 22 до 28 классов время расчета составляет от 6 до 8 часов, т.е. для этой группы в среднем, примерно, по полминуты на одну задачу.

Таким образом, более чем годовой эксперимент, по проверке (тестированию) алгоритма составления школьного расписания в полностью автоматическом режиме, без участия человека, за время которого было решено несколько десятков миллионов тестовых заданий, был с успехом завершен. Практически для всех тестовых заданий (исходных данных) было полностью построено расписание, удовлетворяющее всем ограничениям.

К ученому-археологу возник резонный вопрос: — «Почему бы о результатах проведенного эксперимента не сообщить научной общественности путем, например, публикации в каком-нибудь авторитетном научном журнале?» На что он, не без некоторого стыда сообщил, что рукопись по уверению странной женщины была украдена, не решен вопрос с авторством алгоритма и, что имеются основания опасаться в этой связи уголовного преследования²³.

На этом и порешили: — «Не буди лихо, пока оно тихо», «Г...но не трогай, оно не воняет» и т.д. Широкая научная общественность осталась в неведении.

После семилетней паузы оставления «Древнеегипетского алгоритма» без движения сотрудники ООО «АФМ-Лаборатория» вспомнили о нем. Осенью 2020 года алгоритм ими был немного «отрихтован» (внесены небольшие оптимизирующие изменения в текст программы), что позволило несколько увеличить его быстродействие (сократить время расчета). Проведено очередное тестирование, не такое масштабное как ранее, но все же несколько миллионов расписаний было построено. Ранее полученный результат был подтвержден.

После завершения такого тестирования алгоритма составления школьного расписания (в 2020 году) встал вопрос: — «И что дальше?».

Прежде всего, бросается в глаза то, что консольное приложение, никого не сможет убедить в том, что действительно решается задача составления школьного расписания... разве что, самого программиста который написал это приложение. Создать черное окошко, с время от времени появляющимися там строчками типа: — «Количество

²²Что такое разумная задача, более менее понятно, а вот что такое корректно заданная задача, вопрос очень и очень не простой. В первом приближении можно считать задачу корректно заданной, если ее решение существует. А вот вопрос: — «При каких условиях (ограничениях) решение задачи существует?» к сожалению на сегодня остается без ответа.

²³Злые языки поговаривали о том, что рукопись с «Древнеегипетским» алгоритмом была получена ученым-археологом вовсе не от старой египтянки, а похищена им из одной египетской гробницы в которой он проводил свои научные исследования.

решенных задач = 12547564» под силу даже плохо успевающему пятикласснику. Таким образом, нормальный человек, просто не поверит такой, с позволения сказать, программе, и правильно сделает. Стало ясно — обойтись без полноценного win-приложения, с тем, что бы каждый желающий смог сформировать в нем исходные данные для своей школы и смог сам убедиться в правильности работы алгоритма, никак не получится. Таким образом, ориентируясь на данные рассуждения, в конце концов все же было разработано полноценное win-приложение — программа «АФМ: Аннушка»²⁴.

6.3 Полнота модели учебного процесса

За последние 60 лет, огромных усилий по разработке программ составления школьных расписаний, как маститых ученых, так и простых энтузиастов, было использовано (разработано) огромное количество моделей организации учебного процесса, от самых примитивных до сильно запутанных. Модель организации учебного процесса, используемая в «Древнеегипетском» алгоритме и программе «АФМ: Аннушка», достаточно адекватно отражает сложившиеся педагогические традиции обучения в средней школе как у нас в стране так и за рубежом на постсоветском пространстве.

Кроме традиционных требований к организации учебного процесса программа «АФМ: Аннушка» позволяет реализовать требования, которые ранее не могли быть реализованы должным образом. Так, например, полная (сто процентная) загрузка дефицитных (специальных) помещений, встраивание в учебное расписание элективных курсов и другое.

Кроме базовых ограничений (см. далее раздел 7.1), эта модель предусматривает:

- создание (расчет, построение) расписания для школ работающих в одну смену;²⁵
- расчет расписания для шести дней по шесть уроков в день или для пяти дней по семь уроков в день, при обучении в одну смену;
- запрет для классов на проведения урока по какому-либо предмету более одного раза в день;
- возможность одному и тому же преподавателю вести разные предметы;
- ограничение нагрузки для классов разных параллелей (разных возрастов учащихся) в соответствии с санитарными нормами;
- возможность полной загрузки специальных помещений (спортивный зал, кабинет информатики, мастерские для уроков труда и т.п.);
- для совместителей и возможно администрации школы, при небольшой нагрузке, планирование проведения уроков только в назначенные ими дни;
- возможность формирования потоков из классов и резервирование времени в сетке расписания для проведения в это время элективных курсов (предметов по выбору);
- разделение классов на группы для проведения уроков по иностранному языку и/или других предметов с разделением классов на группы, в которых преподают разные преподаватели;

²⁴В честь древнеегипетской жрицы Анушер программу, на русский манер, назвали — Аннушка. По заверению ученого-археолога, Анушер — женское имя.

²⁵Хотя некоторые пользователи придумали как программу «АФМ: Аннушка» можно обмануть. Подробнее о таком способе обмана см. раздел 19.

- предоставление почти всем преподавателям методических дней при условии их невысокой недельной загруженности и небольшой нагрузке в отдельном классе;²⁶
- предоставление возможности некоторому количеству преподавателей приходить в школу ко второму уроку, в случае невысокой нагрузки у этих преподавателей.

Здесь мы очень кратко перечислили основные характеристики модели учебного процесса. Более детально с этой моделью можно ознакомиться поработав с программой демонстрирующей работу алгоритма составления школьных расписаний — «АФМ: Аннушка», специально для этого предназначенной.

6.4 Основные достоинства алгоритма и программы

Скажем здесь буквально два слова о достоинствах, далее достаточно подробно описанного «Древнеегипетского» алгоритма составления школьных расписаний, и реализующей этот алгоритм программы «АФМ: Аннушка».

В первую очередь к достоинству алгоритма стоит отнести то, что с его помощью гарантировано строится (рассчитывается) школьное расписание в полностью автоматическом режиме без участия человека (естественно не считая формирования исходных данных для решения отдельных задач), если расписание при сформированных ограничениях вообще существует.

Кроме того, обращение (работа) с программой, реализующий этот алгоритм, является предельно простой, за счет того, что вся задача разделяется на шесть главных, с точки зрения пользователя, независимых подзадач и за счет того, что каждая небольшая вводимая порция исходных данных формируется в отдельном окне. Как показывает практика, простота работы с программой по составлению школьного расписания является, наряду с полностью автоматическим его составлением, критически-важным свойством данного прикладного программного обеспечения.

И наконец, тестирование алгоритма (использование программы «АФМ: Аннушка») для пользователей — совершенно бесплатно.

7 О существовании решения математических задач

Прежде чем перейти к подробному описанию «Древнеегипетского» алгоритма и подробному описанию работы с программой «АФМ: Аннушка» остановимся на одном важном вопросе который, как правило упускают из виду начинающие составители школьных расписаний.

7.1 Примеры задач

Рассмотрим в качестве примеров две элементарные задачи.

Задача о трех карандашах. Представим себе, что на столе лежат три карандаша или какие-нибудь три палочки, схематично изображенные на рисунке 4.

Требуется из этих карандашей сложить треугольник.

²⁶Под невысокой недельной нагрузкой преподавателей подразумевается нагрузка меньше ставки, т.е. менее 18 часов в неделю. А под небольшой нагрузкой в отдельном классе — менее 5-и часов в неделю для варианта сетки расписания 6×6 и менее 4-х часов в неделю для варианта 5×7 .

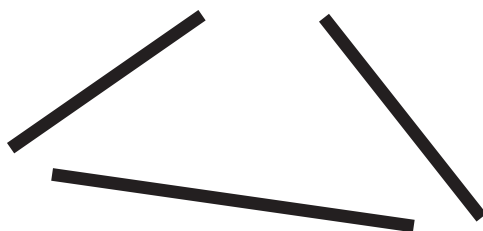


Рис. 4: Первая схема расположения на столе трех карандашей

Возникает, на первый взгляд, довольно глупый вопрос: всегда ли это можно сделать? Действительно, чего проще — зять в руки карандаши и сложить из них треугольник.

Для ответа на этот, казалось бы глупый вопрос, посмотрим на следующий рисунок 5.



Рис. 5: Вторая схема расположения на столе трех карандашей

Из этого рисунка сразу становится понятно, что для того что бы было возможно из трех карандашей сложить треугольник, нужно, что бы длина любого из трех карандашей была меньше суммы длин двух оставшихся.

Таким образом, мы приходим к выводу о том, что даже такая «смешная» задачка не всегда имеет решения. Другими словами, не всегда решение этой задачи существует.

Задача о нахождении корней квадратного уравнения. Пусть задано квадратное уравнение $ax^2 + bx + c = 0$ относительно x , где $a \neq 0$. Требуется найти такое число (числа) x при котором это уравнение обращается в тождество. Число (числа) x в этом случае называется корнем (корнями) квадратного уравнения.

Известно [9], что дискриминантом квадратного уравнения $ax^2 + bx + c = 0$ называется выражение $D = b^2 - 4ac$. Если дискриминант D квадратного уравнения $ax^2 + bx + c = 0$ положителен, то оно имеет два корня x_1 и x_2 , которые можно найти по формуле $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$; если дискриминант равен нулю, то уравнение имеет единственный корень $x = -\frac{b}{2a}$; если дискриминант меньше нуля, то уравнение не имеет ни одного корня.

Другими словами, если дискриминант D меньше нуля, то наша задача о нахождении корней квадратного уравнения не имеет решения.

Задача отыскания решения системы нелинейных уравнений [1]. Рассмотрим систему уравнений

$$\begin{cases} 4x_1^2 + x_2^2 = 4 \\ x_1 - x_2^2 + t = 0 \end{cases}$$

Здесь x_1 и x_2 — неизвестные, t — параметр.

Первое уравнение задает на плоскости x_1Ox_2 эллипс, второе уравнение — параболу. Координаты точек пересечения этих кривых дают решение системы. Если значение параметра t изменяются от -2 до 2 , то возможны, например, следующие ситуации (см. рис. 6): а) $t = -2$ — решений нет; б) $t = -1$ — одно решение; в) $t = 0$ — два решения; г) $t = 1$ — три решения; д) $t = 2$ — четыре решения.

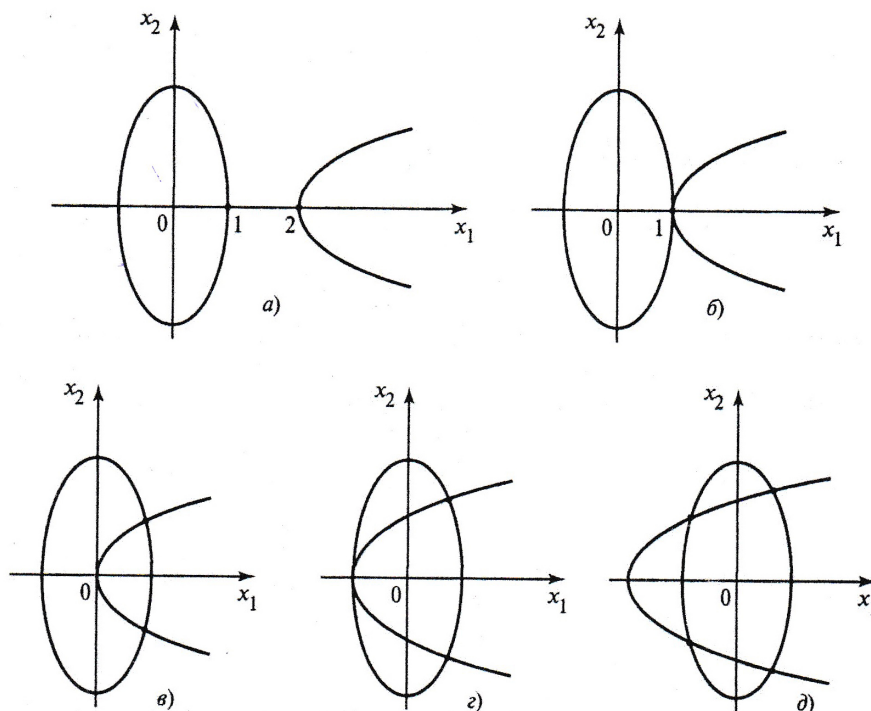


Рис. 6: Графики уравнений в зависимости от параметра t

Из этих трех элементарных примеров понятно (видно, следует), что, вообще говоря, решение любой математической задачи, как правило, начинается с вопроса: существует ли ее решение и если существует, то при каких условиях? Далее следует вопрос: если решение существует, то сколько имеется таких решений?

Закончив наши легковесные рассуждения, мы можем плавно переходить непосредственно к интересующий нас теме — существование решения задачи составления школьного расписания. Для этого рассмотрим сравнительно простую задачу.

Задача составления примитивного школьного расписания. Имеется множество $I = \{1, 2, \dots, n\}$ классов, множество $J = \{1, 2, \dots, m\}$ учителей и временное множество, т.е. множество часов (или других единиц времени), в течении которых может осуществляться обучение $K = \{1, 2, \dots, t\}$, где t — продолжительность учебной недели.

Кроме того, задана неотрицательная целочисленная матрица

$$R = \begin{pmatrix} r_{11} & r_{12} & \dots & r_{1m} \\ r_{21} & r_{22} & \dots & r_{2m} \\ \dots & \dots & \dots & \dots \\ r_{n1} & r_{n2} & \dots & r_{nm} \end{pmatrix},$$

$n \times m$, где элемент r_{ij} — нагрузка j -го преподавателя в i -том классе, $1 \leq i \leq n$, $1 \leq j \leq m$.

Требуется построить расписание, которое будем называть *примитивным школьным расписанием*, представляющее собой матрицу

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1m} \\ s_{21} & s_{22} & \dots & s_{2m} \\ \dots & \dots & \dots & \dots \\ s_{n1} & s_{n2} & \dots & s_{nm} \end{pmatrix},$$

каждый элемент которой s_{ik} показывает, в каком классе i проводит занятие учитель j в интервале времени k , т.е. элемент s_{ik} должен принимать одно из значений множества J либо ноль. Очевидно, что для соответствия матриц S и R должно выполняться условие

$$\sum_{k=1}^t \delta(s_{ik} - j) = r_{ij} \quad (i \in I, j \in J),$$

где

$$\delta(x) = \begin{cases} 1, & \text{при } x = 0; \\ 0, & \text{при } x \neq 0. \end{cases}$$

Расписание S будем называть допустимым, если оно удовлетворяет следующим дополнительным условиям:

$$s_{i_1 k_1} \neq s_{i_2 k_1}, \quad i_1 \neq i_2; \quad i_1, i_2 \in I, \quad k_1 \in K \quad (s_{i_1 k_1}, s_{i_2 k_1} \neq 0),$$

т.е. ни один учитель не может вести урок более чем у одного класса;

$$\sum_{i=1}^n \text{sign}(s_{ik}) \leq n \quad (k \in K),$$

где

$$\text{sign}(x) = \begin{cases} 1, & \text{при } x > 0; \\ 0, & \text{при } x = 0; \\ -1, & \text{при } x < 0, \end{cases}$$

т.е. число одновременно обучаемых классов не может превышать их количества.

Известно [11], что для существования, выше сформулированной задачи необходимо и достаточно выполнение следующих трех условий:

$$m \geq n;$$

$$\sum_{j=1}^m r_{ij} \leq t \quad (i \in I);$$

$$\sum_{i=1}^n r_{ij} \leq t \quad (j \in J).$$

То есть, учителей должно быть не меньше чем классов; нагрузка любого класса не может превышать продолжительность учебной недели; нагрузка любого учителя не может превышать продолжительность учебной недели. Таким образом, для нашей, правильно, выше сформулированной (поставленной), задачи — решение существует всегда.

7.2 О существовании решения задач адекватно описывающих учебный процесс

Задача составления примитивного школьного расписания отражает всего лишь, так называемые, базовые требования к искомому расписанию. Быстрый алгоритм составления такого (примитивного) расписания достаточно прост и известен, по крайней мере, уже на протяжении последних пятидесяти с лишним лет.

Однако реальный учебный процесс, принятый в современных школах, значительно затейливей, а сама постановка задачи дополняется десятком — другим требований (ограничений), среди которых могут быть, перечисленные нами в разделе 6.3.

Некоторые из этих требований, например, запрет для классов на проведения урока по какому-либо предмету более одного раза в день, не могут довести задачу до состояния не существования ее решения, другие легко.

В качестве иллюстрации сказанного рассмотрим очевидный пример.

Одно из требований к учебному расписанию, отраженному в разделе 6.3, является требование о предоставлении учителям методических дней. Предположим, что в школе имеется 20 классов и 40 учителей. Кроме того, разрешим учителям самостоятельно определять для себя такие методические дни или «удобные» для работы часы.

Предположим, что большинство (21 учитель из 40-ка) в качестве методического дня выберет четверг, поскольку четверг является баннным днем. Тогда по четвергам для проведения уроков в 20-ти классах останется только 19 учителей. Из чего очевидно следует — решение задачи при таких условиях не существует.

Аналогичная ситуация может сложиться и с «удобными» для работы часами. Кому-то из учителей захочется приходить в школу по позже, кому-то уходить из школы по раньше. В результате в некоторый час учителей окажется меньше чем классов. В этом случае вопрос о существовании решения задачи становится так же совершенно очевиден — решения нет.

А как быть, если очевидность в вопросе о предоставлении методических дней отсутствует? В этом случае придется сослаться на, уже упоминавшуюся, работу израильских математиков [18], которые пришли к выводу о том, что данный вопрос практически неразрешим²⁷.

8 Особенности пользовательского интерфейса программы «АФМ: Аннушка»

Из только что сказанного в разделе 7.2 следует, что буйную фантазию и методические припадки составителей реального школьного расписания нужно все-таки чем-то ограничивать. Такие ограничения появляются в большинстве окошек для ввода исходных данных в виде минимальных и максимальных значений формируемых параметров.

К сожалению эти минимумы и максимумы обусловлены не точным расчетом, а интуицией разработчиков программы и вычислительными экспериментами. Однако, других способов, на сегодняшний день пока не просматривается.

Есть и другая, сразу заметная особенность пользовательского интерфейса программы «АФМ-Аннушка». Заключается такая особенность в следующем.

подавляющее большинство программ, которые широко используются во всем мире, что называется в ежедневном режиме, располагают огромным меню. Можно сказать по

²⁷Задача NP-полна.

другому — большими возможностями. Это и всем известные офисные приложения, различные графические редакторы, редакторы видео, системы автоматизированного проектирования, бухгалтерские программы и т.п. Для серьезного изучения таких программ и уверенной работы с ними в большинстве случаев требуется не один месяц упорного труда.

Потенциальный пользователь программы «АФМ-Аннушка» отличается двумя особенностями. **Первая.** Уровень его квалификации в большинстве случаев — начинающий пользователь ПК. **Вторая.** Крайняя (до невозможности) «занятость».

— «Ой занята. Так занята, что ни ... ни ... некогда».

Таким образом, совершенно не приходится надеяться на то, что такой пользователь, засучив рукава, начнет изучать программу с огромным и плохо обозримым меню. Вследствие чего, разработчиками программы был предпринят отчаянный шаг. А именно, построить пользовательский интерфейс программы в виде (по типу) программы мастера. Такой тип программного обеспечения состоит в том, что программа после своего запуска начинает жесткий опрос (если хотите допрос) пользователя, требуя от него ввода исходных данных мизерными порциями которые не подлежат дальнейшему разумному дроблению. После окончания опроса, и по ходу дела проверки правильности ввода исходных данных, программа выдает требуемый результат.

Плюс такого подхода состоит в том, что программа не заставляет пользователя сильно задумываться. Минус в том, что программа лишена определенной гибкости, желательной для тех кто готов посвятить несколько месяцев своего времени для изучения всех ее премудростей.

Но на этом особенности пользовательского интерфейса не заканчиваются.

Было замечено, что пользователи не склонные к длительным и тяжелым размышлениям, предпочитают не делать что-то заново, а исправлять под себя уже кем-то сделанное. Такую возможность нужно было обязательно предоставить потенциальным пользователям программы «АФМ-Аннушка».

Для реализации этой идеи разработчики программы «АФМ-Аннушка» воспользовались генератором исходных данных из своего консольного приложения. И теперь пользователи могут, исправляя по заведомо правильному образцу (шаблону), формировать свои собственные исходные данные. Разумеется возможность «все снести к ...» пользователям была предоставлена.

В результате программу можно использовать, во-первых, для тестирования алгоритма составления расписания, во-вторых, как обучающую программу и в-третьих, как рабочую для достижения основной цели — составлению расписания занятий школы в которой работает (служит) пользователь.

На этом мы заканчиваем общие рассуждения о составлении школьного расписания и об особенностях пользовательского интерфейса программы «АФМ: Аннушка», плавно переходя к подробному и поэтому довольно скучному описанию работы с ней, напомнив читателю народную мудрость: — «Как получилось, так и хотели».

9 Начало работы с программой «АФМ: Аннушка»

9.1 Установка программы и ее иконка

Обычной практикой описания работы с программой является раздел с описанием ее установки на ПК пользователя. Но мы с целью экономии бумаги и времени читателя делать

этого не станем, надеясь на то, что пользователь и сам догадается о том, что после скачивания с сайта производителя и запуска установочного файла ему понадобится только разглядеть кнопку **Далее** и все время на нее нажимать, не меняя опций предлагаемых по умолчанию, пока на рабочем столе не появится иконка программы, изображенная на рисунке 7.

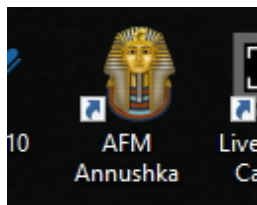


Рис. 7: Иконка программы

Напомним только лишь, что для того что бы добраться до установочного файла, который пользователь скачал на свой ПК, следует сначала кликнуть иконку **Этот компьютер**, далее кликнуть пункт **Загрузки** (см. рис. 8), и на наконец, кликнуть (произвести двойной клик мышкой) по файлу `setup_AFM_Annushka_025_x64.exe` (см. рис.9). Если пользователь скачал архив программы (заархивированный файл с расширением RAR), то его, естественно, сначала нужно будет разархивировать с помощью соответствующей программы.

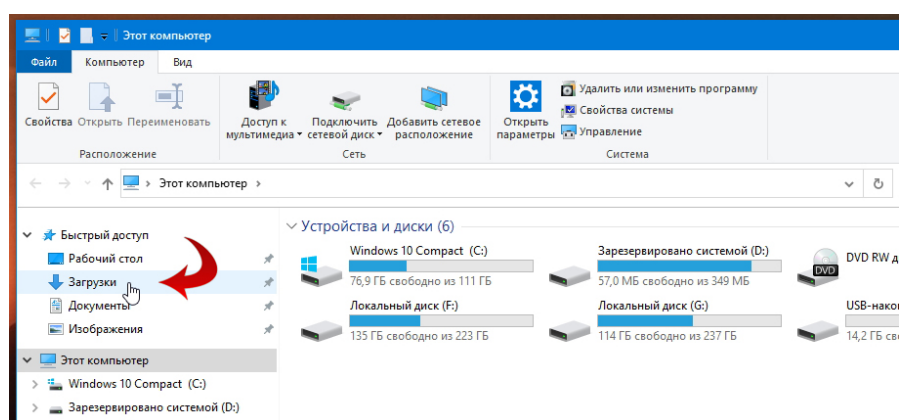


Рис. 8: Пункт Загрузки

Стоит отметить, что программа поставляется как для 32-х разрядных операционных систем Windows, так и для 64-х разрядных. Если пользователь использует 64-х разрядную операционную систему, то устанавливать нужно 64-х разрядный вариант программы. Этот вариант работает существенно быстрее. Разумеется, что и 32-х разрядный вариант в 64-х разрядной системе будет работать, но медленнее. 32-х разрядный вариант программы подготовлен исключительно для любителей операционной системы Windows XP (если таковая еще встречается).

9.2 Главное окно программы

После того, как программа скачена, установлена и запущена, на экране ПК пользователя должно появиться ее главное окно, изображенное на рисунке 10.

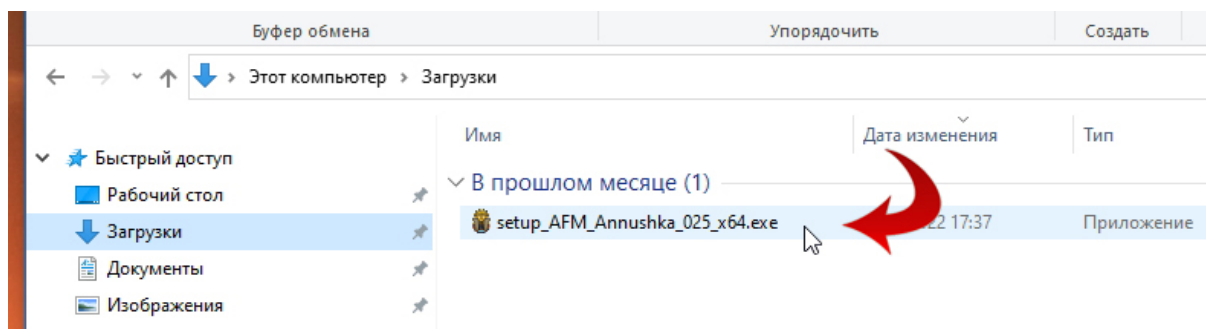


Рис. 9: Файл загрузки

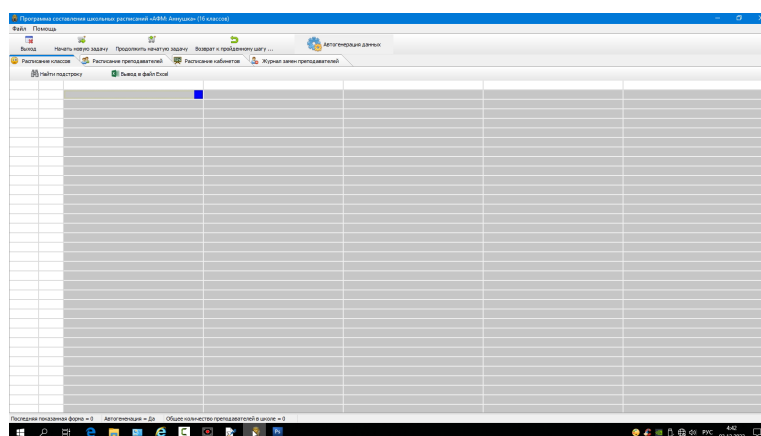


Рис. 10: Главное окно программы

Для пояснения того, что пользователь видит воспользуемся следующим укрупненным рисунком 11.

А видит пользователь, ниже заголовка окна — главное меню программы. Поскольку пока в этом главном меню почти ничего интересного для нас нет, здесь это меню комментировать не будем, а его интересные пункты оставим на потом.

Ниже главного меню — панель управления, как и принято в большинстве программ.

Розовая мечта большинства пользователей ПО это то, что бы на экране была видна только одна огромная кнопка. Здесь же мы видим аж целых пять. Назначение второй кнопки (если считать слева направо) — **Начать новую задачу**, пояснять не станем, а назначение остальных поясним далее по мере необходимости.

Ниже панели управления располагаются четыре вкладки. Нас из этих вкладок пока будет интересовать только одна — **Расписание классов**, у которой имеются две кнопки — **Найти подстроку** и **Вывод в файл Excel**. Об этих кнопках разговор пойдет позже.

Еще ниже располагается сетка расписания, в одной из ячеек которой виден синий квадратик и, наконец, в самом низу — панель состояния (она видна, если хорошенько приглядеться, к рисунку 12 и в которой можно разглядеть три не совсем коротких надписи — **Последняя показанная форма = 0**, **Автогенерация = Да** и **Общее количество преподавателей в школе = 0**).

Синий квадратик указывает на ячейку которая является текущей (имеющий фокус ввода). Если пользователь начнет кликать мышкой по сетке расписания или перемещаться по ней с помощью клавиш со стрелками, а так же крутить колесик мышки, то синий

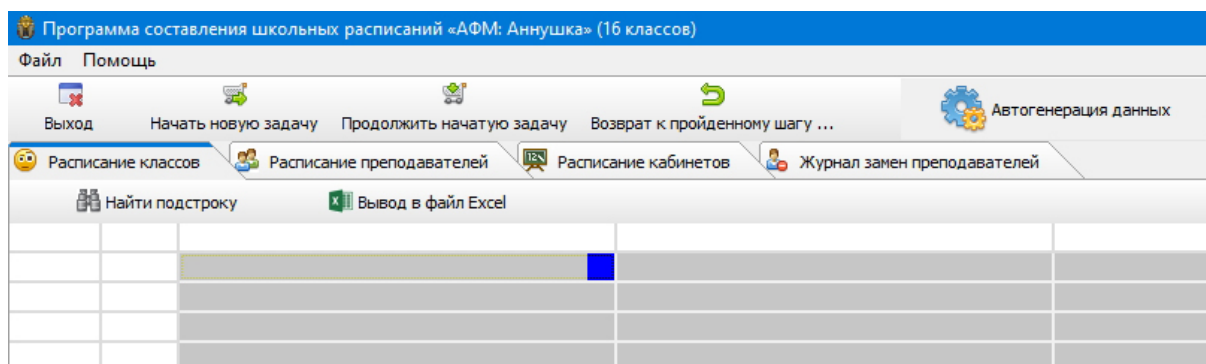


Рис. 11: Верхняя левая часть главного окна программы

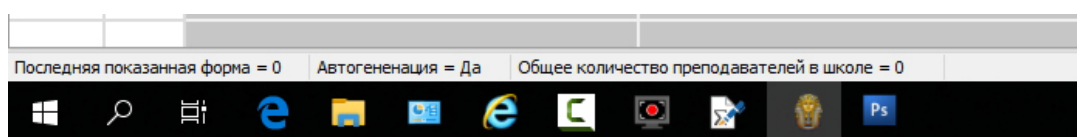


Рис. 12: Часть панели состояния

квадратик будет следовать за указателем мышки (в соответствии с назначением клавиш или направлением вращения колесика). Другими словами, перемещаться в текущую ячейку сетки расписания. Значения величин после знака равенства в первой и третьей надписи строки состояния будут меняться по мере развития процесса составления расписания, а на второй надписи, мы остановимся достаточно подробно после того, как весь процесс составления расписания будет освещен.

9.3 Кнопка панели управления Выход

На первый взгляд может показаться, что разъяснять назначение кнопки панели управления **Выход** – не стоит. Действительно — стыдобища-то какая, описывать назначение кнопки **Выход**.

В то же время, некоторые особо внимательные пользователи задались вполне резонным вопросом: почему в главном меню программы нет пункта **Сохранить**? Ведь все, кто плотно поработал с каким-нибудь текстовым редактором, впрочем как и с практически любой другой программой, привыкли к такой функции, и боясь внезапного отключения питания от своего ПК, старательно и часто ее используют, дабы проделанная ими работа не исчезла безвозвратно.

Ответом на данный вопрос может быть следующее. Разработчики программы заботясь о любимых ими пользователях решили не обременять их лишними заботами и обязали программу автоматически сохранять текущее расписание на жесткий диск ПК незаметно для них, перед каждым нажатием (во время каждого нажатия) на кнопку **Выход**. Таким образом, эта кнопка является кнопкой «двойного назначения».

Но на этом, в своей любви к пользователям, разработчики программы не остановились и решили проявить заботу о тех у кого отсутствует в комплекте с их ПК устройство бесперебойного питания. Пользователям была предоставлена возможность сохранять текущее расписание после каждого нажатия привычной для них комбинации клавиш **Ctrl+S**, хотя этого не предусмотрено в главном меню. Кроме того, для их же спокойствия, после

нажатия данной комбинации клавиш выскакивает окошко, изображенное на рисунке 13.

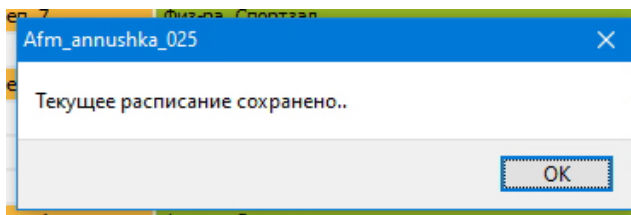


Рис. 13: Окошко сообщения о сохранении текущего расписания

Разработчики надеются на то, что такое окошко не сильно будет раздражать нетерпеливых пользователей, привыкших к быстрой работе на своем ПК.

10 Нулевой этап алгоритма

Следует понять, что прежде чем приступить к основным этапам решения задачи составления школьного расписания, придется пройти предварительный (нулевой) этап, с помощью которого вводятся (задаются) самые общие характеристики задачи, подлежащей решению.

10.1 Окно Список решаемых задач

Не долго думая ждем на кнопку панели управления — **Начать новую задачу**. На экране ПК сразу же появиться окно — **Список решаемых задач**, изображенное на рисунке 14.

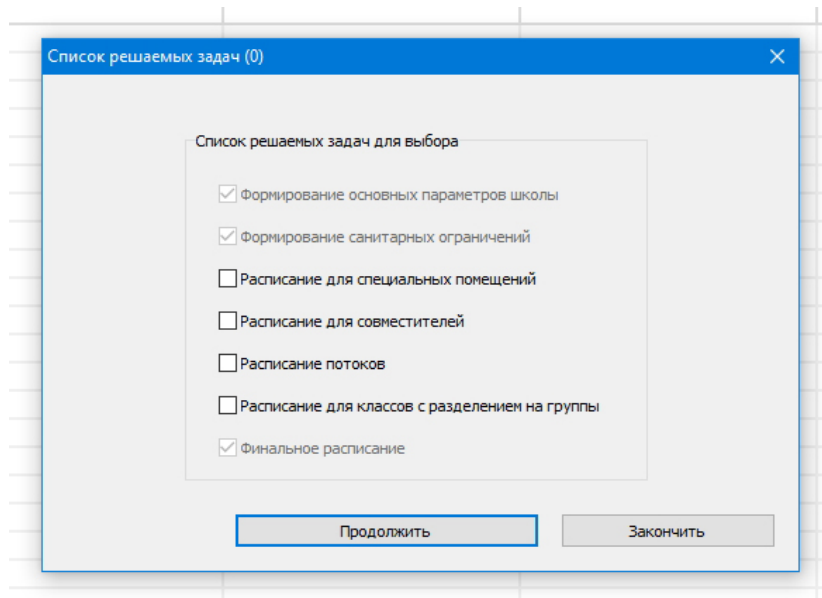


Рис. 14: Окно Список решаемых задач

Прежде всего скажем о том, что в этом окне имеются две кнопки — **Продолжить** и **Закончить**. Важно то, что такие кнопки имеются *практически во всех* окнах которые будут появляться в программе.

Назначение кнопки **Продолжить**, достаточно понятно. При ее нажатии, если все вводимые данные окна сформированы правильно и при необходимости прошли проверку, то появится очередное (следующее) окно. Назначение кнопки **Закончить** поясним позже весьма подробно. Теперь что касается вводимых исходных данных, имеющихся в данном окне.

Как видно из рисунка 14 в этом окне имеются семь пунктов:

1. Формирование основных параметров школы;
2. Формирование санитарных ограничений;
3. Расписание для специальных помещений;
4. Расписание для совместителей;
5. Расписание потоков;
6. Расписание для классов с разделением на группы;
7. Финальное расписание.

Каждый из этих пунктов является не чем иным, как отдельной, из последовательно решаемых задач при составлении расписания занятий. При этом первый, второй и последний пункты уже отмечены галочкой. Эти пункты находятся в неактивном состоянии (т.е. пользователю запрещено их редактировать и соответствующие окна обязательно появятся на экране ПК при составлении расписания занятий).

Первый пункт запрещено редактировать поскольку без его наличия задача составления расписания теряет свой смысл. Второй пункт запрещено редактировать поскольку при составлении школьных расписаний необходимость указывать санитарные запрещения возникает практически всегда, а последний пункт запрещено редактировать поскольку хоть какое-то расписание программа должна построить (хоть как-то или чем-то завершить свою работу).

Оставшиеся четыре пункта (задачи) пользователь программы может редактировать. То есть выбрать — будут такие задачи решаться или нет. О самих задачах более подробно будет сказано в следующих разделах.

И наконец, о чем уже наверное читатель и сам догадался, для выбора соответствующей задачи пользователю необходимо кликнуть мышкой по соответствующему пункту, после чего в квадратике этого пункта появиться галочка.

Поскольку нам предстоит познакомиться со всеми окнами программы, предназначенными для формирования исходных данных, будем считать, что в данном окне пользователем отмечены галочками (выбраны) все пункты.

10.2 Окно Базовые параметры школы

Итак, после того как пользователь решил для себя какие задачи ему (программе) предстоит решить и нажал в текущем окне на клавишу **Продолжить**, появиться новое (следующее по порядку) окно — **Базовые параметры школы**, изображенное на рисунке 15.

Мы видим, что в этом окне (точно так же, как и во всех последующих) программа уже автоматически подставила на соответствующие места исходные данные, полученные ей случайным образом. Но при этом у пользователя имеется возможность исправить предлагаемые данные.

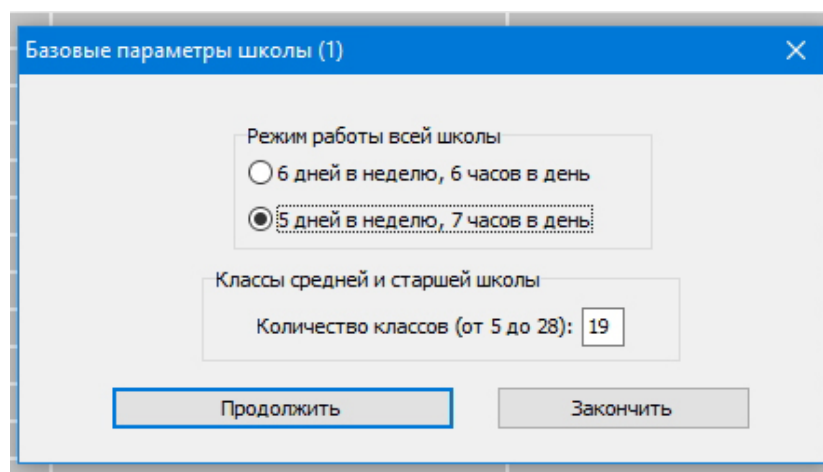


Рис. 15: Окно Базовые параметры школы

В частности, можно с помощью клика мышкой поменять вариант сетки расписания — 5 дней в неделю, 7 часов в день, на вариант — 6 дней в неделю, 6 часов в день. Кроме того, можно предлагаемое, в данном случае, программой значение Количество классов равное 19-ти изменить на нужное пользователю. При этом, вводимое значение может находиться в интервале от 5 классов до 28-ми.

После того как изменения будут корректно внесены (или без внесения изменений) пользователь может нажать на кнопку Продолжить. Понятно, что если пользователь попытается игнорировать ограничения по количеству классов, то программа не оставит этого без внимания и аккуратно (можно сказать — тактично и бережно) обругает его с помощью окошка, изображенного на рисунке 16.

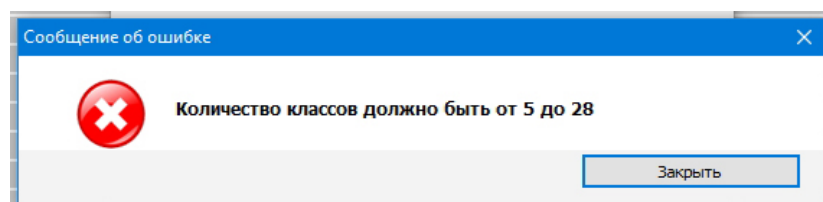


Рис. 16: Окошко Сообщение об ошибке

При вводе значения количества классов важно помнить, что программа, по умолчанию, предназначена для составления расписания занятий одновременно в средней (2-я ступень образования) и старшей (3-я ступень образования) школе, т.е. для классов начиная, с пятого (начало средней школы) и заканчивая одиннадцатым классом (конец старшей школы).²⁸ Причем одновременно в средней и старшей школе. Таким образом, все параметры расписания школы (преподаватели, учебные предметы и т.д.) будут одновременно рассчитываться как для средней так и для старшей школы, учитывая то обстоятельство, что для всех трех ступеней образования занятия проходят в одном и том же здании. Следовательно, классы начальной школы (с 1-го по 4-й) останутся за пределами внимания программы.

²⁸Имеется ввиду система среднего образования принятая на сегодня в Российской Федерации.

10.3 Окошки сообщений об ошибках

Посмотрим еще раз внимательно на рисунки 15 и 16. На рисунке 15 хорошо видно, что пользователь предупрежден о том, что в качестве значения поля **Количество классов** могут быть числа от 5 до 28. Представленное на рисунке 16 окошко **Сообщение об ошибке** фактически дублирует уже имеющуюся у пользователя информацию. Таким образом, для такого пользователя рисунок 16 (точнее говоря окошко **Сообщение об ошибке**) не представляет большого интереса, кроме того, программа должна реагировать на его неправильные действия всегда.

При разработке программы была поставлена следующая задача. Представим себе, что цель пользователя не в том, что бы протестировать алгоритм составления учебного расписания, а в том, что бы «завалить программу» при помощи ввода (формирования) абсурдных исходных данных. Естественной реакцией разработчиков на такое предположение (а это предположение нужно иметь ввиду при разработке любой программы), оказалась задача тщательной проверки всех вводимых пользователем исходных данных. Как результат, количество окошек с сообщениями об ошибках пользователя перевалило за весьма и весьма внушительное число.

Таким образом, с одной стороны, программа предусматривает обругивание пользователя обильно, часто и смачно, а с другой стороны дисциплинированного и дружески настроенного пользователя конкретный вид такой ругани мало интересует. Более того, есть еще и третья сторона, а именно. Подробное описание всех окошек об ошибках пользователя может удвоить (а то и больше) размер предлагаемого к прочтению текста.

Опираясь на выше представленную логику было принято решение о том, что бы исключить из настоящего описания упоминания о том, что программа может среагировать сообщением об ошибке пользователя при вводе им неверной информации. То есть, само сообщение об ошибке будет являться своего рода руководством пользователя.

Исключения из этого правила будут сделаны только в тех случаях, когда пользователя подобные сообщения могут серьезно озадачить или поставить в тупик.

Только что представленными двумя окнами нулевой этап составления школьного расписания заканчивается.

11 Первый этап алгоритма

Первый этап алгоритма заключается в решении задачи установки запретов на проведение занятий в соответствии с санитарными нормами (если этот процесс вообще можно назвать задачей). Здесь имеется ввиду — недельная нагрузка для классов. Например, для РФ и многих других стран на постсоветском пространстве, при шестидневке для 11-х классов она составляет 36 часов (запретов нет), для 5-х классов 32 часа (четыре запрета). При пятидневке для 11-х классов 34 часа (один запрет), для 5-х классов 30 часов (пять запретов). Основным требованием решения этой задачи является установка запретов на проведение занятий в последний урок и, как правило, в последние дни недели (естественно, что если такие запреты вообще необходимы). Таким образом, построенное расписание на данном этапе состоит из множества временных участков (временных моментов — день, урок) в которые нельзя назначать (ставить) нагрузку, предусмотренную учебным планом школы.

Ниже будут представлены все необходимые исходные данные для прохождения этого этапа алгоритма.

11.1 Окошко Количество параллелей в школе

На первый взгляд очередное появившееся, после нажатия кнопки **Продолжить**, окошко — **Количество параллелей в школе**, изображенное на рисунке 17, может показаться лишним и совершенно ненужным.

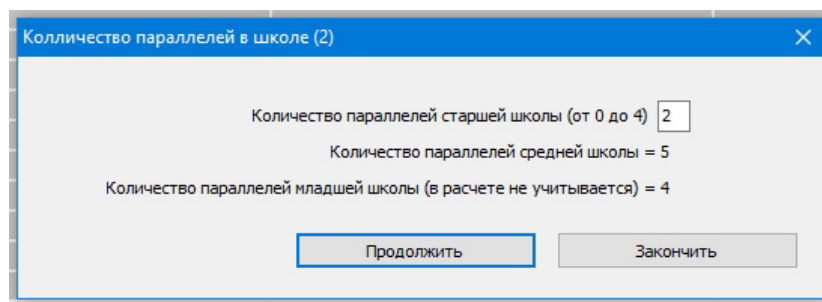


Рис. 17: Окошко Количество параллелей в школе

Появление этого окошка объясняется тем, что вводимые в нем данные могут существенно расширить, если можно так выразиться, диапазон применения программы. Можно выразиться и по другому. Программа предлагает пользователю определиться: для какой школы составляется расписание? Одиннадцатилетки, двенадцатилетки и т.п.

В окошке мы видим количество параллелей (количество лет обучения) в младшей школе, средней школе и старшей школе. Данные для младшей школы и средней школы редактированию не подлежат. Причем младшая школа вообще к составлению расписания в описываемой программе не имеет отношения. Количество лет обучения (количество параллелей) в старшей школе может быть отредактировано.

Так, если мы в поле **Количество параллелей старшей школы** введем цифру 0, то получим школу — девятилетку, если цифру 1, то десятилетку (Советский стандарт времен правления дорогого Л.И.Брежнева), если цифру 2, то теперешней Российской стандарт (одиннадцатилетка), если цифру 3, то Украинский стандарт (двенадцатилетка), если 4 — Итальянский стандарт обучения в старшей школе.

По умолчанию программа в поле ввода **Количество параллелей старшей школы** помещает цифру 2. Это значение ориентированы на Советскую (ныне Российскую) систему среднего образования, используемую в настоящее время, и в некоторых постсоветских странах. Основное отличие Советской системы среднего образования, например, от Европейской заключается в том, что учащиеся начиная с первого и до самого старшего класса обучаются в одном и том же здании. Данный подход к образованию прежде всего обосновывается экономическими соображениями (грубо говоря — унижительной бедностью, как следствием основ конституционного строя — воруют все, кто может).

В отличие от такого положения дел, в большинстве стран Европы, обучение для каждой ступени среднего образования (1-й, 2-й и 3-й) проводится в отдельных (разных) зданиях. Вопрос о том, хорошо это или плохо здесь не обсуждается. Возможно, что когда-нибудь в «Прекрасной России будущего» вместо танков, пушек, бомб, беспилотников и истребителей, начнут строить просторные школьные здания для всех детей, и тогда может возникнуть необходимость рассчитывать расписания отдельно для каждой ступени среднего образования, размещающейся в отдельном здании.

Поскольку «глупая программа» не в состоянии сама догадаться какой класс, в таком случае, является самым старшим (тринадцатый или девятый, как, например в Италии)

и какое количество параллелей имеется в школе (школьном здании), ее необходимо, что называется, «ткнуть носом» (т.е. указать эти данные в явном виде). Обратим внимание на то, что конкретное значение самого старшего класса программа незаметно для пользователя вычисляет сама, как сумму параллелей всех ступеней образования.

Для пользователя же, должно быть ясным, что программа по умолчанию предполагает — в школе имеются все параллели без пропусков и исключений. То есть, случай когда какая-либо из параллелей отсутствует, например, шестые или девятые классы, не рассматривается. Чисто теоретически, указанное ограничение несколько снижает область применения программы, но практически, случай с отсутствием параллели вряд ли может встретиться в жизни (в практике использования данной программы). При этом пользователь может легко исправить данное ограничение в «ручном» режиме путем несложных манипуляций с автоматически полученными данными.

Подчеркнем еще раз, что само понятие *параллель обучения* является чисто декоративным понятием и служит лишь, либо для автоматической генерации исходных данных, либо для удобного ввода (формирования) таких исходных данных. Дистиллированный (химически чистый) алгоритм составления школьного расписания понятия не имеет ни о каких параллелях и различает классы только лишь по их номеру в общем списке, что дает большой простор для фантазии при редактировании (изменении) номеров параллелей и названий классов. Алгоритму интересны только ячейки сетки расписания куда запрещено ставить нагрузку, а это либо запреты санитарных ограничений, либо уже распределенная нагрузка.

11.2 Окно Список классов с параллелями

Следующее окно, изображенное на рисунке 18, как обычно уже заполнено некими автоматически созданными данными.

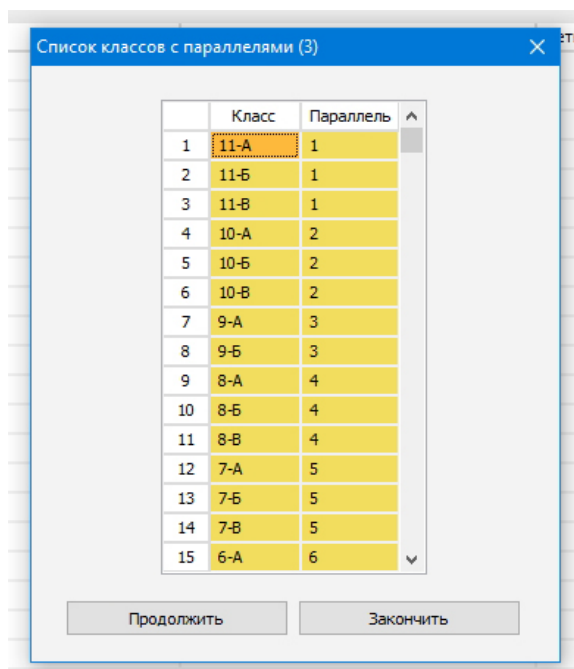


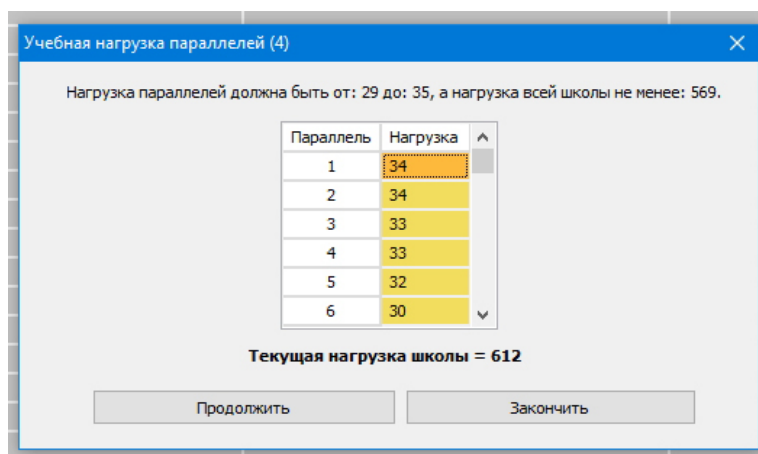
Рис. 18: Окно Список классов с параллелями

Эти данные, как обычно,²⁹ можно исправить пользователю в соответствии с решаемой им задачей. При исправлении данных этого окна пользователь должен руководствоваться тремя правилами:

1. Название класса представляет из себя обычный текст, длина которого не может превышать 7-ми символов;
2. У самых старших классов номер параллели должен быть равен 1, у самых младших, должен быть равен количеству параллелей в школе;
3. Номера стоящие в верхних ячейках, не могут быть больше номеров параллелей, стоящих в нижних ячейках. То есть, в нижней ячейке номер может быть либо больше, либо таким же (программа за выполнением этого правила следит).

11.3 Окно Учебная нагрузка параллелей

Для следующего окна, изображенного на рисунке 19, программа предлагает не случайно полученные данные, а нагрузку параллелей, предусмотренную санитарными нормами принятыми в Российской Федерации.



Параллель	Нагрузка
1	34
2	34
3	33
4	33
5	32
6	30

Текущая нагрузка школы = 612

Нагрузка параллелей должна быть от: 29 до: 35, а нагрузка всей школы не менее: 569.

Продолжить Закончить

Рис. 19: Окно Учебная нагрузка параллелей

Поскольку, как уже отмечалось, программа итак «глуха» и «бестолкова», а если ее еще начать путать количеством параллелей, то ситуация может дойти до непредсказуемых последствий и необратимых изменений. Поскольку она не в состоянии читать мысли пользователя, пользователю предоставляется возможность данные о нагрузке параллелей скорректировать в соответствии с санитарными нормами конкретной страны (административной территории) для соответствующих возрастных групп или в соответствии со здравым смыслом. При этом ответственность за правильность потребляемой программой информации возлагается только на пользователя и от него требуется повышенного внимания.

Редактирование значений нагрузок осуществляется стандартным способом, но в рамках неприятной (можно сказать противной) надписи — **Нагрузка параллелей должна**

²⁹О том, что все окна при своем появлении заполняются тестовыми данными, которые пользователь может исправить мы больше напоминать не будем.

быть от: 29 до: 35, а нагрузка всей школы не менее: 569. Сразу возникает два вопроса:

1. Откуда взялись эти цифры?
2. Зачем нужны такие ограничения?

Ответ на первый вопрос достаточно прост. Цифра 29, это константа принятая из тех соображений, что о здоровье младших школьников государство серьезно задумывается. Цифра 35 равна часовой продолжительности учебной недели (в случае сетки расписания 6×6 эта цифра примет значение равное 36-ти). Цифра 569, не что иное, как результат некоего бесхитростного вычисления, которая будет каждый раз меняться в зависимости от стандартных (заранее известных программе) значений величины нагрузки параллелей, количества классов в каждой из параллелей и количества классов во всей школе.

Теперь, зачем нужны такие ограничения. Как высказался один из программистов ООО «АФМ-Лаборатория»: — «Что бы жизнь медом не казалась»³⁰. Ну, а если серьезно, то такие ограничения вызваны некими соображениями по поводу прогноза дальнейшего существования решения предстоящих задач. Об этих соображениях мы здесь, голословно и бездоказательно рассуждать не станем.

11.4 Окошко Формирование санитарных ограничений для классов

Появление следующего окошка зависит от того, была ли выбрана задача **Расписание для специальных помещений** из списка решаемых задач на нулевом шаге (см. рис. 14).

Если пользователь не собирается рассчитывать расписание для специальных помещений, то на экране ПК появиться окошко, изображенное на рисунке 20, которое информирует его о том, что уже можно решать задачку **Формирование санитарных ограничений для классов** и предлагает воспользоваться кнопкой **Сформировать**.

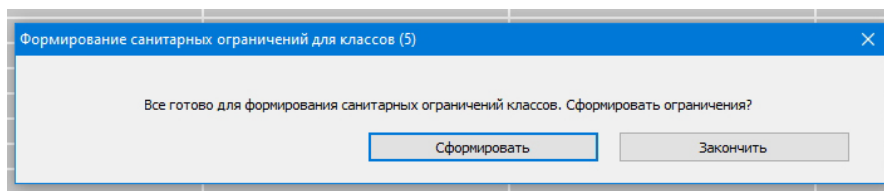


Рис. 20: Окошко Формирование санитарных ограничений для классов

Если пользователь нажмет кнопку **Сформировать**, то окошко моментально изменит свой вид, на представленный рисунком 21, а кнопка **Сформировать** превратиться в кнопку **Продолжить**.

Но это еще не все изменения. Внимательный пользователь заметит, что некоторые ячейки сетки расписания окрасятся красным цветом. Проницательный же пользователь сообразит, что красным цветом окрашены те ячейки, в которых не могут быть помещены данные о проводимых занятиях в связи с ограничениями, накладываемыми санитарными правилами, нормами или здравым смыслом.

Если пользователь выбрал задачу **Расписание для специальных помещений**, то вместо окошка, изображенного на рисунке 20 появится окошко, изображенное на рисунке 22,

³⁰Разработчики ПО народ весьма специфический. Действительно: что будет с человеком если его на протяжении многих лет по пятьдесят раз в день (или около того) будут называть идиотом?

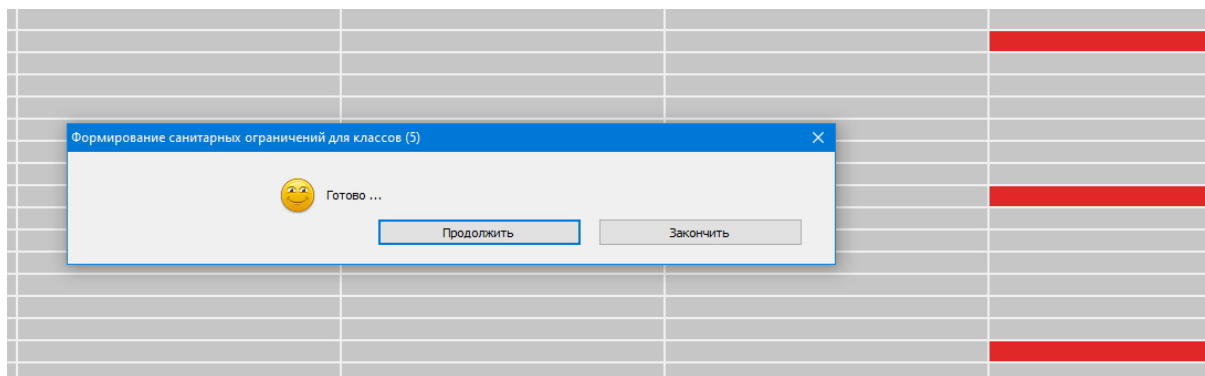


Рис. 21: Окошко после расстановки санитарных ограничений

которое информирует пользователя о том, что санитарные ограничения будут сформированы одновременно (вместе) с решением этой задачи.

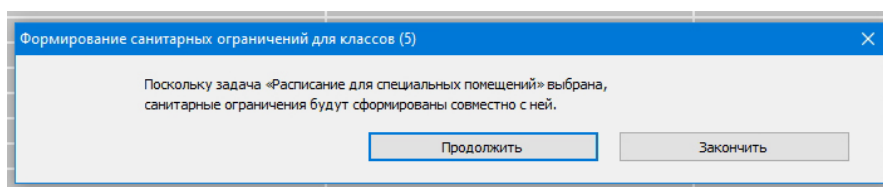


Рис. 22: Окошко Формирование санитарных ограничений для классов

Внимательный читатель наверное уже догадался о том, что программа в данном случае фактически пропускает первый этап алгоритма откладывая его на потом. Совсем точно говоря, не пропускает, а переносит его для выполнения вслед за вторым этапом. Другими словами меняет последовательность расчета расписания. Отказаться совсем от расстановки санитарных ограничений программа не позволяет, если не считать возможности в окне **Учебная нагрузка параллелей**, изображенном на рисунке 19, в правую колонку всюду (во все строчки) поместить число равное продолжительности учебной недели — 35 или 36 в соответствии с выбранной сеткой расписания.

Естественно, что критически настроенный читатель сразу же поставит вопрос ребром: зачем и почему так было сделано? Попробуем на него ответить.

Задача расстановки санитарных ограничений в соответствии с ранее сформулированными требованиями (правилами), если ее решать на чистой (пустой) сетке расписания, является даже не простой, а совершенно банальной. При этом расстановка получается идеальной. Но тогда следующую задачу — **Расписание для специальных помещений**, которая должна решаться на втором этапе алгоритма, придется решать с уже полученными ограничениями. Такие ограничения чаще всего приводят к тому, что не удастся загрузить специальные помещения на 100%, поскольку в данном случае решения со сто процентной загрузкой может просто не существовать.

Как следствие, возникает вопрос: что важнее — идеальная расстановка санитарных ограничений или сто процентная загрузка специальных помещений?

Применительно к реалиям Российской Федерации (удушающая бедность образования), приоритет был отдан полной загрузке специальных помещений, ну а уж санитарные ограничения будут расставлены какнибудь, пусть даже и не совсем идеально.

На этом первый этап алгоритма, фактически подготовка к составлению школьного расписания, заканчивается. Раскраска ячеек сетки расписания при одновременном, на взгляд пользователя программы, решении задач о загрузке специальных помещений и расстановке санитарных ограничений будет показана в следующем разделе.

12 Второй этап алгоритма

Второй этап алгоритма заключается в решении задачи составления расписания для специальных помещений. Под специальными помещениями школы (в отличие от помещений общего назначения) имеются ввиду помещения предназначенные для проведения там только какого-то одного предмета, требующего для обучения специального оборудования. Например, физкультуры — спортзал, оснащенный спортивным инвентарем, информатики — компьютерный класс, оснащенный компьютерами, труда — мастерская, оснащенная необходимыми станками (швейными машинками) и т.п. Такие помещения имеются в каждой школе и, как правило, являются дефицитными. Другими словами их постоянно не хватает для качественного обучения, имеющегося контингента учащихся. Поэтому основным требованием для этой задачи является — стопроцентное использование подобных помещений. То есть, если школа работает шесть дней в неделю и там каждый день проводится по шесть уроков, то в специальном помещении за неделю должно пройти 36 занятий ($6 \times 6 = 36$). Для пятидневки с семью уроками, каждую неделю загрузка должна составлять 35 занятий ($5 \times 7 = 35$).

Построенное на этом этапе расписание представляет из себя тройки — время проведения занятия, класс, специальное помещение. Если говорить совсем точно, то из четверок, поскольку за специальным помещением однозначно следует предмет. Преподаватели для учебных классов, которые проводят занятия в специальных помещениях, на этом этапе алгоритма пока не назначаются, они будут назначены позже.

Теперь об исходных данных, необходимых для решения этой задачи.

12.1 Окно Список специальных помещений

Итак, лихо расправившись со всеми предыдущими окошками по формированию исходных данных пользователь может приступить к разделу «Древнеегипетского» алгоритма — **Расписание для специальных помещений**. Разумеется если в свое время, в окне **Список решаемых задач**, напротив этого раздела им была поставлена галочка.

Данный раздел начинается с окна **Список специальных помещений**, которое изображено на рисунке 23.

Как не трудно догадаться из рисунка 23, количество специальных помещений ограничено интервалом от 1 до некоторого числа. Величина этого числа зависит от количества классов для которых ведется расчет. Его минимальное значение для небольшого количества классов составляет — 2, а максимальное, для большого по меркам программы количества — 4. При этом, во время редактирования данных пользователь должен помнить о том, что длина названий помещений и предметов не может превышать значения в 15 символов.

Еще раз напомним, что специальными помещениями в рамках терминологии принятой в программе, являются помещения оснащенные специальным оборудованием и которых не хватает для нормального ведения учебного процесса, а поэтому их заполненность должна приближаться к предельному значению. Кроме того, следует иметь ввиду, что специальное помещение может быть логическим понятием, включающим в себя, два

Список специальных помещений (6)

Количество специальных помещений (от 1 до 4): 4

	Помещение	Предмет
1	СпецПом_1	СпецПредм_1
2	СпецПом_2	СпецПредм_2
3	СпецПом_3	СпецПредм_3
4	СпецПом_4	СпецПредм_4

Продолжить Закончить

Рис. 23: Окно Список специальных помещений

физических помещения. Например, «Первый компьютерный класс» и «Второй компьютерный класс», что должно быть отмечено (указано, обозначено) в колонке с названием Помещение.

Такое указание является обычным текстом для которого не существует никаких особых правил оформления поскольку не влияет на алгоритм расчета расписания. Логическое специальное помещение воспринимается программой как один неразделимый объект. Таким образом, разделение учеников, например, по компьютерным классам нужно будет осуществлять «вручную».

12.2 Окно Распределение нагрузки классов в специальных помещениях

Очередное появляющуюся окно которое подлежит редактированию и предназначенное для распределения нагрузки классов среди специальных помещений представлено на рисунке 24.

Распределение нагрузки классов в специальных помещениях (7)

	Помещение	Предмет	Нагрузка	11-A	11-B	11-В	11-Г	10-A	10-B	10-В	10-Г	9-A	9-B	9-В	9-Г	8-A	8-B	8-В	8-Г	7-A	7-B	7-В	7-Г
1	СпецПом_1	СпецПредм_1	35	1	0	2	0	0	2	1	2	2	1	2	2	3	0	0	1	0	2	1	0
2	СпецПом_2	СпецПредм_2	35	1	2	1	3	2	2	0	2	3	2	0	1	2	3	1	0	1	2	0	1
3	СпецПом_3	СпецПредм_3	35	1	1	2	1	3	1	1	2	2	1	1	1	0	1	1	1	3	0	3	
4	СпецПом_4	СпецПредм_4	35	2	0	1	0	1	1	3	1	3	0	0	2	2	3	1	0	1	3	1	1

Распределить нагрузку классов в спортивных залах Продолжить Очистить Закончить

Рис. 24: Окно Распределение нагрузки классов в специальных помещениях

В процессе редактирования таблицы, представленной на рисунке 24, ее ячейки могут быть окрашены тремя цветами — красным, желтым или зеленым, что можно видеть на рисунке 25.

Окраска ячеек красным цветом означает, что нагрузка помещения для представленного примера превышает свой предел, окраска желтым цветом означает, что нагрузка помещения достигла своего предела и не может быть увеличена. Зеленый цвет означает,

Распределение нагрузки классов в специальных помещениях (7)

	Помещение	Предмет	Нагрузка	11-A	11-B	11-В	11-Г	10-A	10-B	10-В	10-Г	9-A	9-B	9-В	9-Г	8-A	8-B	8-В	8-Г	7-A	7-B	7-В	7-Г
1	СпецПом_1	СпецПредм_1	36	1	0	2	0	0	2	1	2	2	2	2	2	3	0	0	1	0	2	1	0
2	СпецПом_2	СпецПредм_2	35	1	2	1	3	2	2	0	2	3	2	0	1	2	3	1	0	1	2	0	1
3	СпецПом_3	СпецПредм_3	34	1	1	2	1	3	1	1	2	1	1	1	1	0	1	1	1	3	0	3	
4	СпецПом_4	СпецПредм_4	35	2	0	1	0	1	1	3	1	3	0	0	2	2	3	1	0	1	3	1	1

Распределить нагрузку классов в спортивных залах

Продолжить

Очистить

Закончить

Рис. 25: Таблица с окрашенными ячейками в три цвета

что в некоторых ячейках еще можно, имеющиеся значения увеличить. При этом общая величина нагрузки помещения по мере редактирования таблицы постоянно отражается в столбце **Нагрузка**.

Максимальное значение нагрузки отдельного класса в специальном помещении, не может превышать значения равного трем. Естественно, что при наличии строк таблицы окрашенных красным цветом, после нажатия кнопки **Продолжить** программа вернет пользователя к дальнейшему редактированию таблицы, и будет его возвращать до тех пор пока не останется ни одной строки красного цвета.

Если пользователю не нравится исправлять заранее случайно помещенные данные в таблицу, то он может нажать на кнопку **Очистить**, и начать формирование таблицы, что называется «с чистого листа».

Такой вариант развития событий иллюстрируется рисунком 26.

Помещение	Предмет	Нагрузка	11-A	11-B	11-В	11-Г	10-A	10-B	10-В	10-Г	9-A	9-B	9-В	9-Г	8-A	8-B	8-В	8-Г	7-A	7-B	7-В	7-Г
1 СпецПом_1	СпецПредм_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2 СпецПом_2	СпецПредм_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3 СпецПом_3	СпецПредм_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4 СпецПом_4	СпецПредм_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Buttons:

Рис. 26: Таблица нагрузок классов очищенная от данных

Обратим внимание читателя на то, что если после окончания пользователем редактирования таблицы, какое-либо значение в столбце с заголовком **Нагрузка** окажется нулевым, то программа в этом случае вернет его для дальнейшего формирования нагрузок классов. Это продиктовано тем, что решение задачи с такими исходными данными представляется совершенно бессмысленным.

Наверное внимательный читатель уже обратил внимание на то, что на предыдущих рисунках имеются не три обычные кнопки, а четыре. И четвертой кнопкой является кнопка — **Распределить нагрузку классов в спортивных залах**.

Если нажать на эту кнопку, то появится окошко **Нагрузка классов по физкультуре**, изображенное на рисунке 27.

Пользователь программы может отредактировать нагрузку принятую в его школе.

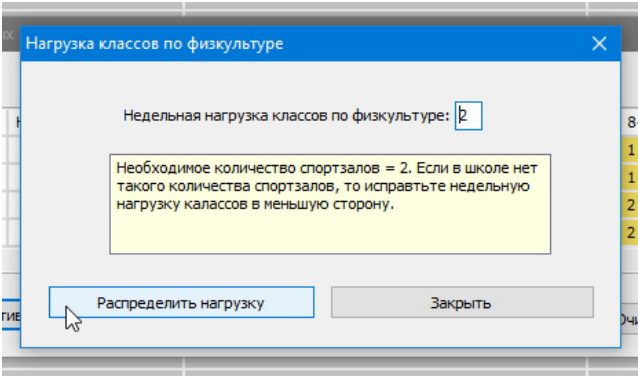


Рис. 27: Окошко Нагрузка классов по физкультуре

При этом в поле вывода, окрашенного в светло-желтый цвет, будет отражено то количество спортивных залов которое необходимо для проведения занятий у всех классов школы при такой нагрузке. После нажатия на кнопку **Распределить нагрузку**, окно **Распределение нагрузки классов в специальных помещениях** может принять вид, изображенный на рисунке 28.

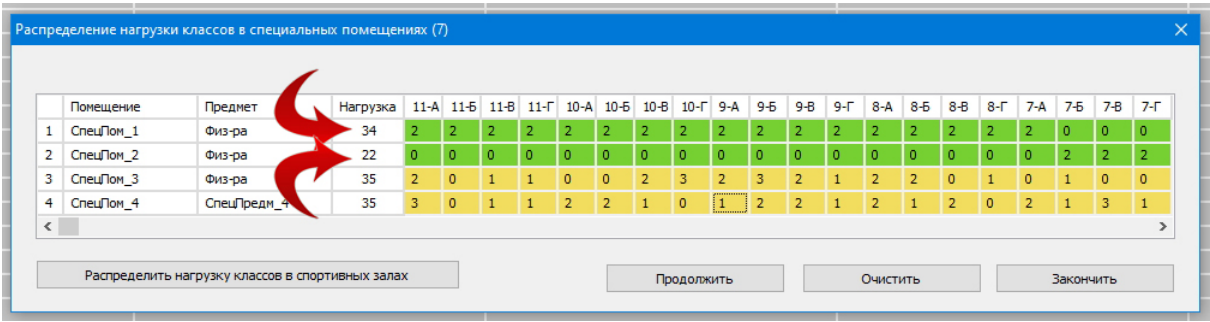


Рис. 28: Окошко Нагрузка классов по физкультуре

Разумеется это окно можно было бы привести к такому виду полностью ручным способом, но поскольку для такого распределения и подсчета необходимого количества спортзалов большого ума не требуется, разработчики программы решили данный процесс автоматизировать в надежде на то, что пользователи будут им за это благодарны.

Но пользователи вместо благодарности, внимательно приглядевшись к рисунку 28, скорее всего сообразят: что-то здесь не то.

Попробуем разобраться: что именно — не то?

Во-первых. Почему нагрузка по физкультуре у всех классов равна 2-м часам в неделю, а не 3-м? Ведь все прекрасно помнят, что еще десяток с лишним лет назад, два очень больших и очень тупых начальника торжественно декларировали: — «Нам тут доложили, что девяносто процентов выпускников средних школ в России имеют хронические заболевания. Дабы искоренить данный факт повелеваем: во всех школах России быть недельной нагрузке по физкультуре трем часам».

Вопрос о том почему эти двое тупоголовых (тупой и еще тупее) не повелели провести реконструкцию всех школ, так, что бы количества спортивных залов в школах хватило для проведения физкультуры по три часа в неделю, мы здесь не обсуждаем. Должно быть по их мнению бомбы и пушки важнее детского здоровья.

Во-вторых. Из рисунка 28 видно, что загрузка первого спортзала (СпецПом_1) составляет всего 34 часа вместо обещанных 35-ти, а второго спортзала (СпецПом_2), всего навсего 22 часа вместо все тех же 35-ти. Где правда на земле и где обещанная сто процентная загрузка спецпомещений?

Со вторым пунктом претензий все более менее понятно — ума не хватило автоматизаторам. В результате, придется таблицу нагрузки классов дорабатывать в ручную. Что здесь можно сделать?

Для начала классам начиная от 11-А до 8-Г в строчке таблицы относящейся ко второму спортзалу вместо нуля поставим единичку. Кроме того для 7-Б в первой строчке тоже вместо нуля поставим единичку. Таким образом в этих классах нагрузка будет составлять три часа в неделю. Получившаяся при таком редактировании таблица изображена на рисунке 29.

Помещение	Предмет	Нагрузка	9-А	9-Б	9-В	9-Г	8-А	8-Б	8-В	8-Г	7-А	7-Б	7-В	7-Г	6-А	6-Б	6-В	6-Г	5-А	5-Б	5-В	5-Г
1 СпецПом_1	Физ-ра	35	2	2	2	2	2	2	2	2	2	1	0	0	0	0	0	0	0	0	0	0
2 СпецПом_2	Физ-ра	35	1	1	1	1	1	1	1	1	0	2	2	2	2	2	2	2	2	2	2	2
3 СпецПом_3	СпецПредм_3	35	1	1	3	1	0	3	2	2	0	1	0	1	0	1	0	1	2	2	0	0
4 СпецПом_4	СпецПредм_4	35	1	0	0	0	0	1	2	3	0	3	1	2	1	1	0	3	0	0	1	3

Рис. 29: Окошко Нагрузка классов по физкультуре

А как же быть с остальными классами у которых нагрузка осталась в два часа? Остальным придется — по одежке протягивать ножки и передавать тупому и который еще тупее, привет с пожеланием изучать арифметику.

Напоследок отметим, что количество вариантов по редактированию таблицы нагрузки классов — огромно, а на вкус и цвет товарищей нет. У пользователя появляется возможность в полной мере проявить совою необузданную фантазию.

12.3 Окошко Расчет расписания для специальных помещений

Завершающим окошком, при решении задачи построения расписания для специальных помещений, является окошко, изображенное на рисунке 30.

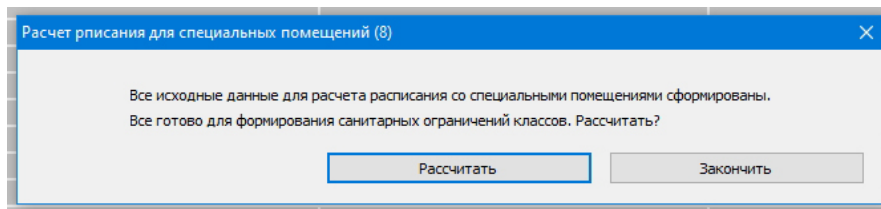


Рис. 30: Окошко Расчет расписания для специальных помещений

После нажатия на кнопку **Рассчитать** она превратиться в кнопку **Продолжить**, а вместо надписи, информирующей о том, что все готово для расчета, появиться улыбающаяся рожица. Хотя, чему тут радоваться — решенная задача весьма банальна.

Окошко с радующейся рожицей изображено на рисунке 31.

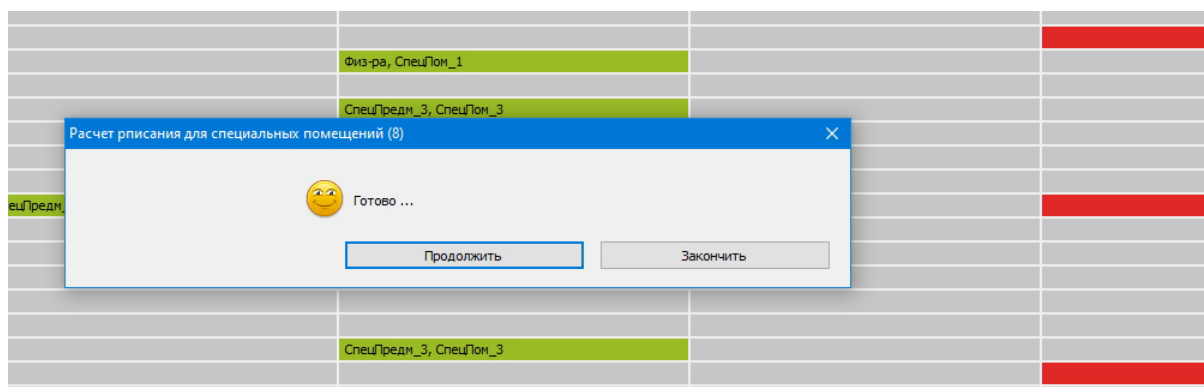


Рис. 31: Окошко после окончания расчета

Заметим, что в сетке расписания появились ячейки окрашенные салатовым и красным цветом. В салатовых ячейках через запятую указаны название предмета и название специального помещения. Красные ячейки, как уже упоминалось ранее, означают запрет проведения занятий в данный день и час.

Как обычно, пользователь может либо продолжить ввод исходных данных для решения следующей задачи, нажав на кнопку **Продолжить**, либо прервать данное занятие до следующего раза, нажав на кнопку **Закончить**.

После завершения расчета расписания для специальных помещений и расстановки всех санитарных ограничений пользователю доступна возможность перехода во вкладку **Расписание преподавателей** и добавление преподавателей в это расписание. Но на вопрос: как это сделать, дабы не прерывать логику обсуждения «Древнеегипетского» алгоритма, мы сейчас отвечать не станем поскольку эта возможность у пользователя сохраняется и дальше, вплоть до завершения расчета финального расписания. Мы дадим соответствующие пояснения позже.

И последнее на что хотелось бы обратить внимание, заканчивая данный раздел.

Для сетки расписания 6×6 при расстановке санитарных ограничений (запретов), почти всегда удастся неукоснительное соблюдение правила о запрете проведения уроков в последний урок последних дней недели. Это обусловлено тем, что таких запретов приходится расставлять сравнительно мало.

При сетке же расписания 5×7 количество запретов существенно больше, и не всегда идеальная расстановка существует, даже при расстановке запретов в последний урок не обязательно последних дней недели.

13 Третий этап алгоритма

На третьем этапе алгоритма строится расписание для совместителей. Основная особенность этой задачи заключается в том, что совместителям позволено в ультимативной форме заявлять дни по которым они будут работать. Кроме того, некоторому количеству совместителей позволено отказаться от работы в первый урок всех рабочих дней когда они преподают. Это предусмотрено на тот случай, если совместителями являются женщины и они не могут рано прибывать в школу, поскольку заняты доставкой малолетних детей в детские сады или ясли.

Построенное на данном этапе расписание представляет из себя четверки — класс, преподаватель, время проведения занятия, предмет изучения. Позже для каждого занятия (класс, преподаватель, время, предмет) подбирается помещение для проведения этого занятия.

Теперь переходим к описанию вводимых исходных данных для решения этой задачи.

13.1 Окно Формирование списка совместителей и их нагрузок

Раздел программы — Расписание для совместителей, как уже наверное догадался читатель, открывается окном, изображенном на рисунке 32.

Формирование списка совместителей и их нагрузок (9)

Количество совместителей (от 1 до 9):

	ФИО совместителя	Предмет	Нагрузка всего	Первый день (примерно)	Второй день (примерно)
1	ФИО_совм. 1	СВ_Предмет 1	5	3	2
2	ФИО_совм. 2	СВ_Предмет 2	3	3	0
3	ФИО_совм. 3	СВ_Предмет 3	6	3	3
4	ФИО_совм. 4	СВ_Предмет 4	4	3	1
5	ФИО_совм. 5	СВ_Предмет 5	5	3	2
6	ФИО_совм. 6	СВ_Предмет 6	3	3	0

Продолжить Очистить Закончить

Рис. 32: Окно Формирование списка совместителей и их нагрузок

Как видно из этого рисунка количество совместителей можно выбрать из интервала от 1 до 9. Следует отметить, что максимальное количество совместителей, которое может ввести пользователь, зависит от количества классов для которых составляется расписание занятий. Так, число 9 появилось в данном окне из расчета, что количество классов в школе равно 28-ми. При количестве классов в школе равным 12-ти, максимальное количество совместителей уменьшится до числа 5, а в случае пяти классов до совсем небольшого числа равного двум.

Особенностью расчета расписания для совместителей является то, что *они могут вести уроки только в тех классах для которых не предусмотрено разделение на группы*. Кроме того, следует обратить особое внимание на колонку с названиями предметов.

Пристального внимания эта колонка заслуживает потому, что названия предметов в нее вводятся пользователем не из заранее заготовленного списка, а в свободной (произвольной) форме. При этом, такие названия будут фигурировать в расписании наряду (наравне) с предметами для классов без разделения на группы, список которых будет сформирован пользователем при решении задачи — **Финальное расписание**.

В свою очередь, при решении задачи **Финальное расписание** пользователю нужно будет помнить (программа этого не помнит), что часть нагрузки из учебного плана уже «роздана» совместителям, что конечно же не добавляет удобства в процесс составления школьного расписания. Но как говорил товарищ Сталин: — «Других писателей для Вас у меня нет».

Как ранее было сказано, совместители отличаются от штатных преподавателей, в рамках терминологии описываемой программы тем, что им позволено в ультимативной форме требовать для себя назначение занятий в указанные ими дни. А это означает, что с увеличением количества совместителей стремительно увеличивается вероятность того,

что искомого расписания занятий для всей школы *не существует*. Таким образом, авторы программы настоятельно рекомендуют пользователям не злоупотреблять этим видом расчета для «своих в доску» штатных преподавателей.

Исключение может быть сделано только, например, для директора школы которому не хватает директорской зарплаты и он вынужден вести занятия в классах, но по некоторым дням он в школе гарантированно отсутствует, или для завуча, который то же, в некоторые дни гарантированно отсутствует в школе.

Окончательная рекомендация здесь будет такая — *если есть возможность избежать составления расписания для совместителей, то обязательно это сделайте*.

Наконец отметим, что для тех кто не любит что-то исправлять, а предпочитает в начале работы иметь «чистый лист», существует кнопка **Очистить**. Изображение окна после нажатия этой кнопки мы здесь не приводим в виду его банальности.

И последнее, ФИО совместителя не может превышать 23-х символов, название предмета 20-ти, максимальная нагрузка совместителя во все рабочие дни, которых максимум 2, не может превышать 6-ти часов, максимальная предварительно установленная нагрузка в отдельный день 3-х часов.

В процессе расчета расписания совместителей предварительно установленная дневная нагрузка и количество рабочих дней может измениться (и как правило изменяется, в большую или меньшую сторону для дневной нагрузки, и только в меньшую сторону для количества рабочих дней).

13.2 Окно Формирование таблицы рабочих дней совместителей

Следующим окном, в случае составления расписания для совместителей, будет окно, изображенное на рисунке 33.

	ФИО	Предмет	Дней	Понедельник	Вторник	Среда	Четверг	Пятница	Суббота
1	ФИО_совм. 1	СВ_Предмет 1	2	Розовый	Синий	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый
2	ФИО_совм. 2	СВ_Предмет 2	2	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый
3	ФИО_совм. 3	СВ_Предмет 3	2	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый
4	ФИО_совм. 4	СВ_Предмет 4	2	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый
5	ФИО_совм. 5	СВ_Предмет 5	1	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый
6	ФИО_совм. 6	СВ_Предмет 6	2	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый	Светло-зеленый

Рис. 33: Окно Формирование таблицы рабочих дней совместителей

Как видно из рисунка 33 в таблице рабочих дней присутствуют три цвета — розовый, светло-зеленый и синий. Ячейка окрашенная в розовый цвет означает то, что в этот день у этого совместителя — день не рабочий. Ячейка окрашенная в светло-зеленый цвет означает то, что в этот день у этого совместителя — день рабочий. Синий квадратик в правой части ячейки означает то, что эта ячейка является текущей (имеет фокус ввода).

Столбец с заголовком — **Дней**, автоматически формируется программой исходя из данных, внесенных пользователем в предыдущем окне (см. рис. 32).

Редактирование таблицы рабочих дней совместителей пожалуй следует (можно) начать с нажатия кнопки **Очистить**. Для перекраски розовой ячейки в светло-зеленую нужно по этой ячейке произвести двойной клик мышкой или после выделения нужной ячейки

с помощью нажатия соответствующих клавиш со стрелками (переместив синий квадратик в нужную ячейку) нажать клавишу **Enter**.

На экране ПК появится окошко с вопросом, изображенное на рисунке 34.

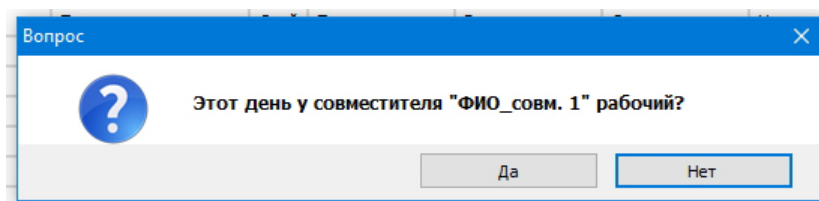


Рис. 34: Окошко с вопросом

После нажатия на кнопку **Да** окошко исчезнет, а ячейка окажется перекрашенной. Должно быть ясно, что для совершения обратного действия, т.е. перекраски ячейки из светло-зеленого цвета в розовый нужно нажать на кнопку **Нет**. При этом, если у некоторого совместителя количество окрашенных в светло-зеленый цвет ячеек может оказаться больше чем показано цифрой, имеющейся в столбце с заголовком — **Дней**, то программа не позволит пользователю осуществить такое действие.

Вот в общем-то и все, что касается этого окна. Можно переходить к описанию следующего по очереди.

13.3 Окно Формирование запретов на первый урок для совместителей

Редактирование окна, изображенного на рисунке 35, практически ничем не отличается от редактирования предыдущего (см. рис. 33).

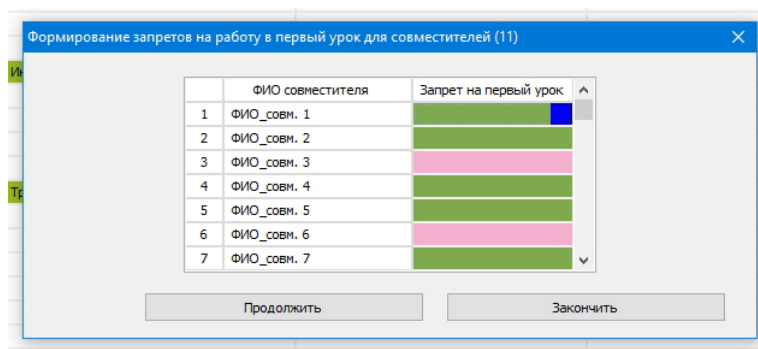


Рис. 35: Окно Формирование запретов на первый урок для совместителей

Смысл цветов тот же. Действия для изменения цветов те же.

Единственно о чем стоит упомянуть это то, что количество совместителей получивших возможность не приходить в школу к первому уроку не может превышать одного или двух в зависимости от общего количества совместителей. И еще раз напомним, что любое увеличение ограничений, в том числе и это, увеличивает вероятность столкновения с ситуацией, когда решения задачи составления расписания занятий *не существует*. Так-что не стоит «своим в доску совместителям» без крайней необходимости предоставлять возможность, устанавливаемую с помощью данного окна.

13.4 Окно Формирование таблицы распределения педагогической нагрузки совместителей

Следующее окно, которое должно появиться на экране ПК пользователя, и изображено на рисунке 36, имеет довольно страшненький, если не сказать устрашающий вид.

	Класс	[3] ФИО_совм. 1, СВ_Предмет 1 Введенная нагрузка = 3	[4] ФИО_совм. 2, СВ_Предмет 2 Введенная нагрузка = 4	[6] ФИО_совм. 3, СВ_Предмет 3 Введенная нагрузка = 6
1	11-А			
2	11-Б			1
3	11-В		1	
4	11-Г		1	1
5	10-А			
6	10-Б			1
7	10-В			
8	10-Г		1	
9	9-А			
10	9-Б	1		
11	9-В			2
12	9-Г			
13	8-А			
14	8-Б			
15	8-В			

Рис. 36: Окно Формирование таблицы распределения педагогической нагрузки

Но на самом деле не все так страшно. Начнем понемногу разбираться.

Самая первая (левая) колонка представляет из себя номера строк таблицы. Далее в следующей колонке идут названия классов, о чем свидетельствует заголовок этой колонки — **Класс**.

Далее идут колонки количество которых совпадает с количеством совместителей. То есть, эти колонки относятся к совместителям. Каждая из таких колонок имеет два заголовка. Один заголовок располагается в первой строке таблицы, другой во второй.

Верхний заголовок состоит из трех частей, при этом вторая и третья часть отделены друг от друга запятыми. Перечислим эти части:

1. Число, заключенное в квадратные скобки;
2. ФИО совместителя;
3. Название предмета которому будет учить преподаватель (совместитель).

Число заключенное в квадратные скобки, не что иное как — **Нагрузка всего**, взятая из окна **Формирование списка совместителей и их нагрузок** (см. рис. 32).

Что такое **ФИО совместителя** и **Название предмета** пояснять не станем.

Прежде чем объяснить что такое **Введенная нагрузка**, представляющая из себя нижний (второй) заголовок колонки, разберемся с цветами в которые окрашена таблица.

В этой таблице имеются четыре цвета — синий, светло-синий, розовый и светло-зеленый. Синим цветом окрашена текущая ячейка (имеющая фокус ввода), которую пользователь может редактировать. Светло-синим цветом выделена текущая строка таблицы и текущая колонка таблицы, сделано это для удобства ориентации пользователя в таблице. Розовым цветом окрашены ячейки в которых отсутствует величина (значение)

нагрузки совместителя в некотором классе, а светло-зеленым цветом — ячейки в которых это значение имеется (ненулевое).

Общая рекомендация по редактированию данной таблицы следующая. Для начала пользователь может покликать по таблице мышкой или постучать клавишами со стрелками с целью понять — как должна выглядеть таблица после окончания ее редактирования. После чего можно нажать на кнопку **Очистить**. Мало того, что в таблице исчезнут ячейки окрашенные светло-зеленым цветом, но и еще изменятся все заголовки таблицы, расположенные в ее второй строке. Для наглядности продемонстрируем состояние этих заголовков рисунком 37.

		[3] ФИО_совм. 1, СВ_Предмет 1	[4] ФИО_совм. 2, СВ_Предмет 2	[6] ФИО_совм. 3, СВ_Предмет 3
	Класс	Введенная нагрузка = 0	Введенная нагрузка = 0	Введенная нагрузка = 0
1	11-А			
2	11-Б			
3	11-В			
4	11-Г			

Рис. 37: Часть окна **Формирование таблицы распределения нагрузки**

Из рисунка видно, что все значения заголовка **Введенная нагрузка** оказались нулевыми. Проницательный читатель уже наверное смекнул, что далее придется редактировать таблицу. Делать это можно стандартным способом, принятым в программе — двойной клик мышкой по нужной ячейке или после наведения с помощью клавиш со стрелками синего указателя на нужную ячейку — нажатие клавиши **Enter**.

В результате указанных действий на экране ПК должно появиться небольшое окошко **Ввод нагрузки совместителя**, изображенное на рисунке 38.

Ввод нагрузки совместителя

Нагрузка совместителя в классе (от 0 до 2):

Рис. 38: Окошко **Ввод нагрузки совместителя**

Из этого рисунка видно, что ячейка готова принять число (значение) от нуля до двух. Значение ноль предусмотрено на тот случай если пользователю будет необходимо изменить ранее введенное ненулевое значение на нулевое, т.е. у некоторого совместителя отобрать нагрузку для некоторого класса (в некотором классе).

После ввода в ячейку нагрузки, величина **Введенная нагрузка** (т.е. заголовок колонки) изменится на введенную нагрузку из окошка **Ввод нагрузки совместителя**.

Задача пользователя будет заключается в добавлении (редактировании) нагрузок до тех пор пока каждое число в каждом заголовке колонки — **Введенная нагрузка**, не окажется равным каждому числу, заключенному в квадратные скобки, в каждом заголовке первой строки таблицы.

Достигнув желаемого можно будет нажать на кнопку **Продолжить** для перехода к следующему окну.

13.5 Окошко Построение расписания для совместителей

Завершающим окошком при решении задачи **Расписание для совместителей** будет стандартное окошко, сначала с надписью о готовности всех исходных данных для решения задачи, а потом с улыбающейся рожицей. Поскольку изображения таких окошек мы уже дважды приводили, повторяться не будем, а перейдем к окнам появляющимся при решении следующей задачи — **Расписание потоков**. Отметим только лишь, что в сетке расписания появятся ячейки окрашенные сиреневым цветом, изображенные на рисунке 39, в которых будут отражены — фамилии совместителей и их предметы.

	Урок 4	ФИО_совн. 2, СВ_Предмет 2			
	Урок 5		Физ-ра, СпецПом_2		Физ-ра, СпецПом_2
	Урок 6		СпецПредм_3, СпецПом_3		
	Урок 7				СпецПредм_3, СпецПом_3
5-8	Урок 1				СпецПредм_4, СпецПом_4
	Урок 2		СпецПредм_3, СпецПом_3		
	Урок 3				Физ-ра, СпецПом_2
	Урок 4				
	Урок 5	ФИО_совн. 5, СВ_Предмет 5		ФИО_совн. 1, СВ_Предмет 1	СпецПредм_3, СпецПом_3
	Урок 6				
	Урок 7	Физ-ра, СпецПом_2			
5-7	Урок 1	ФИО_совн. 5, СВ_Предмет 5	ФИО_совн. 4, СВ_Предмет 4	ФИО_совн. 4, СВ_Предмет 4	
	Урок 2				
	Урок 3	ФИО_совн. 2, СВ_Предмет 2			
	Урок 4	Физ-ра, СпецПом_2			Физ-ра, СпецПом_2

Рис. 39: Появление ячеек окрашенных сиреневым цветом

Но прежде чем все же перейти к описанию следующего этапа алгоритма (программы) сделаем еще два небольших пояснения.

Во-первых. Скептически настроенный завуч или другой человек составляющий расписание школы может недовольно задать следующий вопрос: как же так, программа позволяет (ограничивает) максимальную нагрузку совместителя в одном классе двумя часами и двумя днями работы, итого шесть часов в неделю? А если для полного счастья в жизни совместителю требуется семь или девять часов учебной нагрузки в неделю, что тогда.

Тогда тупую программу можно легко обмануть, увеличив учебную нагрузку по крайней мере вдвое. Делается это следующим образом.

Пусть, например, совместитель (завуч, директор) — Иванов, не может находиться в школе только в среду. Формируем для него учебную нагрузку в понедельник и вторник. Как не трудно догадаться, эта нагрузка не может превысить шести часов. Далее вводим фиктивного совместителя под фамилией — «Иванов 2» и формируем для него нагрузку в четверг и пятницу. Поскольку учебные часы настоящего Иванова и фиктивного не могут пересекаться, они же работают в разные дни, настоящий Иванов совершенно свободно может преподавать вместо фиктивного.

Второе пояснение (замечание). Еще один вопрос который может возникнуть у скептически настроенного читателя: а почему-это, программа ограничивает нагрузку совместителя в одном классе?

Предположим, что совместитель выбрал для работы только один рабочий день. И далее формирует нагрузку для некоторого класса — два часа, что естественно, меньше его максимальной нагрузки — шесть часов. Но алгоритм неукоснительно соблюдает правило для всех классов — в один день только один урок по некоторому предмету, а нагрузка в два часа, при одном рабочем дне, этому правилу противоречит. Стало быть, при таких обстоятельствах, решение задачи не существует.

Аналогичные рассуждения можно провести для нагрузки в три часа при двух рабочих днях, которые мы здесь повторять не станем.

Приведенный случай отсутствия существования решения задачи весьма банален. Хотелось бы конечно знать простой критерий существования решения на все практические ситуации. Но как отмечалось ранее, в этом вопросе помощником может послужить только вычислительный эксперимент.

Такой эксперимент был проведен³¹ поскольку суть вычислительного эксперимента весьма проста — прогон бешеного (пятьдесят, лучше сто миллионов) количества тестовых задач со случайно сформированными данными и в случае отсутствия решения у задачи анализ ее исходных данных, в попытке вывести закономерность.

Статистика оказалась следующей.

Из каждого миллиона тестовых задач, по расчету расписания для совместителей, сотня — другая не решалась. При этом не решались задачи исключительно для сетки расписания 6×6 , с предельно большим количеством классов и предельно большим количеством специальных помещений. Иначе говоря, с предельно большим количеством уже занятых ячеек сетки расписания.

Но что характерно — при возврате к предыдущему шагу алгоритма для перерасчета с теми же исходными данными расписания для специальных помещений (как это сделать будет подробно объяснено далее) все нерешенные задачи были решены. Из данного факта мы приходим к выводу о том, что существование решения в задаче о совместителях зависит не только от количества уже занятых ячеек сетки расписания, но и от их конкретного размещения в этой сетке. Что вполне логично, поскольку каждый совместитель задает для себя вполне конкретные рабочие дни.

Теперь что касается вывода закономерности (критерия отсутствия решения).

Для анализа исходных данных, задач которые не удалось решить, использовались два хорошо зарекомендовавших себя на практике метода:

1. Метод грубой силы;
2. Метод тупого вглядывания.

С помощью метода грубой силы удалось отличать не достаточно «окученные» задачи от задач не имеющих решения.

Метод же тупого вглядывания к сожалению успеха не имел. Остается только надеяться, что из примерно десяти тысяч весьма экзотических задач, программа не покажет пользователю окошко, изображенное на рисунке 40.

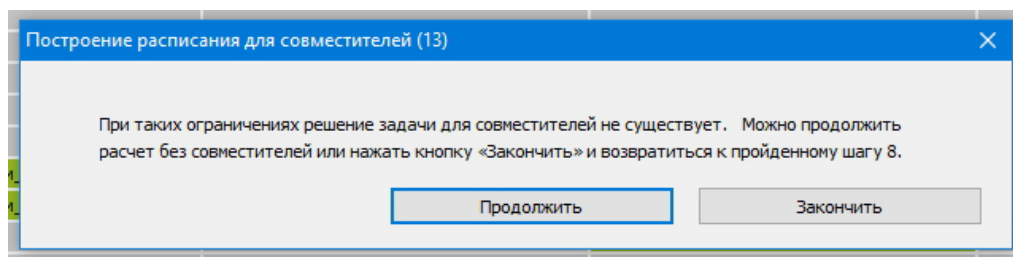


Рис. 40: Окошко с сообщением об отсутствии решения задачи

Вот теперь, пожалуй, можно остановиться с дополнительными пояснениями к задаче расчета расписания для совместителей и перейти к описанию следующего этапа алгоритма.

³¹Естественно, с помощью консольного варианта программы.

14 Четвертый этап алгоритма

Четвертый этап алгоритма состоит из решения задачи о проведении занятий по выбору учащихся (элективных курсов). Сутью этой задачи является то, что некоторое количество классов, в некоторый учебный час, объединяются в потоки, что бы затем в этот час разделиться на небольшие группы и разойтись по своим элективным курсам. Построение расписания будет заключаться в том, что каждому потоку будет назначено время в которое будут проводиться элективные курсы, но преподаватели и помещения будут назначаться уже после того как все расписание будет окончательно построено. То есть, на этом этапе преподаватели для проведения элективных курсов и помещения не назначаются. Таким образом, построенное расписание состоит из множества двоек — поток (с включенными в него классами) и момент времени (день недели, номер урока). Отметим, что в программе «АФМ: Аннушка» в поток может входить от одного до двух классов.

При построении такого расписания соблюдается правило — для любого потока в один учебный день может быть назначено не более одного учебного часа для проведения элективного курса. Кроме того, соблюдается другое правило — в любой момент времени (учебный час) не могут быть запланированы элективные курсы более чем для одного потока. В том случае, если суммарная нагрузка всех потоков превышает продолжительность учебной недели (в учебных часах), т.е. решение задачи заведомо не существует, то элективные курсы не могут быть запланированы в любой момент времени (учебный час) более чем для двух потоков. Это правило (ограничение) представляется вполне разумным, поскольку при проведении элективных курсов резко возрастает потребность в помещениях для их проведения. Оно введено именно с той целью, что бы не возникло ситуации когда нескольким потокам в одно и тоже время требуется большое количество свободных помещений.

Как уже было отмечено, помещения для проведения элективных курсов, на этом шаге, так же как и преподаватели не подбираются, они будут подбираться вместе с преподавателями после построения всего расписания.

Здесь интересно отметить, что в Российской Федерации в период с 2000 года по 2010 год намечалась реформа среднего образования, которая заключалась в так называемой профилизации обучения (см., например, [3, 4, 12, 14]). При этом основные надежды возлагались на свободу выбора учащимися элективных курсов. Сейчас, с высоты 2023 года об этой реформе можно сказать следующее. «Всё педагогическое сообщество, с удивительной солидарностью, набрало полный рот соплей и харкнуло в рожу этим разработчикам». Не последнюю роль (а может быть и первостепенную) в этом процессе сыграли трудности составления школьного расписания с интегрированными в него часами для проведения элективных курсов. Хотя, конечно, не стоит сбрасывать со счетов банальную нехватку школьных зданий, точнее, свободных помещений для проведения элективных курсов, другими словами — удручающую бедность в нашей стране для подавляющего большинства населения.

14.1 Окошко Количество потоков и старших параллелей объединяемых в потоки

Прежде чем нажать на кнопку Продолжить, появившегося окошка Количество потоков и старших параллелей объединяемых в потоки, изображенного на рисунке 41, пользователю программы стоило бы хорошенечко подумать.

Для того, что бы выбор вводимых данных был пользователем вполне осмысленным

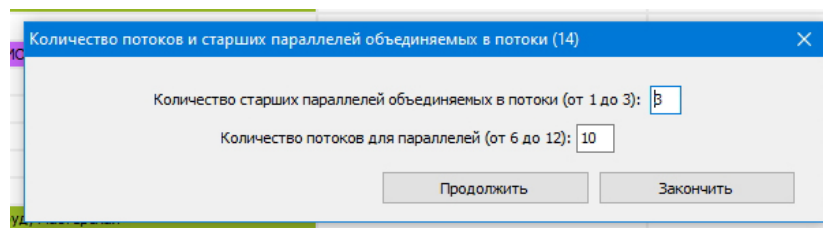


Рис. 41: Окошко Количество потоков и старших параллелей объединяемых в потоки

следует еще раз пояснить — что именно, в терминах программы, подразумевается под распространенным и имеющим большое количество значений, словом — *поток*.

В программа «АФМ: Аннушка» под термином **поток** подразумевается множество учащихся состоящее из одного или двух классов, как правило (желательно) из одной параллели. Целью формирования потока (множества учащихся) является разделение потока на отдельные группы учащихся по интересам и проведение для этих групп занятий по элективным курсам. Для осуществления этой цели для каждого потока резервируется время в сетке расписания занятий всей школы.

По умолчанию, при формировании потоков программа руководствуется следующими правилами:

- Потоки формируются из учащихся старших параллелей;
- Количество старших параллелей из которых формируются потоки определяется пользователем;
- Потоки формируются из всех классов без исключения, входящих в старшие параллели, количество которых задано пользователем;
- Любой класс может входить в потоки только один раз.

Таким образом, если пользователь в поле ввода — **Количество старших параллелей** введет цифру 1, то потоки будут формироваться только из всех одиннадцатых (самых старших) классов. Если будет введена цифра 2, то из всех одиннадцатых и десятых классов, если цифра 3, то из всех одиннадцатых, десятых и девярых классов. При вводе конкретного значения в это поле, пользователь должен руководствоваться указанным интервалом — от 1 до 3.

Следующее поле — **Количество потоков**, ограничено интервалом — от **Минимума** до **Максимума**. Другими словами программа не позволит пользователю ввести число большее максимума и меньше минимума.

Возникает вопрос — откуда берутся эти числа?

Максимум равен количеству классов, входящих в старшие параллели. В данном примере это 12 классов. Если пользователь задаст именно это число потоков, то в каждый поток будет входить только один класс.

Если пользователь введет число потоков соответствующее минимуму, то в каждый поток будет входить по два класса, при четном количестве классов и по два класса кроме последнего потока, при нечетном количестве классов. Понятно, что последний поток в этом случае, будет состоять только из одного класса.

После того, как пользователь определился с количеством параллелей и количеством потоков он может нажимать кнопку **Продолжить**.

14.2 Окно Формирование потоков

Появившееся окно — Формирование потоков, изображено на рисунке 42.

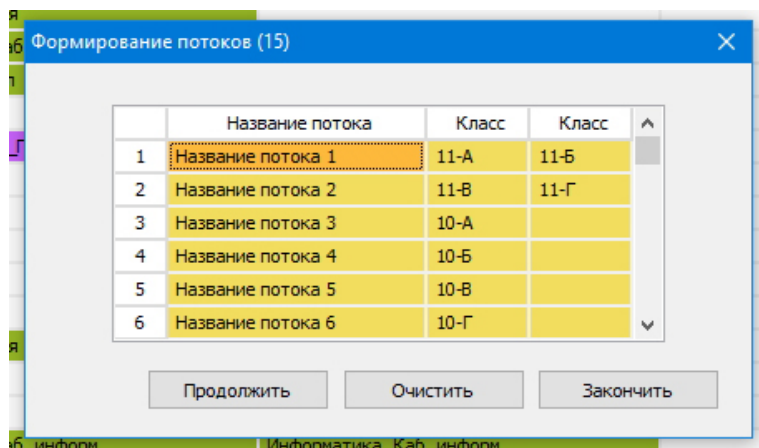


Рис. 42: Окно Формирование потоков

По мнению разработчиков программы, правила формирования потоков, принятые в этой программе по умолчанию достаточно разумны. Однако, не предоставить пользователю возможности руководствоваться своими правилами, было бы весьма опрометчиво.

На этот случай в окне **Формирование потоков** имеется кнопка **Очистить**. После ее нажатия окно будет иметь вид, изображенный на рисунке 43.

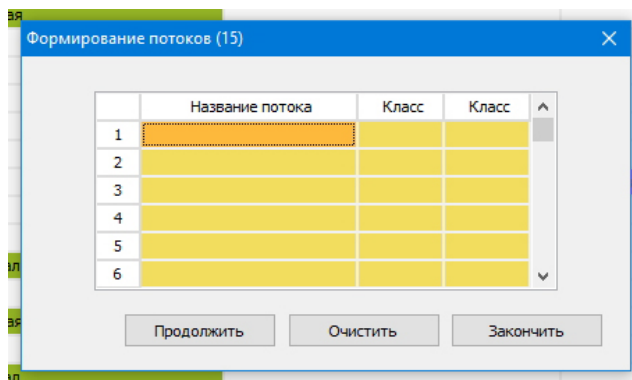


Рис. 43: Окно Формирование потоков после его очистки

Проницательный читатель уже наверное сообразил, что пользователю можно редактировать не только названия потоков, но и классы, входящие в эти потоки. Строго говоря, такое редактирование можно было бы проводить и без очищения окна (таблицы формирования потоков), но как показалось разработчикам программы кнопка **Очистить** может придать процессу редактирования особый шарм.

На редактировании названий потоков особо останавливаться не будем — все как обычно, подводим курсор к нужной ячейке и давим на клавиши. Поясним подробно процесс редактирования классов, входящих в потоки.

Предположим, что мы находимся не в самом начале процесса редактирования, а где-то в середине этого процесса. Окно **Формирование потоков** будет выглядеть, например, так, как изображено на рисунке 44.

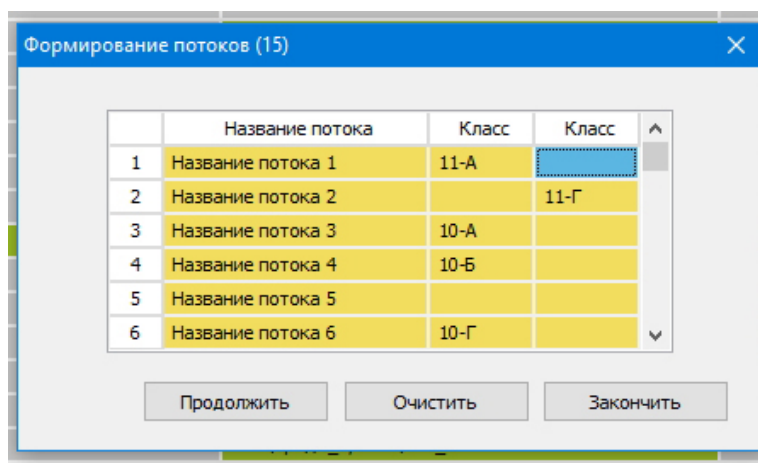


Рис. 44: Окно Формирование потоков при редактировании

Из этого рисунка видно, что часть классов уже помещено в таблицу формирования потоков, а текущей ячейкой, которая окрашена в светло-голубой цвет, является ячейка в которую пользователь только собрался поместить очередной класс. Для этого он должен либо сделать двойной клик мышкой по этой ячейке либо нажать клавишу **Enter**. После одного из таких действий на экране ПК появиться дополнительное окошко **Выбор класса**, изображенное на рисунке 45.

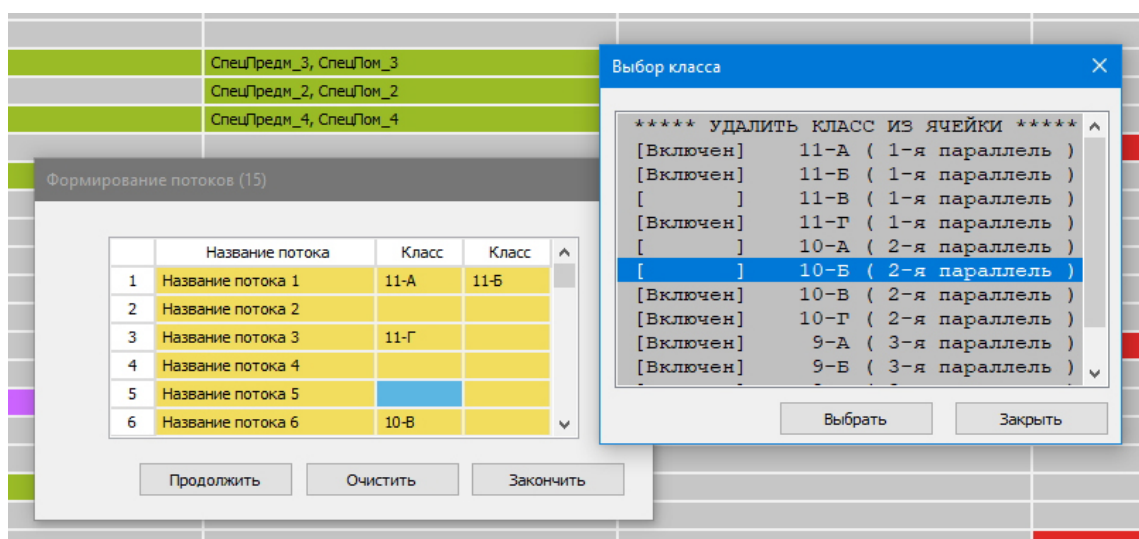


Рис. 45: Дополнительное окошко Выбор класса

В этом окошке имеются три колонки. Центральная колонка состоит из старших классов, которые можно включать в потоки. Самая правая колонка уточняет — к какой параллели относится тот или иной класс. А первая (левая) колонка состоит из отметок, показывающих, включен ли уже класс в потоки или нет. Как не трудно догадаться, отметка **[Включен]** свидетельствует о том, что этот класс уже включен в какой-то поток и этот класс повторно включить, в тот же или какой-то другой поток, не получится. Отметка же **[]**, состоящая из квадратных скобок внутри которых ничего нет, свидетельствует об обратном.

Для помещения класса в выбранную ячейку таблицы формирования потоков нужно в окошке **Выбор класса** подвести курсор (голубую полосу) к нужному классу и совершить стандартное для программы действие — двойной клик мышкой, нажатие клавиши **Enter** или нажатие кнопки **Выбрать**. После чего окошко исчезнет, а в выделенной ячейке **Таблицы формирования потоков** появится название выбранного класса. Для помещения следующего класса в следующую ячейку действия должны повториться.

Отметим, что если пользователю захочется очистить какую-либо ячейку от названия класса, ему нужно будет навести курсор на эту ячейку и в окошке **Выбор класса** выбрать первую (самую верхнюю) строку списка — ******* УДАЛИТЬ КЛАСС ИЗ ЯЧЕЙКИ *******. Кроме того, уже имеющиеся в ячейках таблицы классы (уже выбранные классы), можно еще и заменять. Попробуйте это сделать без объяснения «могучей техники процесса».

После того, как пользователь закончит редактирование таблицы формирования потоков и нажмет кнопку **Продолжить** некоторые классы, присутствующие в окошке **Выбор класса** могут остаться не выбранными, а некоторые потоки остаться без выбранных для них классов. Поскольку «бестолковая программа» самостоятельно не может определить — забыл ли пользователь выбрать некоторый класс или делает это совершенно сознательно, программа попросит его уточнить данное обстоятельство с помощью вопроса (окошко вопрос), представленного на рисунке 46.

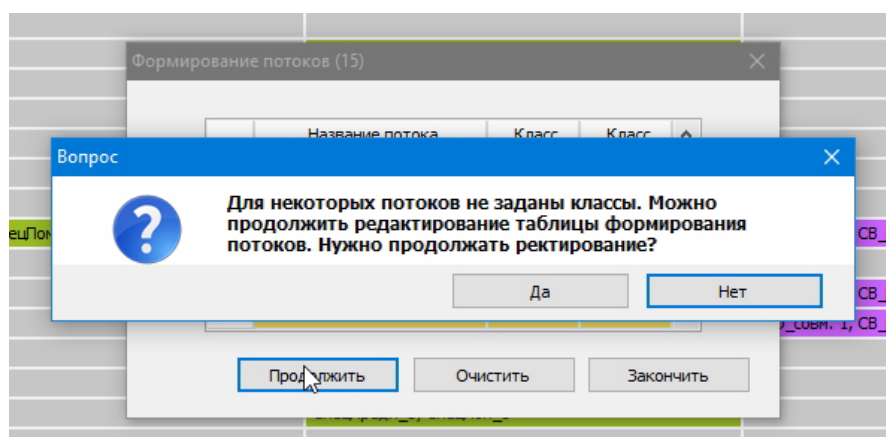


Рис. 46: Окошко Вопрос

Если пользователь нажмет кнопку **Нет**, то программа может перейти к следующему окну для формирования исходных данных. При этом, если какой-либо поток остался без выбранного класса, то количество потоков будет программой автоматически скорректировано (уменьшено). Если какой-либо класс остался не выбранным, то программа проинформирует пользователя об этом с помощью вопроса аналогичного вопросу изображенному на рисунке 46, и в случае отказа пользователя от дальнейшего редактирования таблицы потоков, в расписании для этого класса не будет зарезервировано времени на проведение элективных курсов.

14.3 Окно Формирование нагрузок потоков

После того как пользователь все же «разберется» с «бестолковой программой» на экране ПК появится очередное окно **Формирование нагрузок потоков** (см. рис. 47).

Действия пользователя при редактировании этого окна (таблицы нагрузок потоков) настолько банальны, что описывать здесь мы их не станем, отметим лишь, что цифра 0

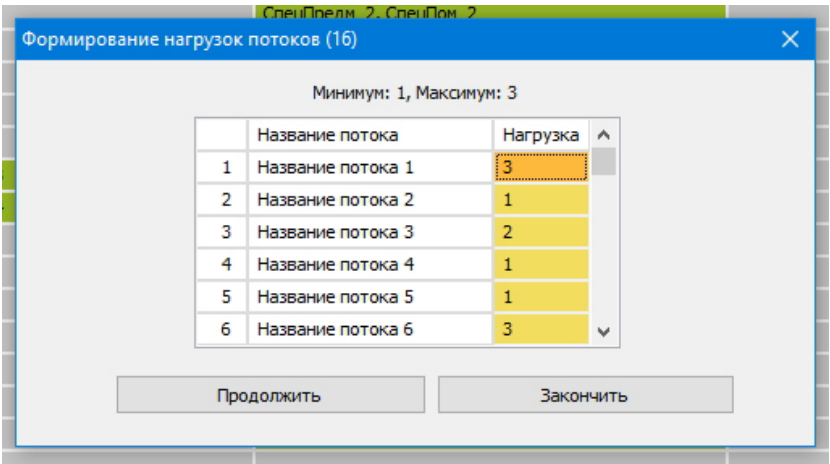


Рис. 47: Окно Формирование нагрузок потоков

и цифры больше 3-х не пройдут, поскольку цифра 0 — бессмысленна, а цифра 3 является максимальной нагрузкой для любого потока.

14.4 Окошко Построение расписания для потоков

Как всегда завершающим штрихом решения задачи, в данном случае задачи **Расписание потоков**, будет являться окошко **Построение расписания для потоков**, которое мы здесь как и в прошлый раз приводить не станем, поскольку сообщения о готовности решить задачу и улыбающиеся рожицы могут сильно утомить читателя. Однако отметим, что появившиеся ячейки в сетке расписания, относящиеся к потокам, будут окрашены цветом, напоминающим нечто среднее между серым и голубым (см. рис. 48).

	Урок 7	Название потока 2			
11-Г	Урок 1	Физ-ра, СпецПом_2			
	Урок 2	Физ-ра, СпецПом_1		Название потока 2	
	Урок 3				
	Урок 4	СпецПом_4, СпецПом_4	СпецПом_4, СпецПом_4		
	Урок 6	Физ_сопн_5, СВ_Предмет 5			
	Урок 7	Название потока 2	Физ-ра, СпецПом_1		
10-А	Урок 1			Физ-ра, СпецПом_1	
	Урок 2	Физ-ра, СпецПом_2			
	Урок 3		СпецПом_4, СпецПом_4	Физ_сопн_2, СВ_Предмет 2	
	Урок 4				
	Урок 5			Название потока 3	
	Урок 6				

Рис. 48: Окраска потоковых ячеек серо-голубым цветом

В самих же ячейках будет красоваться название потока. Этим завершается четвертый этап алгоритма.

15 Пятый этап алгоритма

На следующем этапе алгоритма (пятом) строиться расписание для уроков, как правило, по изучению иностранного языка. Особенностью этой задачи является то, что класс может быть разделен на группы (в нашем варианте на две), при этом количество предметов (иностраннных языков) может быть больше одного (в нашем варианте до трех). Преподаватели не могут в ультимативном порядке заявлять о том в какие дни они будут

работать. Однако, для преподавателей с небольшой недельной нагрузкой³² и небольшой максимальной нагрузкой в отдельном классе³³, гарантирован один или два выходных (методических дня), которые будут предоставлены им автоматически. Точно также, как и на третьем этапе алгоритма, некоторые преподаватели, обучающие иностранному языку, могут потребовать освободить их от уроков в первый час рабочего дня когда они проводят занятия.

Построенное расписание на этом этапе состоит из четверок — момент времени, изучаемый предмет, класс (пары групп), пары преподавателей. Точно так же, как и на третьем этапе алгоритма, позже, каждому занятию, а точнее, каждой группе учеников и их преподавателям подбираются помещения для проведения запланированных занятий. После окончания 5-го этапа алгоритма, в зависимости от учебного плана школы, расставляется обычно от 40% до 50% всей учебной нагрузки, предусмотренной этим планом.

Перейдем теперь к описанию исходных данных, необходимых для решения сформулированной задачи.

15.1 Окно Формирование учебного плана для предметов с разделением на группы

В отличие от предыдущей задачи — Расписание потоков для планирования и проведения элективных курсов, которая в Российской Федерации является «далекой экзотикой» и чем-то малознакомым, задача — Расписание для классов с разделением на группы, является повсеместно решаемой, и задачей весьма почтенного возраста. Как правило, в средней и старшей школе классы делятся на группы, в первую очередь, при изучении иностранного языка, что обусловлено спецификой этого предмета.

Обратим внимание на то, что не следует путать обучение по предметам с разделением на группы, например, такого предмета как «Труд», с рассматриваемой далее задачей. При обучении предмету «Труд», класс тоже может разделяться на две группы — мальчики и девочки. Мальчики идут в мастерскую и обучаются работе на токарных, фрезерных и сверлильных станках, а девочки идут в другое оборудованное помещение, обучаться работе на швейных машинках. Но расписание для такой задачи не что иное, как задача — Расписание для специальных помещений. Просто *логическое* специальное помещение эквивалентно, *двум физическим* специальным помещениям — слесарная мастерская и швейная мастерская.

Назовем основные параметры задачи, которую нам предстоит рассмотреть.

- Количество предметов, при изучении которых в школе, классы могут быть разделены на группы, лежит в интервале от одного до трех;
- Максимальная недельная нагрузка для такого предмета в отдельном классе равна трем часам;
- Минимальная недельная нагрузка по предмету для классов всей школы, принята равной количеству классов в школе для которых составляется расписание (средняя и старшая школа).

Теперь можно представить очередное окно, подлежащее осмыслению (см. рис. 49).

³²Под небольшой недельной нагрузкой подразумевается недельная нагрузка менее 19-ти часов для сетки расписания 6×6 и менее 16-ти часов для сетки расписания 5×7.

³³Под небольшой максимальной нагрузкой в отдельном классе подразумевается недельная нагрузка менее 5-ти часов для сетки расписания 6×6 и менее 4-х часов для сетки расписания 5×7.

Предмет	11-А	11-Б	11-В	11-Г	10-А	10-Б	10-В	10-Г	9-А	9-Б	9-В	9-Г	8-А	8-Б	8-В	8-Г
1 Ин.яз.1	2	2	3	2	3			1	1	3		3	1		2	2
2 Ин.яз.2	3		3	1	2	2	1		3	1	3	3	2		2	3
3 Ин.яз.3	2	3	3		2	2		3	1		1		2	3	1	1

Рис. 49: Окно Формирование учебного плана для предметов с разделением

Как обычно таблица, имеющаяся в этом окне, уже заполнена случайно сформированными данными. Пользователь, после того как определиться с количеством предметов, может либо начать исправлять эти данные, либо нажать на кнопку **Очистить** и приступить к самостоятельному заполнению таблицы, не забывая о том, что максимальная длина названия предмета составляет 20 символов, а максимальная нагрузка класса не должна превышать 3-х часов.

Отсутствие значения нагрузки в соответствующей ячейке или нагрузка равная нулю означает то, что в данном классе данный предмет не преподается (не изучается).

Контроль (проверка) правильности заполнения таблицы будет осуществляться программой после нажатия кнопки **Продолжить**. В случае выявления ошибок заполнения таблицы, программа не позволит пользователю перейти к следующему окну, возвращая его снова и снова к редактируемой таблице, до тех пор пока не сочтет ее правильно заполненной. Думается, что представленное окно, в отличие от следующего не должно доставить пользователю особых хлопот.

15.2 Окно Расчет количества преподавателей для предметов с разделением на группы

Окно которое может доставить неприятности пользователю представлено на рисунке 50.

Предмет	Минимум	Максимум	Количество преподавателей
1 Ин.яз.1	3	6	4
2 Ин.яз.2	4	8	5
3 Ин.яз.3	3	6	4

Рис. 50: Окно Расчет количества преподавателей для предметов с разделением

Спрашивается — каким образом производится расчет количества преподавателей? Очень просто. Общая учебная нагрузка школы по некоторому предмету делится на норму нагрузки отдельного преподавателя, которая составляет 18 часов в неделю. Именно эта вычисленная и округленная до целого числа цифра заноситься в крайний правый столбец таблицы.

Пользователь, вычисленное значение, может корректировать, но в пределах от некоторого, вычисленного **Минимума** до некоторого **Максимума**. Понятно, что чем больше преподавателей получают нагрузку, тем меньшая недельная нагрузка окажется у каждого из них для начисления заработной платы.

Максимум вычисляется достаточно просто — это удвоенное значение **Минимума**. А вот с **Минимумом** не так все просто, как хотелось бы. Понятно, что чем меньше преподавателей, тем больше их недельная нагрузка и тем выше заработная плата (тем хуже преподаватели учат своему предмету). Разработчики программы, пребывая в полной уверенности в том, что найдется огромное количество пользователей которые при редактировании таблицы будут сокращать количество преподавателей до тех пор пока их недельная нагрузка не достигнет значения в 39 часов, решили ограничить значение количества преподавателей значением — **Минимум**. При этом величина минимума, путем экспериментов, тупо подбиралась разработчиками с помощью большого количества вариантов исходных данных и полученных на их основе результатов.

Возникает резонный вопрос — а, что если нет такого количества преподавателей, которое определено минимумом. Ответом на него может быть пословица: — «По одежки протягивай ножки». Значит придется корректировать учебный план или составлять расписание вручную. У разработчиков не хватило мужества предоставить пользователям возможность вводить исходные данные для задачи, решение которой не существует.

Будем предполагать и надеяться, что пользователю все же удастся втиснуться в предлагаемый интервал значений, а нажатие на кнопку **Продолжить** приведет к появлению следующего окна.

15.3 Окно Список преподавателей и запретов на первый урок

С появившимся окном, изображенном на рисунке 51, все просто.

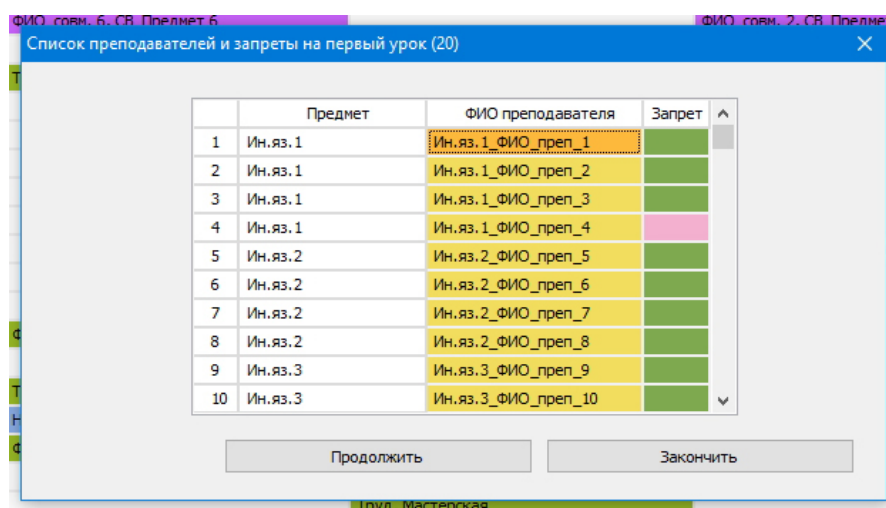


Рис. 51: Окно Список преподавателей и запретов на первый урок

Пользователю предстоит только отредактировать колонку **ФИО преподавателя** и далее в крайнем правом столбце, имеющим заголовок **Запрет**, с помощью стандартных приемчиков (двойной клик мышкой по нужной ячейке или с помощью клавиши Enter) определиться с тем, кому будет позволено не приходить в школу к первому уроку. Правда, в случае небольшого количества преподавателей, таких может оказаться не больше

одного. Окошко с вопросом — есть ли у преподавателя запрет на первый урок или нет, мы здесь не приводим по причине его банальности. Отметим, что длина ФИО преподавателя не может превышать 23-х символов. Кроме того, ФИО преподавателя не может совпадать с ФИО совместителя. Другими словами совместители не могут быть одновременно и преподавателями в классах с разделением на группы.

Напомним, что после решения задачи о спецпомещениях (второй этап расчета школьного расписания) пользователю позволено сразу же приступить к формированию списка преподавателей которые будут учить классы в этих спецпомещениях. Если пользователь воспользовался такой возможностью, то введенный им на данном шаге список преподавателей для классов с разделением на группы будет сравниваться со списком преподавателей в спецпомещениях поскольку совпадения в этих списках так же недопустимы.

Наконец, еще раз напомним «Золотое правило» разработчика школьных расписаний — *если есть возможность избежать каких-либо ограничений*, в данном случае предоставление преподавателю права не приходить к первому уроку, *непрерывно воспользуйтесь такой возможностью*.

15.4 Окно Распределение педагогической нагрузки в классах с разделением на группы

Следующее окно, изображенное на рисунке 52 представляет из себя автоматически полученное распределение педагогической нагрузки для преподавателей по предметам с разделением на две группы.

Класс	Предмет	Нагрузка	Колич. групп	ФИО преподавателя	ФИО преподавателя
1	11-А	Ин.яз.1	2	Ин.яз.1_ФИО_преп_1	Ин.яз.1_ФИО_преп_3
2	11-А	Ин.яз.2	2	Ин.яз.2_ФИО_преп_5	Ин.яз.2_ФИО_преп_8
3	11-А	Ин.яз.3	1	Ин.яз.3_ФИО_преп_11	Ин.яз.3_ФИО_преп_13
4	11-Б	Ин.яз.1	1	Ин.яз.1_ФИО_преп_3	Ин.яз.1_ФИО_преп_4
5	11-Б	Ин.яз.2	1	Ин.яз.2_ФИО_преп_6	Ин.яз.2_ФИО_преп_7
6	11-Б	Ин.яз.3	1	Ин.яз.3_ФИО_преп_10	Ин.яз.3_ФИО_преп_12
7	11-В	Ин.яз.1	1	Ин.яз.1_ФИО_преп_2	Ин.яз.1_ФИО_преп_4
8	11-Г	Ин.яз.3	3	Ин.яз.3_ФИО_преп_9	Ин.яз.3_ФИО_преп_11
9	10-А	Ин.яз.1	1	Ин.яз.1_ФИО_преп_2	Ин.яз.1_ФИО_преп_3
10	10-А	Ин.яз.2	1	Ин.яз.2_ФИО_преп_5	Ин.яз.2_ФИО_преп_8
11	10-А	Ин.яз.3	1	Ин.яз.3_ФИО_преп_11	Ин.яз.3_ФИО_преп_12
12	10-Б	Ин.яз.3	3	Ин.яз.3_ФИО_преп_9	Ин.яз.3_ФИО_преп_11
13	10-Г	Ин.яз.1	3	Ин.яз.1_ФИО_преп_1	Ин.яз.1_ФИО_преп_4
14	10-Г	Ин.яз.2	3	Ин.яз.2_ФИО_преп_5	Ин.яз.2_ФИО_преп_7
15	10-Г	Ин.яз.3	3	Ин.яз.3_ФИО_преп_12	Ин.яз.3_ФИО_преп_13
16	9-А	Ин.яз.1	3	Ин.яз.1_ФИО_преп_3	Ин.яз.1_ФИО_преп_4
17	9-А	Ин.яз.2	1	Ин.яз.2_ФИО_преп_5	Ин.яз.2_ФИО_преп_6
18	9-А	Ин.яз.3	2	Ин.яз.3_ФИО_преп_10	Ин.яз.3_ФИО_преп_12
19	9-Б	Ин.яз.1	3	Ин.яз.1_ФИО_преп_2	Ин.яз.1_ФИО_преп_3
20	9-Б	Ин.яз.2	1	Ин.яз.2_ФИО_преп_5	Ин.яз.2_ФИО_преп_7
21	9-Б	Ин.яз.3	3	Ин.яз.3_ФИО_преп_10	Ин.яз.3_ФИО_преп_11
22	9-В	Ин.яз.3	1	Ин.яз.3_ФИО_преп_11	Ин.яз.3_ФИО_преп_13
23	8-А	Ин.яз.2	3	Ин.яз.2_ФИО_преп_6	Ин.яз.2_ФИО_преп_7

Рис. 52: Окно Распределение педагогической нагрузки в классах

Пользователь программы может либо приступить к исправлению распределения, полученного автоматически, либо нажать на кнопку **Очистить** для того, что бы выполнить такое распределение полностью «вручную» — с «чистого листа». Разумеется, ничто не

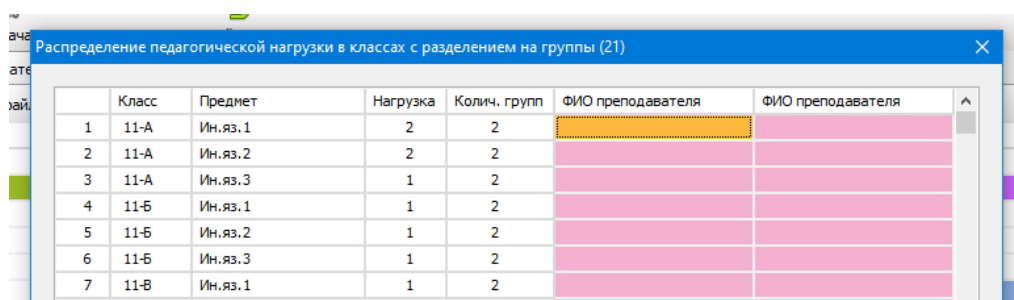
может ему помешать нажать на кнопку **Продолжить** для перехода к следующему, заключительному (при решении задачи **Расписание для классов с разделением на группы**) окошку.

Остается пояснить — как осуществляется такое редактирование?

Прежде всего нужно сказать о том, что каждая строчка таблицы соответствует нагрузке какого-либо класса, которая была запланирована три шага (три окна) тому назад (см. рис. 49), и эта нагрузка должна быть подкреплена двумя преподавателями.

Ячейка окрашенная в грязно-желтый цвет является текущей (имеющий фокус ввода), которую можно редактировать. Перемещать фокус ввода ячейки, как обычно можно с помощью мышки и клавиш со стрелками. Ячейки окрашенные зеленым цветом свидетельствуют о том, что нагрузка преподавателя не превышает предельного значения равного 24-м часам в неделю. Если вдруг, при автоматическом распределении нагрузки, значение нагрузки преподавателя превысит максимум (24 часа), то ячейка будет окрашена красным цветом.

При нажатии на кнопку **Очистить** все ячейки окажутся окрашенными розовым цветом (см. рис. 53), что означает — ячейки ждут назначения своих преподавателей.



	Класс	Предмет	Нагрузка	Колич. групп	ФИО преподавателя	ФИО преподавателя
1	11-А	Ин.яз. 1	2	2		
2	11-А	Ин.яз. 2	2	2		
3	11-А	Ин.яз. 3	1	2		
4	11-Б	Ин.яз. 1	1	2		
5	11-Б	Ин.яз. 2	1	2		
6	11-Б	Ин.яз. 3	1	2		
7	11-В	Ин.яз. 1	1	2		

Рис. 53: Ячейки окрашенные розовым цветом

Теперь собственно о редактировании (назначение преподавателей). Проницательный читатель уже наверное догадался о том, что для вызова окошка со списком преподавателей нужно прибегнуть к стандартным приемчикам (перемещение курсора к нужной ячейке, двойной клик мышкой или нажатие на клавишу **Enter**).

Появившееся справа от таблицы, после осуществления такого действия, окошко **Выбор преподавателя**, изображено на рисунке 54.

Как видно из этого рисунка окошко **Выбор преподавателя** состоит из двух колонок. Первая — число в квадратных скобках, ни что иное, как недельная нагрузка преподавателя, а вторая — **ФИО преподавателя**. Выбор осуществляется с помощью стандартных приемчиков, которые не станем повторять. Если редактируемая ячейка не содержала никакого значения, то в ней появится выбранное из списка значение (**ФИО Преподавателя**), если ячейка содержала какое-то значение (**ФИО Преподавателя**), то это значение будет заменено выбранным. Естественно предположить, что после каждого такого акта выбора, вся колонка с недельной нагрузкой преподавателей будет пересчитываться.

Поскольку нагрузок для классов обычно бывает значительное количество, процесс редактирования таблицы распределения педагогической нагрузки может затянуться весьма на долго. Стоит ли принять распределение нагрузки которое предлагает программа или начать такое распределение редактировать, учитывая «человеческий фактор», решать пользователю.

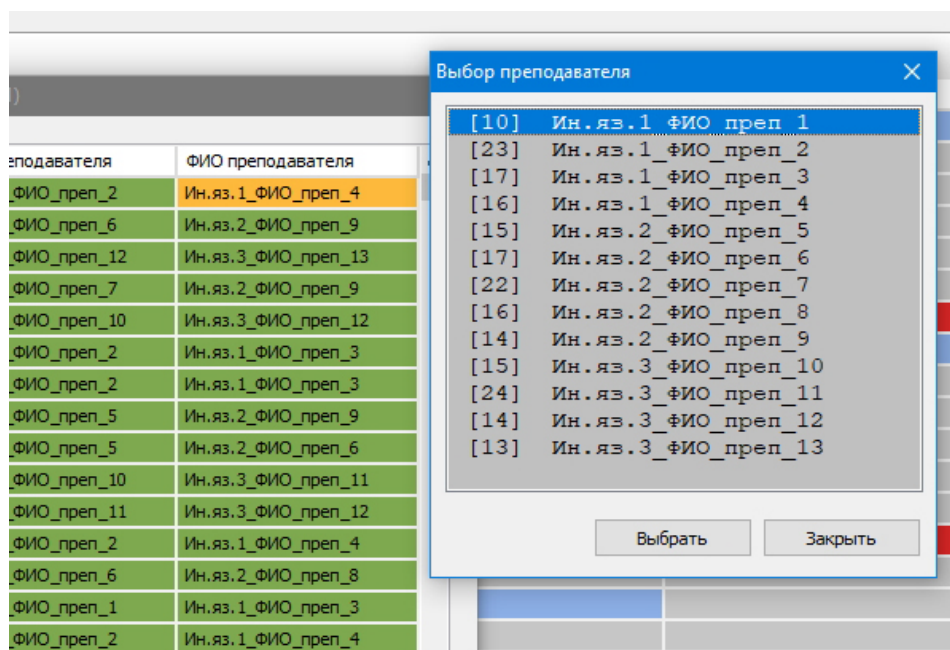


Рис. 54: Окошко Выбор преподавателя

15.5 Окошко Построение расписания для классов с разделением на группы

Для некоторых предыдущих задач мы не стали приводить изображения окошек информирующих о том, что все данные сформированы для начала решения задачи и последующего сообщения — Готово..., украшенное веселой рожицей.

На этот раз мы такое изображение приведем (см. рис. 55).

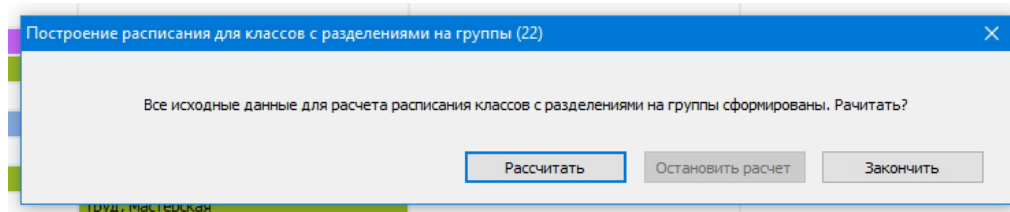


Рис. 55: Окошко Построение расписания для классов с разделением

Связано это с тем, что окошко имеет не две, а три кнопки. Кроме кнопок **Рассчитать** и **Закончить** имеется еще одна кнопка — **Остановить расчет**. При появлении на экране ПК данного окошка, кнопка **Остановить расчет** находится в неактивном состоянии (нажимать на нее бесполезно). Активной она станет только после того, как пользователь нажмет кнопку **Рассчитать**.

При тестировании программы среди миллионов заданий были редкие случаи когда процесс расчета расписания, хоть и ненадолго, но все же затягивался. Именно для такого случая и для нетерпеливого пользователя предназначена эта кнопка. Если процесс расчета не будет завершен за доли секунды и в описываемом окошке появиться вместо улыбающейся рожицы, надпись **Идет расчет...** и крутящееся колесико, а пользователь

не пожелает ждать окончания расчета, ему можно будет нажать на кнопку **Остановить расчет** и расчет будет остановлен.

Вероятнее всего, пользователю не удастся разглядеть ни надпись **Идет расчет...**, ни крутящееся колесико, ни активную кнопку **Остановить расчет** поскольку он будет завершен за доли секунды, но пользователю знать о «такой неприятности» нужно.

После окончания расчета, ячейки расписания с указанием предмета и пары преподавателей будут окрашены желтым цветом. В целях экономии бумаги мы здесь не приводим рисунка с окрашенными ячейками желтым цветом, прося читателей, в том числе и недоверчивых, поверить нам на слово или убедиться в этом самостоятельно, попробовав осуществить расчет с помощью запущенной программы «АФМ: Аннушка».

16 Шестой этап алгоритма

На шестом этапе алгоритма строиться (можно еще сказать достраивается) расписание для всех оставшихся предметов без разделения на группы. У преподавателей нет возможности заявить ультиматум о том в какие дни они будут работать, но для части тех преподавателей у которых низкая недельная нагрузка и низкая максимальная нагрузка в одном классе,³⁴ будет предоставлено один или два выходных (методических дня), в автоматическом режиме. Отметим, что предоставление методического дня не гарантировано для всех без исключения выше названных преподавателей, но вероятность такого предоставления очень высокая.

Кроме того, для части преподавателей есть возможность отказаться от работы в первый урок. Такая часть преподавателей формируется из тех, у кого так же небольшая недельная нагрузка и небольшая максимальная нагрузка в одном классе.

По окончании шестого (последнего) этапа вся сетка расписания оказывается заполненной занятиями без «окон» для всех классов. Построенное расписание представляет из себя четверки — класс, преподаватель, предмет, время. Помещение (аудитория, кабинет) для проведения занятия пока не назначено.

Для решения поставленной задачи требуются исходные данные, формируемые в очередных окнах программы, о которых речь пойдет далее.

16.1 Окно Список учебных предметов

Начиная с этого окна пользователь приступает к решению последней и самой сложной задачи — **Финальное расписание**. Это окно, изображенное на рисунке 56, имеет достаточно простой вид.

После того, как пользователь введет число предметов, изучаемых в школе, он может приступить к редактированию названий этих учебных предметов. Обратим внимание на то, что редактирование числа предметов в соответствующем поле должно заканчиваться нажатием клавиши **Enter**, иначе количество строк таблицы не будет изменено.

При заполнении списка с названиями предметов пользователь должен помнить, что часть нагрузки из учебного плана уже могла «достаться» (уже распределена) совместителям или зарезервирована для предметов в специальных помещениях (Физ-ра, Труд и т.п. (см. подробности выше)). Кроме того в этот список не входят предметы с разделением классов на группы, поскольку для этих предметов нагрузка из учебного плана уже

³⁴Напомним, о том, что такое небольшая нагрузка пояснялось в разделе 6.3 — «Полнота модели учебного процесса».

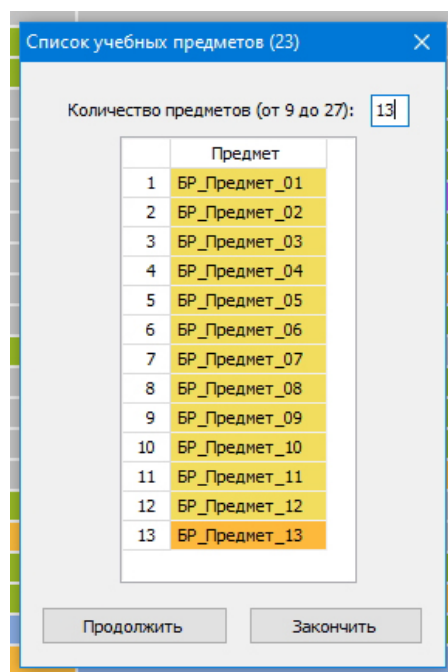


Рис. 56: Окно Список учебных предметов

распределена. Таким образом, в список должны попадать названия предметов нагрузка по которым из учебного плана школы еще требует своего распределения.

При редактировании названий предметов пользователю не стоит забывать о том, что максимальное число символов из которых состоит название предмета не может превышать значения равного 14-ти.

Обратим внимание на минимальное и максимальное значение количества названий предметов формируемых в этом окне.

С минимальным значением равным 9 все относительно просто. Это число появилось потому, что при меньшем числе алгоритм автоматической генерации учебного плана начинал «буксовать» и у разработчиков просто не было выбора. К тому же, число 9 примерно соответствует принятой в настоящее время учебной программе для школ.

За то, что пользователь не сможет добрать нужно количества названий предметов разработчики программы не волновались. Например, предмет — «Химия» легко превращается в пять предметов — «Химия 7 класс», «Химия 8 класс» и т.д. Однако здесь никаких своих рекомендаций мы давать не станем. О том, как правильной обозначать названия предметов пользователю, в отличие от разработчиков, знать лучше.

Максимальное значение количества предметов имеет совою хоть и маленькую но все же историю. Поведаем ее кратко.

В ранних версиях программы максимальное значение вычислялось путем умножения количества классов на число равное 1.3 с последующем округлением результата до целого. Другими словами это значение превышало количество классов в школе на тридцать процентов. Когда минимальное значение классов, принимаемое программой «АФМ: Аннушка», было равно 14 дело обстояло сносно. Главное требование к этому числу — оно не должно быть абсурдным или смешным, выполнялось.

Однако от обиженных потенциальных пользователей программы стали приходить просьбы снизить минимальное число классов, принимаемого программой, по той причине, что школы у них маленькие и 14-ти классов не набирается. При этом, кому-то

требовалось 12, кому-то 10, а кому-то даже 7.

Разработчикам, прямо скажем, через сильное «не хочу» пришлось опуститься до самого технического дна алгоритма составления расписания, а именно принять значение минимума классов равного пяти. Напомним, что наличие всех параллелей, которых к тому же не может быть больше чем количества классов в школе — обязательно, и наличие хотя бы одного класса в каждой из параллелей — обязательно.

После такого нововведения выяснилось, что имеющаяся формула (способ расчета максимума количества названий предметов) в этом случае совершенно не подходит. Например, для 5 классов максимум составлял 7 названий, а минимум 9. Смешно? Конечно смешно . . .

Таким образом, требовался другой способ расчета (другая формула). Пережив легкий творческий припадок разработчики придумали другую формулу. Расскажем о ней.

Обозначим буквой N_{max} максимальное количество названий предметов, а буквой k количество классов в школе. Тогда для количества классов большего или равного девяти, формула приобрела вид:

$$N_{max} = k + \sqrt{k} + 4.$$

Если классов меньше одиннадцати, формула стала иметь вид:

$$N_{max} = 9 + \sqrt{k} + 4.$$

Казалась бы, все в порядке, для 5 классов минимум равен 9, а максимум 15, жить можно. Можно даже хлопать в ладоши и бить себя кулаками в грудь открыв при этом рот. Да вот, что бы не разочаровать пользователей нововведением и прежде чем им позволить рассчитывать расписание с новым диапазоном классов, программу для этого нового диапазона нужно было хотя бы немного протестировать. Как же это было сделано?

Дальнейший текст текущего раздела посвященный короткому рассказу о тестировании нового диапазона классов программы «АФМ: Аннушка» можно считать дополнением к выше изложенному разделу 7 — «О существовании решения математических задач».

Итак, тестирование, естественно с помощью консольной версии программы, было решено провести очень скромненько, без лишних фантазий и фанатизма. Законченный тест для какого-то одного значения общего количества классов в школе состоял из 100 тыс. задач (вариантов исходных данных). Тесты запускались в 20-ти потоках. Итого для каждого значения общего количества классов (5, 6, 7, 8, 9, 10, 11, 12, 13) было решено по 2 миллиона задач.

Так вот, что интересно. В каждом законченном тесте (по 100 тыс. задач) программа один или два, реже три раза выдавала сообщение о том, что решения задачи не существует, несмотря на казалась бы плевые задачки. Разработчикам стало интересно: в чем причина такого безобразия? Они решили живенько «прикрутить» к тексту программы еще несколько строк кода которые выдавали бы значение некоторого параметра задачи в случае несуществования ее решения, с тем что бы, как им казалось, можно было бы как-то повлиять на этот параметр.

Но вот вопрос: какой параметр следует отслеживать (выводить)?

Опираясь на женскую логику и детскую интуицию, решили посмотреть на значение количества названий предметов учебного плана.

Ко всеобщему удивлению и совершенному непониманию, этот параметр, вплоть до значения классов, равного 12-ти оказался равным 13. Причем не просто равным 13, а равным 13 без единого исключения во всех без исключения тестах.

При прогоне тестов с 13-ю классами, изредка кроме цифры 13 (количество названий предметов учебного плана) появлялась цифра 14 или 15.

Потом выяснилось, что при расчете финального расписания решение может не существовать не из-за того, что уже слишком много расставлено предметов на предыдущих шагах расчета, а из-за того, что в учебном плане фигурирует «не то» количество предметов. Та же задача с другим количеством предметов решалась.

Теперь спрашивается: как после такого (увиденного, чего не может быть), возможно не поверить в существование чертей, некоторые из которых изображены на рисунке 57.

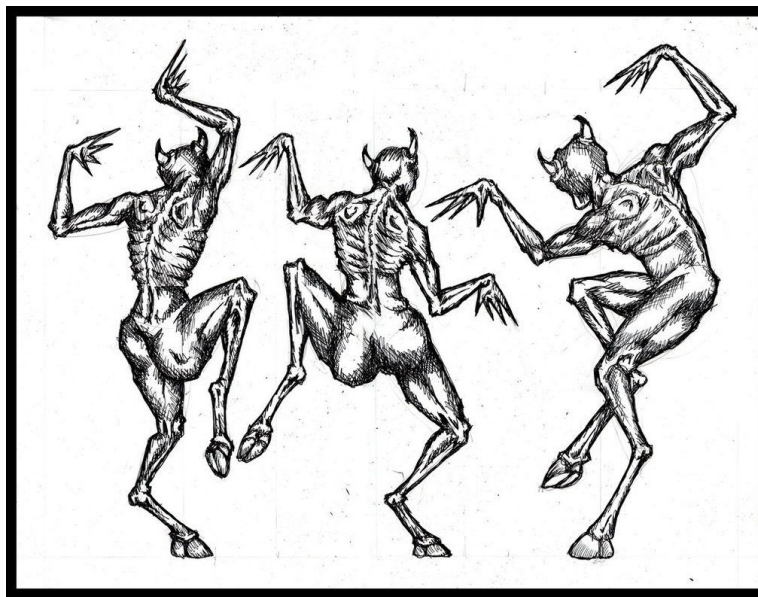


Рис. 57: Пляшущие от удовольствия черты

16.2 Окно Формирование учебного плана для предметов без разделения на группы

Автоматически сформированный программой учебный план, представлен на рисунке 58.

Формирование учебного плана для предметов без разделения на группы (24)

Предельное значение нагрузки учебного плана = 647
Текущее значение нагрузки учебного плана = 647

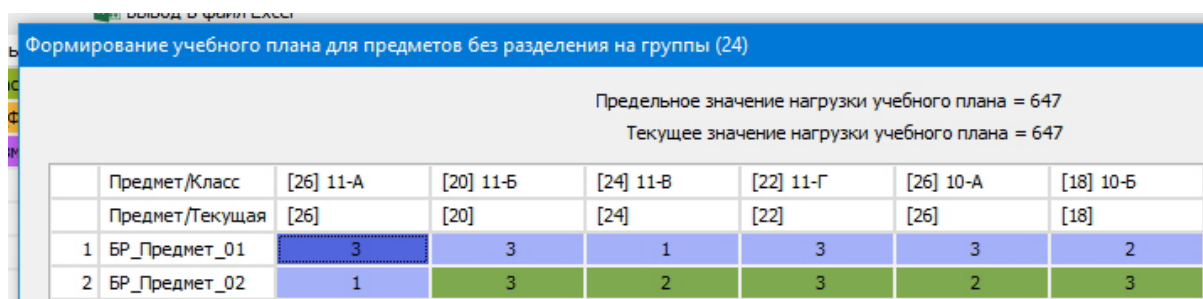
Предмет/Класс	[26] 11-A	[20] 11-B	[24] 11-B	[22] 11-Г	[26] 10-A	[18] 10-B	[25] 10-B	[19] 10-Г	[28] 9-A
Предмет/Текущая	[26]	[20]	[24]	[22]	[26]	[18]	[25]	[19]	[28]
1 БР_Предмет_01	3	3	1	3	3	2			
2 БР_Предмет_02	1	3	2	3	2	3	1	3	
3 БР_Предмет_03	2	2	2	3	3	1	2	2	3
4 БР_Предмет_04	3	2	3	1	2	3	3	3	3
5 БР_Предмет_05	4	2	3	2	3	1	3	1	4
6 БР_Предмет_06	4	3	3	4	3	3	3	2	3
7 БР_Предмет_07	3	2	3	2	3	2	3	3	2
8 БР_Предмет_08	2	1	3	2	3	1	4	1	3
9 БР_Предмет_09	4	2	4	2	4	2	2	3	4
10 БР_Предмет_10							4	1	2
11 БР_Предмет_11									4
12 БР_Предмет_12									
13 БР_Предмет_13									

Продолжить Очистить Закончить

Рис. 58: Окно Формирование учебного плана

В отличие от предыдущего окна, данное окно весьма затейливо. К тому же, пользователю его обязательно придется переделывать и для его правильного формирования нужно будет изрядно потрудиться.

Объяснение начнем с самой верхней строки окна — **Предельное значение нагрузки учебного плана**, которая изображена на очередном рисунке 59, уже показанного окна.



Формирование учебного плана для предметов без разделения на группы (24)

Предельное значение нагрузки учебного плана = 647
Текущее значение нагрузки учебного плана = 647

	Предмет/Класс	[26] 11-А	[20] 11-Б	[24] 11-В	[22] 11-Г	[26] 10-А	[18] 10-Б
	Предмет/Текущая	[26]	[20]	[24]	[22]	[26]	[18]
1	БР_Предмет_01	3	3	1	3	3	2
2	БР_Предмет_02	1	3	2	3	2	3

Рис. 59: Часть окна **Формирование учебного плана**

Как видно из рисунка 59, в данном примере это значение равно числу 647. Откуда взялось это число и что оно означает?

Как известно, для каждого класса санитарными нормами может быть предусмотрена предельная нагрузка учебных занятий в зависимости от параллели класса (возраста учащихся). Если просуммировать предельную нагрузку всех классов и вычесть из этого значения нагрузку, которая уже была включена в расписание для всех классов (уже назначена, расставлена) на предыдущих шагах расчета расписания, то полученное значение пользователь и увидит на экране ПК (в первой строке обсуждаемого окна).

Если пользователь нажмет на кнопку **Очистить**, что ему лучше всего и нужно будет сделать, то вторая сверху строка — **Текущее значение нагрузки учебного плана** покажет число ноль. После такого действия задачей пользователя будет редактирование таблицы учебного плана до тех пор, пока текущее значение не окажется равным предельному. При этом программа будет следить за тем, что бы пользователь не ввел нагрузку отличающуюся от предельной.

Теперь о структуре таблицы и ее цветах. В первой колонке отображаются номера строк, во второй названия учебных предметов, введенные пользователем на предыдущем шаге (в предыдущем окне).

Первая строка (самая верхняя) таблицы, отображает список классов для которых формируется расписание. Причем перед названием каждого класса в квадратных скобках указано число равное предельному значению еще не распределенной учебной нагрузки для этого класса. В процессе редактирования таблицы это число остается неизменным. Во второй (следующей) строке, в квадратных скобках отображается текущее значение нагрузки класса, сформированной пользователем в процессе редактирования таблицы.

Если пользователь понажимает клавиши со стрелками, то он быстро сообразит, что ячейка окрашенная темно-синим цветом является текущей (имеющей фокус ввода), ячейки окрашенные в светло-синий цвет отображают текущую колонку и текущую строку таблицы. Ячейки окрашенные розовым цветом означают то, что для текущего класса и текущего предмета нагрузка имеет нулевое значение. Если нагрузка для соответствующего класса и предмета ненулевая, то ячейка окрашивается зеленым цветом.

Для изменения нагрузки, отображаемой в текущей (имеющей фокус ввода) ячейке, пользователю следует воспользоваться стандартным приемчиком (двойной клик мышкой

или нажатие клавиши **Enter**), после чего появится окошко, изображенное на рисунке 60.

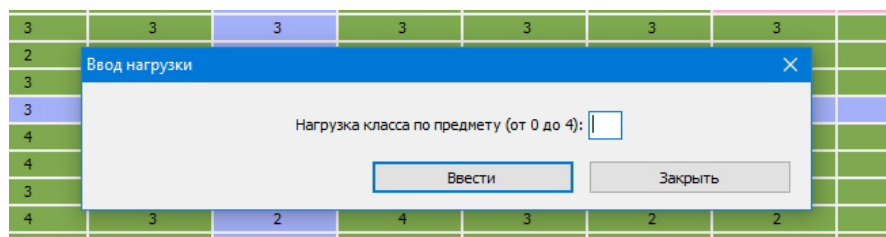


Рис. 60: Окошко редактирования значения нагрузки класса по предмету

Значение нагрузки равно нулю, означает отсутствие нагрузки. Предельное значение нагрузки класса по предмету зависит от количества учебных дней в неделю, заданных пользователем для всей школы. При пяти учебных днях в неделю, эта предельная нагрузка составляет 4 часа, при шести учебных днях — 5 часов.

Таким образом, ничего сложного процесс редактирования учебного плана школы из себя не представляет, если не считать продолжительность самого этого процесса.

Отметим, что поскольку программа следит за соответствием нагрузки каждого класса в отдельности, то следовательно она следит и за числом **Текущего значения учебного плана** для всей школы. Разница в слежении за предельным значением (недопустимостью его превышения) и текущим значением (недопустимостью его занижения) заключается в том, что за превышением нагрузки программа следит постоянно, а недостаток нагрузки программа сможет выявить только после нажатия клавиши **Продолжить**.

Таким образом, если пользователь ошибочно попытается превысить предельное значение нагрузки для какого-то класса, то программа не позволит ему этого сделать, недостаток же нагрузки выявляется только в процессе проверки правильности формирования учебного плана всей школы. И в том и в другом случае программа будет «брыкаться» с помощью окошек — **Сообщение об ошибке**, изображения которых мы здесь приводить не станем в виду банальности этих окошек.

16.3 Окно Список преподавателей

Следующее окно, изображенное на рисунке 61, на счастье пользователей, такое же простое, как и окно **Список предметов**. Правила редактирования в этом окне те же самые, что и в окне **Список предметов**.

Предлагаемое программой количество преподавателей рассчитывается исходя из требуемой нагрузки по каждому предмету и нормой нагрузки для преподавателей, равной 18-ти часам в неделю. Как видно из рисунка 61, предлагаемое программой количество преподавателей может быть скорректировано пользователем исходя из реально сложившейся в школе обстановки с преподавателями.

После того как пользователь введет значение количества преподавателей, не забывая при этом нажать клавишу **Enter** и обращая внимание на допустимый интервал этого значения, он может приступить к редактированию списка (фамилий преподавателей).

Допустимый интервал для количества преподавателей рассчитывается программой с учетом ее прогноза на существование решения задачи. При этом, минимум преподавателей определяется из очевидного условия отсутствия решения (несуществования решения) задачи. Для максимума, естественно, такой прогноз носит весьма приблизительный характер (лишь бы всем преподавателям досталась хоть какая-то нагрузка).

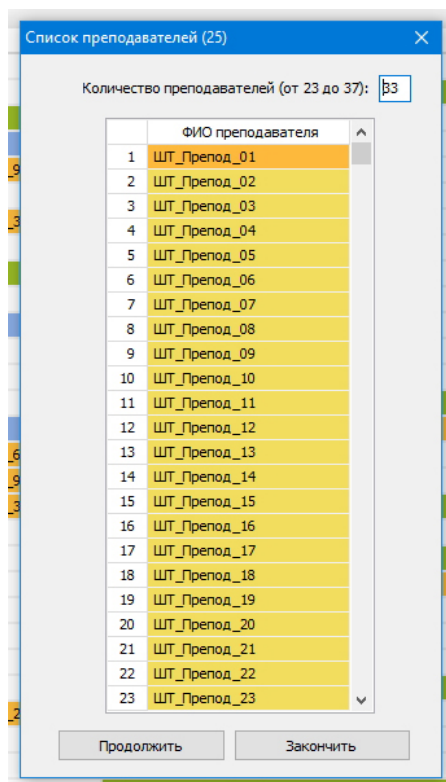


Рис. 61: Окно Список преподавателей

Под ФИО преподавателя программой отводится 23 символа, о чем не следует забывать. Кроме того, ФИО преподавателя не может совпадать с ранее введенными ФИО для преподавателей с разделением классов на группы и ФИО для совместителей. Если фамилии преподавателей для спецпомещений уже заданы пользователем, то программа включит в сравнение и их. Такие множества преподавателей не должны пересекаться. Заканчивается процесс редактирования таблицы, как обычно, нажатием на кнопку **Продолжить**.

16.4 Окно Формирование таблицы компетентности преподавателей

Следующее окно — единственное в программе, имеющее, кроме таблицы, не три, а четыре кнопки. В дополнение к уже известным пользователю кнопкам **Продолжить**, **Очистить** и **Закончить**, присутствует еще одна — **Пропустить**. Как видно из рисунка 62, эта кнопка располагается внизу окна, самой первой (самой левой).

На первый взгляд наличие такой кнопки может показаться странным, но это только на первый взгляд. Дело в том, что единственным назначением этого окна (составления таблицы компетентности) является подготовка исходных данных к автоматическому распределению педагогической нагрузки преподавателей, которая может быть осуществлена на следующем шаге (в следующем окне) программы.

Если пользователь твердо уверен в том, что «бестолковая программа» не сможет распределить педагогическую нагрузку «так, как нужно, с учетом человеческого фактора», то он может не трудиться понапрасну и нажать кнопку **Пропустить**, тогда следующее окно окажется, как бы в состоянии после нажатия кнопки **Очистить**. То есть, заранее подготовленным к «ручному» способу распределения педагогической нагрузки, который потребует от пользователя, пожалуй, самых больших усилий при формировании исход-

	ФИО Преподавателя	Предмет 1	Предмет 2	Предмет 3	Предмет 4
1	ШТ_Преподав_01	БР_Предмет_01			
2	ШТ_Преподав_02	БР_Предмет_02			
3	ШТ_Преподав_03	БР_Предмет_03			
4	ШТ_Преподав_04	БР_Предмет_04			
5	ШТ_Преподав_05	БР_Предмет_04			
6	ШТ_Преподав_06	БР_Предмет_05			
7	ШТ_Преподав_07	БР_Предмет_06			
8	ШТ_Преподав_08	БР_Предмет_06			
9	ШТ_Преподав_09	БР_Предмет_07			
10	ШТ_Преподав_10	БР_Предмет_07			
11	ШТ_Преподав_11	БР_Предмет_08			
12	ШТ_Преподав_12	БР_Предмет_08			
13	ШТ_Преподав_13	БР_Предмет_09			
14	ШТ_Преподав_14	БР_Предмет_09			
15	ШТ_Преподав_15	БР_Предмет_10			
16	ШТ_Преподав_16	БР_Предмет_10			
17	ШТ_Преподав_17	БР_Предмет_10			
18	ШТ_Преподав_18	БР_Предмет_11			
19	ШТ_Преподав_19	БР_Предмет_11			
20	ШТ_Преподав_20	БР_Предмет_11			
21	ШТ_Преподав_21	БР_Предмет_12			

Пропустить Продолжить Очистить Закончить

Рис. 62: Окно Формирование таблицы компетентности преподавателей

ных данных.

Для тех, кто не хочет на следующем шаге (в следующем окне) воспользоваться возможностью «ручного» распределения педагогической нагрузки, надеясь на хотя бы ограниченную (или частичную) «разумность» автоматического распределения нагрузки программой, и не станет нажимать кнопку **Пропустить**, поясним процесс редактирования таблицы компетентности.

Смысл редактирования таблицы достаточно прост — каждому преподавателю сопоставить те учебные предметы, которые он мог бы преподавать. Как правило это пара предметов, например, родной язык и литература, алгебра и геометрия и т.п. В редких случаях преподаватель подготовлен для преподавания трех предметов. В программе, с запасом, предусмотрен случай на четыре предмета. Естественно, что указание хотя бы одного предмета — обязательно.

Смысл расцветки ячеек стандартный. Есть предмет, нет предмета (зеленый, розовый). Текущая ячейка, текущая строка (синий, светло-синий). Вызов окошка выбора предмета то же стандартный — двойной клик мышкой или клавиша **Enter**.

Окошко выбора предмета (**Список предметов**), изображено на рисунке 63.

Как видно из рисунка 63, таблица списка предметов состоит из трех колонок. В третьей колонке (самой правой) размещаются названия предметов, в средней колонке в круглых скобках указано число рекомендованного (не минимального) количества компетентных преподавателей, а в первой колонке (левой), в квадратных скобках, указано число равное количеству преподавателей компетентных преподавать соответствующий предмет, которое сформировал пользователь. Действия пользователя при выборе предмета из списка, точно такие же, как и в аналогичных окошках.

Поскольку после нажатия на кнопку **Продолжить** программа будет контролировать

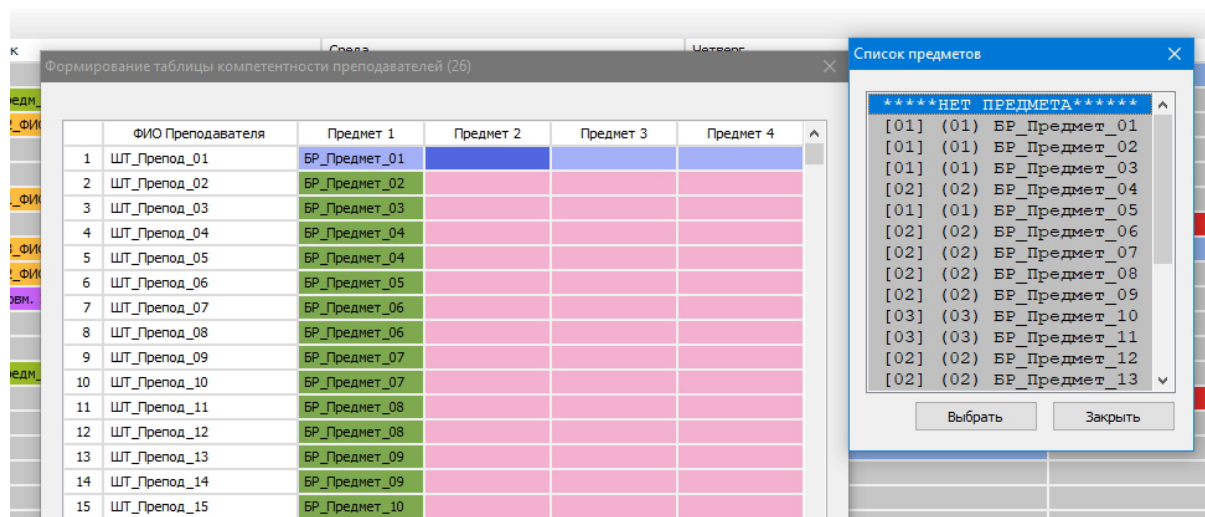


Рис. 63: Окошко Список предметов

необходимый минимум преподавателей способных преподавать каждый предмет, первые две колонки могут оказать существенную помощь для правильного заполнения Таблицы компетентности. Окошко с предупреждением пользователя о том, что по некоторому предмету не хватает преподавателей изображено на рисунке 64.

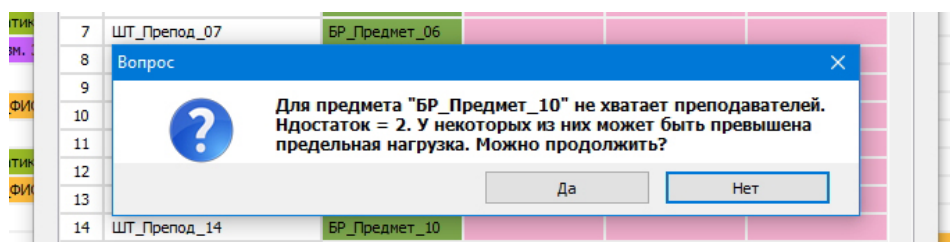


Рис. 64: Окошко Вопрос, оно же — предупреждение

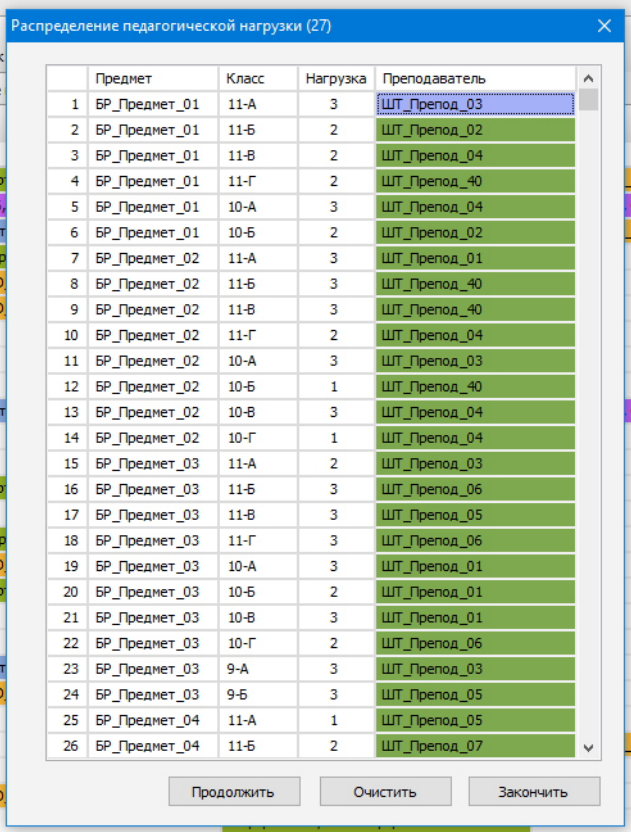
Как видно из этого рисунка, программа может пропустить с ее «точки зрения» не верные исходные данные, но на следующем шаге (в следующем окне) она будет абсолютно неуклонна и нагрузку преподавателя более 26 часов в неделю не пропустит.

Как и обычно, до тех пор пока пользователь не завершит сопоставление предметов для всех преподавателей, программа не позволит ему добраться до следующего окна, если, конечно, он не воспользуется кнопкой Пропустить. Кроме того, программа проконтролирует что бы для каждого предмета имелся хотя бы один компетентный преподаватель.

16.5 Окно Распределение педагогической нагрузки

На следующем рисунке 65 представлена таблица распределения педагогической нагрузки после ее автоматического формирования.

Разумеется, что если на предыдущем шаге пользователем была нажата, умышленно или по неосторожности, кнопка Пропустить, то вся колонка с заголовком Преподаватель окажется окрашенной розовым цветом и без фамилий преподавателей в ячейках. Кроме того, в этом окне пользователю доступна окраска всей колонки в розовый цвет путем на-



Распределение педагогической нагрузки (27)

	Предмет	Класс	Нагрузка	Преподаватель
1	БР_Предмет_01	11-А	3	ШТ_Препода_03
2	БР_Предмет_01	11-Б	2	ШТ_Препода_02
3	БР_Предмет_01	11-В	2	ШТ_Препода_04
4	БР_Предмет_01	11-Г	2	ШТ_Препода_40
5	БР_Предмет_01	10-А	3	ШТ_Препода_04
6	БР_Предмет_01	10-Б	2	ШТ_Препода_02
7	БР_Предмет_02	11-А	3	ШТ_Препода_01
8	БР_Предмет_02	11-Б	3	ШТ_Препода_40
9	БР_Предмет_02	11-В	3	ШТ_Препода_40
10	БР_Предмет_02	11-Г	2	ШТ_Препода_04
11	БР_Предмет_02	10-А	3	ШТ_Препода_03
12	БР_Предмет_02	10-Б	1	ШТ_Препода_40
13	БР_Предмет_02	10-В	3	ШТ_Препода_04
14	БР_Предмет_02	10-Г	1	ШТ_Препода_04
15	БР_Предмет_03	11-А	2	ШТ_Препода_03
16	БР_Предмет_03	11-Б	3	ШТ_Препода_06
17	БР_Предмет_03	11-В	3	ШТ_Препода_05
18	БР_Предмет_03	11-Г	3	ШТ_Препода_06
19	БР_Предмет_03	10-А	3	ШТ_Препода_01
20	БР_Предмет_03	10-Б	2	ШТ_Препода_01
21	БР_Предмет_03	10-В	3	ШТ_Препода_01
22	БР_Предмет_03	10-Г	2	ШТ_Препода_06
23	БР_Предмет_03	9-А	3	ШТ_Препода_03
24	БР_Предмет_03	9-Б	3	ШТ_Препода_05
25	БР_Предмет_04	11-А	1	ШТ_Препода_05
26	БР_Предмет_04	11-Б	2	ШТ_Препода_07

Продолжить Очистить Закончить

Рис. 65: Окно Распределение педагогической нагрузки

жания на кнопку **Очистить**, на тот случай, если пользователь обнаружит, по его представлениям, всю «глупость» автоматического распределения нагрузки и нецелесообразность частичного редактирования такого распределения.

Теперь осталось только повторить — как начать редактирование колонки. Но проницательный читатель об этом уже наверное и сам догадался. Рисунку, со списком преподавателей для выбора, присвоен очередной номер равный 66.

С целью экономии бумаги и времени читателя, комментировать процесс выбора преподавателя мы не станем, отметим лишь, что максимальная недельная нагрузка преподавателя, текущее значение которой заключено в квадратные скобки, составляет 26 часов. Если вдруг при автоматическом распределении нагрузки какому-то преподавателю будет распределено нагрузки более 26-ти часов, то ячейки с этим преподавателем окажутся окрашенными в красный цвет и снижать нагрузку такому преподавателю нужно будет пользователю в «ручном» режиме. Впрочем такая же раскраска ячеек может появиться и в случае редактирования Таблицы распределения педагогической нагрузки самим пользователем. Ему же и предстоит исправлять свои ошибки.

Естественно предположить, что в процессе редактирования пользователем Таблицы распределения педагогической нагрузки может возникнуть ситуация когда кому-то из преподавателей, нагрузки не будет назначено вовсе. На этот случай разработчики программы, так же предусмотрели проверку, но в качестве ограничителя выбрали достаточно странное число равное единице. При этом, по всей видимости, они руководствовались чисто формальным подходом. Действительно, полное отсутствие нагрузки (число 0) — явно бессмысленно, а число 1 просто странно, но формально не бессмысленно.

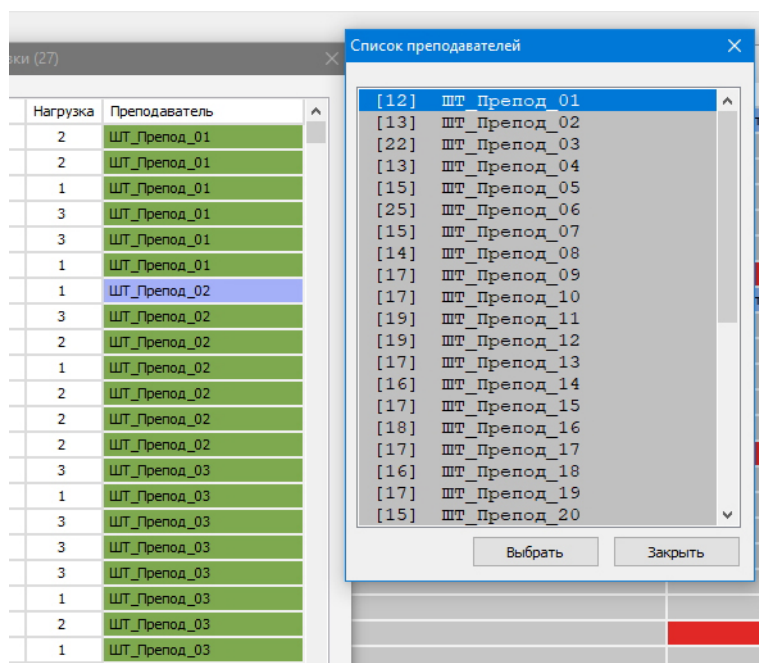


Рис. 66: Окошко Список преподавателей

16.6 Окно Запреты на работу в первый урок

Следующее окно является настолько банальным, что кроме упоминания о нем, пожалуй, ничего говорить и не нужно. Тем более приводить его рисунок.

Хотя стоит еще раз отметить, что любые ограничения легко могут привести к *несуществованию решения задачи* и поэтому программа по умолчанию никому возможности отказа от первого урока сама не предоставляет и будет уговаривать пользователя не предоставлять преподавателям возможности появляться в школе только ко второму уроку весьма настойчиво, можно даже сказать — бесцеремонно. Такое поведение программы объясняется тем, что предоставление кому-либо возможности не появляться в школе к первому уроку почти всегда влечет за собой лишение другого преподавателя методического дня (выходного).

Однако, по некоторым вопросам программа будет совершенно непреклонна. Так, она не позволит освободить от первого урока преподавателя с нагрузкой более 18-ти часов в неделю или с максимальной нагрузкой у некоторого класса более 3-х часов для сетки расписания 5×7 и более 4-х часов для сетки расписания 6×6 .

В то же время, несмотря на «уговоры» программы, пользователь имеет полное право и возможность поэкспериментировать на этот счет с получением решения.

16.7 Окошко Построение расписания для классов без разделения на группы

Вопреки наметившейся тенденции экономии бумаги, рисунок следующего окошка показать все же придется. Тем более, что разговор пойдет не столько об окошке, сколько о процессе расчета финального расписания.

Как наверное уже предположил проницательный читатель, первоначально окошко будет содержать сообщение о том, что все готово для начала расчета и три кнопки —

Рассчитать, Остановить расчет и Закончить. Разумеется, первоначальное состояние кнопки **Остановить расчет** — неактивна.

На рисунке 67 представлено обсуждаемое окошко во время расчета расписания (еще до того, как появиться очень довольная и улыбающаяся рожица).

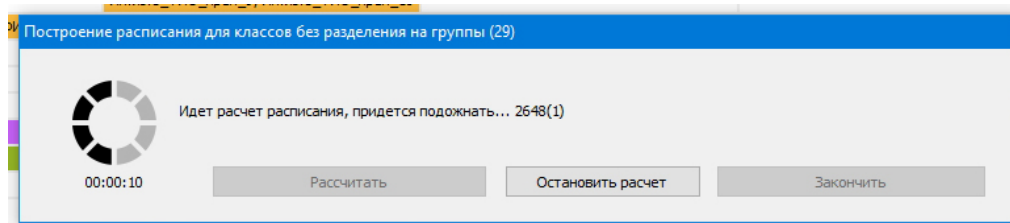


Рис. 67: Окошко Построение расписания для классов без разделения на группы

Первое, что может прийти в голову, глядя на этот рисунок — нахрена козе баян и к чему столько наворотов?

Поясним содержание этого окошка, двигаясь слева направо.

Прежде всего обращает на себя внимание крутящееся колечко (на представленном здесь рисунке оно разумеется не крутится) свидетельствующее о том, что процесс расчета идет. Под этим колечком располагаются «цифровые часы» показывающие время расчета, который все еще продолжается. Далее надпись — **Идет расчет расписания, придется подождать...**, за которой следует некая служебная информация, предназначенная в первую очередь для разработчиков программы.

Так вот, нахрена козе баян?

Дело в том, что если количество классов (по меркам программы) невелико — от 12 до, скажем 17, то пользователь всю эту «красоту» может просто не увидеть, поскольку расчет, как правило закончиться за доли секунды. Однако, при количестве классов подходящих к своему максимальному значению, обстановка может резко измениться. А именно. Если пользователю повезет (именно повезет), то расчет может закончиться за доли секунды или за секунды. А если не повезет, то расчет расписания может затянуться на несколько десятков минут. Причем предсказать заранее длительность расчета нет никакой возможности. Вот именно на такой случай и рассчитано, крутящееся колесико, «цифровые часы» и прочие удовольствия. Для особо нетерпеливых предназначена кнопка **Остановить расчет**, прибывающая во время расчета, в отличии от остальных кнопок, в активном состоянии. Отметим, что единственным способом остановить расчет, разумеется не считая отключение ПК от питания (розетки), является нажатие на кнопку **Остановить расчет**. Комбинация клавиш **Alt+F4** и любые кликанья мышкой по окну — бесполезны, не помогут.

Но мы будем надеяться на то, что пользователь довольно быстро увидит окошко, изображенное на следующем рисунке 68.

Остается только добавить, что разработчики программы решили ячейки наполненные результатом финального расписания никак не раскрашивать. И без того, раскраска вкладки **Расписание классов** сильно напоминает окрас африканского попугая.

Окончательный «боевой окрас» главного окна программы после завершения расчета расписания можно видеть на рисунке 69.

На этом все шаги «Древнеегипетского» алгоритма составления школьного расписания заканчиваются.

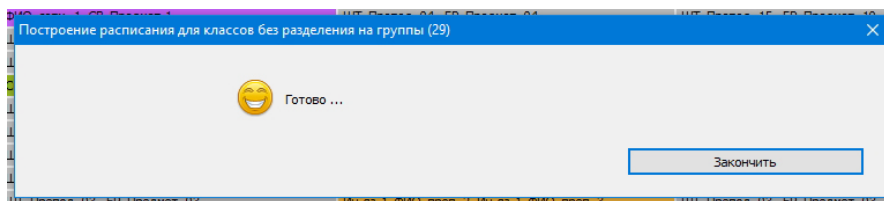


Рис. 68: Окно Построение расписания для классов без разделения на группы

Рис. 69: Главное окно программы после окончания расчета

Общим правилом для всех ранее описанных шагов, является правило — *у каждого класса в один день не может быть более одного урока по какому-либо предмету*. Кроме того, общим правилом для преподавателей является то, что в любой день каждый преподаватель может проводить занятия по нескольким предметам (естественно в разные моменты времени), в том числе и у одного класса, например, алгебру и геометрию, физику и химию, родной язык и литературу и т.п..

На этой оптимистической ноте заканчивается наш тягомотный рассказ о самом сложном и вычислительно-емким (расчетно-емким) разделе составления школьных расписаний. Далее начинается разговор о сравнительно простых вещах, можно сказать — банальных и весьма незатейливых.

17 Заключительный этап составления расписания

После получения финального расписания, построенного на шестом этапе алгоритма, для всех запланированных занятий школы (кроме занятий в специальных помещениях), сопоставляются (назначаются, распределяются) помещения где будут проводиться эти занятия. Так же для проведения занятий в специальных помещениях, расписание для которых было получено на втором этапе алгоритма, назначаются преподаватели.

Кроме того, на **заключительном этапе составления расписания** для всей школы, каждый преподаватель, желающий проводить элективные курсы, может решить для себя когда ему будет удобно их проведение. Время для таких кусов может быть зарезервировано на четвертом этапе алгоритма. И если этот преподаватель сможет набрать себе группу учеников, то он самостоятельно поставит свой элективный курс в расписание, вместе с подобранным им же самим помещением, которое в нужный момент времени не занято проведением других учебных занятий.

Из уже построенного (составленного, рассчитанного), с помощью выше описанного алгоритма, финального расписания можно легко построить расписание преподавателей и расписание кабинетов, что очень часто предусмотрено в программах составления школьных расписаний. Поскольку расписание преподавателей создается программой автоматически, подробно останавливаться на нем мы не станем, а покажем лишь вкладку Расписание преподавателей, изображенную на рисунке 70.

Рис. 70: Часть вкладки Расписание преподавателей

Благодаря этой вкладке можно быстро посмотреть кому из преподавателей достались методические дни, а кому нет.

Далее обратим внимание на первую (слева) колонку с фамилиями штатных преподавателей, по предметам без разделения класса на группы, фрагмент которой изображен на рисунке 71.

ШТ_Препо_03: Н=22; М=3	Урок 7	Класс = 11-В, П
	Урок 2	Класс = 11-А, П
	Урок 2	Класс = 11-Г, П
	Урок 4	Класс = 9-А, Пр
	Урок 5	Класс = 11-В, П

Рис. 71: Фрагмент первой колонки

Из этого рисунка видно, что после фамилии преподавателя идут два обозначения (надписи) $H = 22$ и $M = 3$. Первая надпись показывает недельную нагрузку преподавателя, а вторая максимальную нагрузку этого преподавателя в каком-либо из классов. Как ранее отмечалось, эти два параметра влияют на «решение» программы о предоставлении преподавателю выходного (методического дня).

17.1 Назначение преподавателей в специальных помещениях

Ранее в разделе 12.3 нами было обещано объяснить: каким образом осуществляется назначение преподавателей в специальных помещениях?

Для этого еще раз внимательно посмотрим на рисунок 70 и увидим там кнопку Преподаватели спецпомещений, изображенную на следующем рисунке 72.

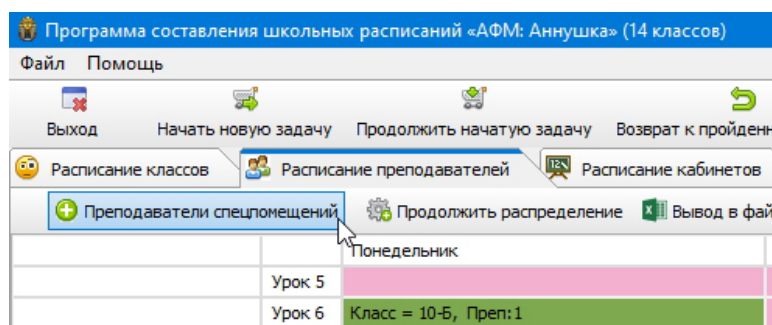


Рис. 72: Кнопка Преподаватели для спецпомещений

Именно эту кнопку и предстоит нажать пользователю для начала процесса назначения преподавателей для спецпомещений. Сам же процесс начинается с вопроса изображенного на очередном рисунке 73.

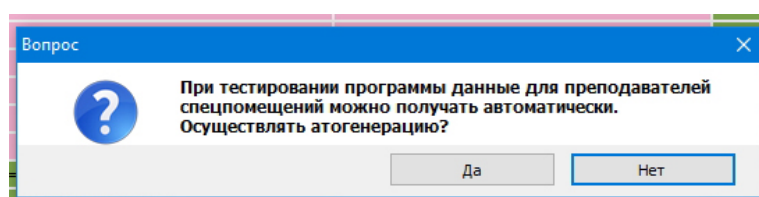


Рис. 73: Вопрос об автоматической генерации данных

Отвечать на этот вопрос отрицательно имеет смысл если пользователю требуется исправить ранее уже введенные данные. При первоначальном же знакомстве с программой лучше нажать кнопку Да.

Если пользователь так и сделает, то на панели управления изучаемой нами вкладки появится надпись, изображенная на рисунке 74.

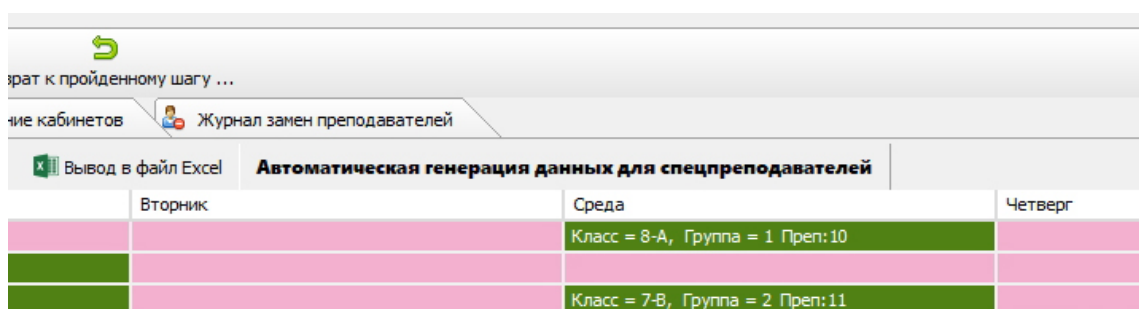


Рис. 74: Надпись информирующая об автоматической генерации данных

17.1.1 Окно Список преподавателей для спецпомещений

Очередное, появившееся окно после положительного ответа на вопрос об автоматической генерации данных, изображено на рисунке 75.

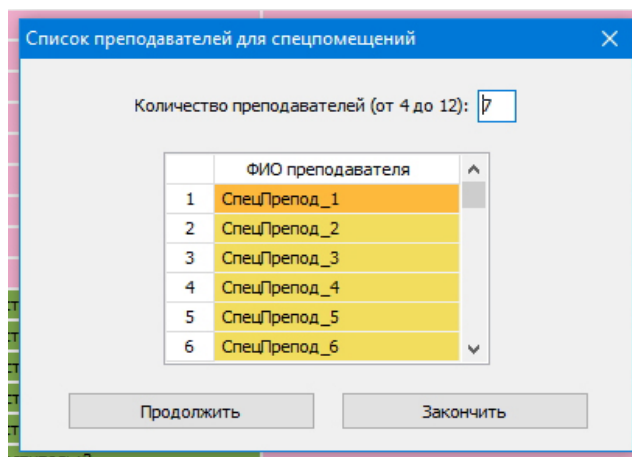


Рис. 75: Вопрос об автоматической генерации данных

В этом окне минимум преподавателей определяется исходя из того соображения, что для каждого спецпомещения будет назначен только один преподаватель, а максимум из того, что будет назначено три преподавателя. В приводимом примере преподаватели подбираются для четырех спецпомещений поэтому минимум равен четырем, а максимум двенадцати.

Естественно, что пользователь редактируя данные этого окна уже должен для себя определиться с тем, какие преподаватели будут преподавать в каких помещениях.

Приемы редактирования данных в этом окне стандартные, поэтому повторяться не будем. Отметим лишь, что изменение количества преподавателей в соответствующем поле ввода должно завершаться нажатием клавиши **Enter**.

Кроме того, еще раз напомним о том, что прежде чем перейти с следующему окну программа проверит на совпадение фамилий введенных преподавателей с фамилиями преподавателей из других групп (совместители, преподаватели для классов с разделением на группы и без разделений). Совпадения по преподавателям недопустимы.

17.1.2 Окно Таблица компетентности преподавателей для спецпомещений

Таблица компетентности преподавателей для спецпомещений, изображенная на рисунке 76, которая появиться после нажатия на кнопку **Продолжить**, редактируется уже хорошо известным для пользователя способом.

При редактировании этой таблицы, скорее всего, пользователю придется сначала нажать на кнопку **Очистить**, поскольку компетенции преподавателей формируются программой автоматически, причем довольно «бестолковым» способом, а далее после двойного клика мышкой по соответствующей ячейке или после нажатия на клавишу **Enter**, из появившегося окошка, изображенного на рисунке 77, пользователю программы придется выбрать нужного преподавателя.

Как видно из рисунка 77, число в квадратных скобках, расположенное перед фамилией преподавателя, отображает количество сопоставлений этому преподавателю некоторого специального помещения. Число равное нулю, означает то, что преподавателю

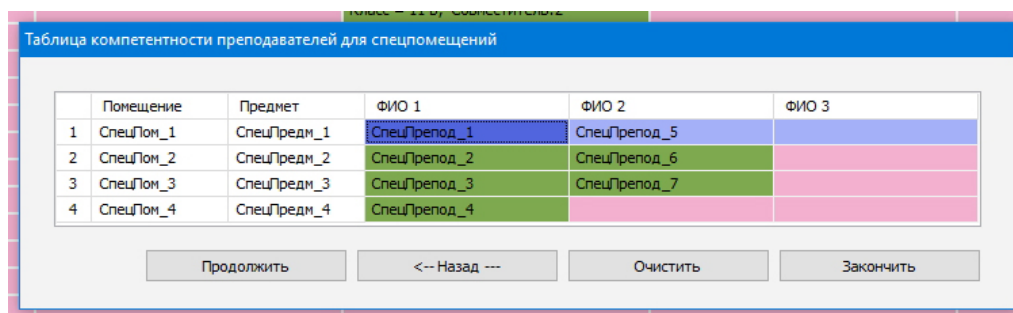


Рис. 76: Окно Таблица компетентности преподавателей для спецпомещений

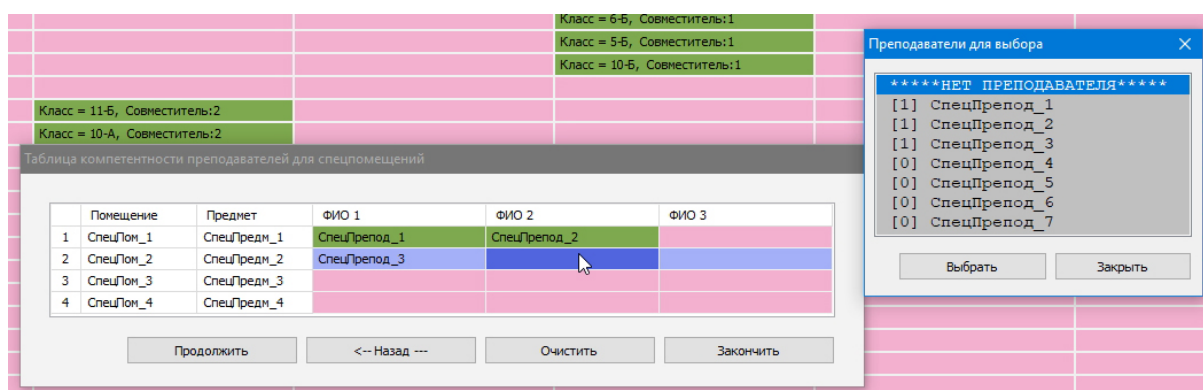


Рис. 77: Окошко Преподаватели для выбора

еще не сопоставлено ни одного спецпомещения. Важно то, что количество сопоставлений (преподаватель — спецпомещение) не может превышать единицы. Другими словами преподаватель может учить (работать, преподавать) только в одном спецпомещении.

Таким образом, задача пользователя заключается в том, что бы каждому преподавателю сопоставить только одно спецпомещение. После выполнения этой задачи и нажатия на кнопку **Продолжить**, программа, перед тем как появиться следующее окно, проверит каждому ли преподавателю сопоставлено спецпомещение и каждому ли спецпомещению сопоставлен преподаватель.

Как видно из рисунка 76 и 77 в обсуждаемом окне появилась ранее не встречавшаяся кнопка — **<-- Назад --**. Ее назначение должно быть понятно без особых объяснений.

Назначение этой кнопки весьма банально — если пользователь ошибся при вводе данных или передумал, то у него есть возможность вернуться к предыдущему окну и исправить ошибочно сформированные данные. Например, если пользователь допустил ошибку в фамилии преподавателя он может такую ошибку исправить.

17.1.3 Окно Распределение нагрузки среди спецпреподавателей

Если проверка, сформированных пользователем данных в предыдущем окне прошла удачно, на экране ПК появиться окно, изображенное на рисунке 78 — **Распределение нагрузки среди спецпреподавателей**.

В этом окне имеются вкладки для каждого спецпомещения. Перемещаться с одной вкладки на другую можно с помощью клика мышкой по соответствующему корешку вкладки или с помощью клавиш со стрелками после перемещения фокуса ввода (с помо-

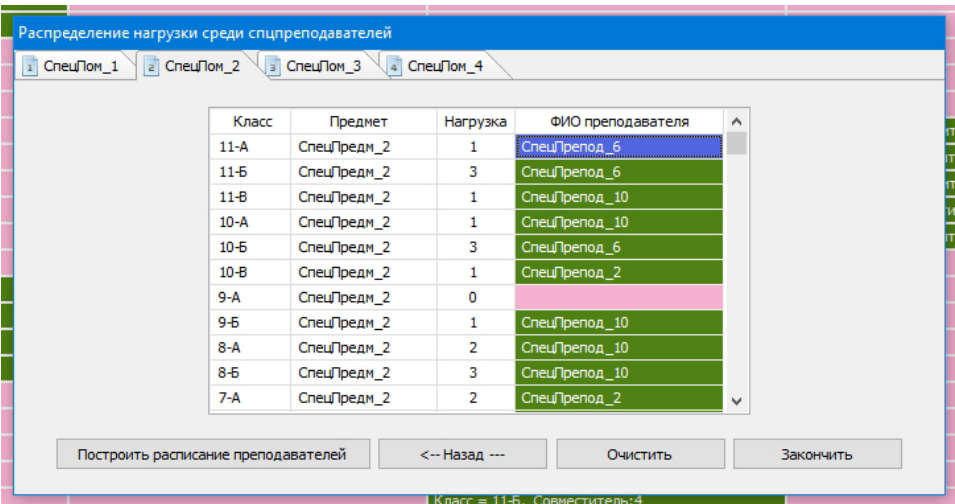


Рис. 78: Окно Распределение нагрузки среди спецпреподавателей

пью клавиши Tab) на соответствующий корешок.

Содержание каждой вкладки состоит из таблицы, разделенной на четыре колонки. Как видно из рисунка 78 в первой колонке отображаются названия классов, во второй названия предметов, в третьей нагрузка каждого класса по предмету, а в последней преподаватель который будет проводить соответствующий урок.

Если в спецпомещении для проведения уроков предусмотрено больше одного преподавателя, то в случае автоматического формирования данных, распределение преподавателей по классом программа осуществляет случайным образом. Поэтому пользователю придется всю колонку ФИО преподавателя подвергнуть тщательному редактированию. Как обычно перед редактированием колонки пользователю доступна кнопка Очистить. Но можно обходиться и без нее.

На следующем рисунке 79 показано окошко Преподаватели для выбора, которое появляется поле нажатия на клавишу Enter или двойного клика мышкой по соответствующей ячейке.

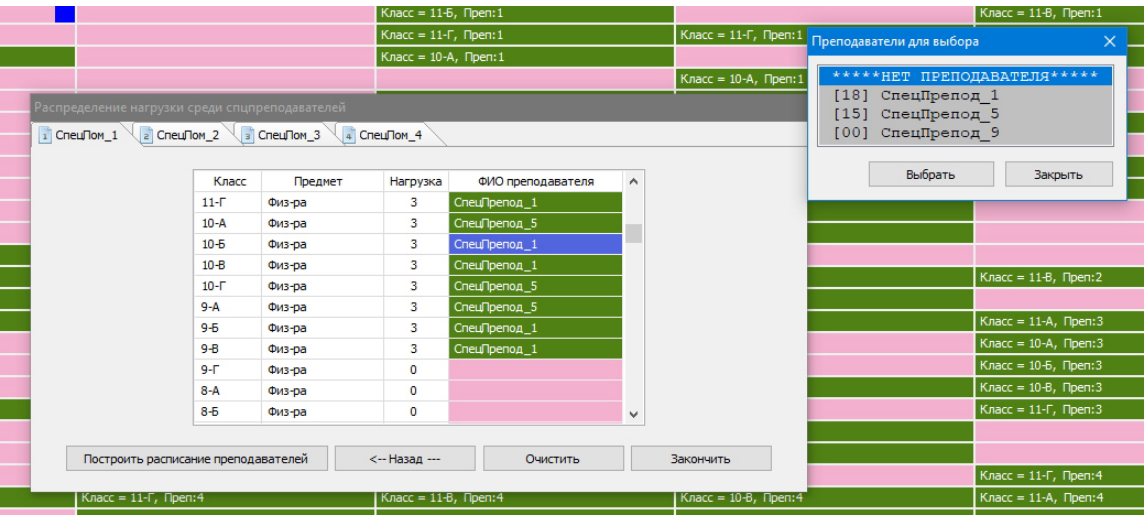


Рис. 79: Процесс редактирования преподавателей спецпомещений

Из представленного рисунка видно, что во-первых, в окошке для выбора преподавателей представлены только те преподаватели которые были сопоставлены в таблице компетентности соответствующему спецпомещению, и во-вторых, ячейки таблицы окна **Распределение нагрузки среди спецпреподавателей** для которых нагрузка равна нулю, окрашены розовым цветом. Если пользователь попытает в такую ячейку поместить ФИО преподавателя, то программа его грязно (тактично) обругает.

Проницательный читатель уже наверное догадался, что в окошке **Преподаватели для выбора** перед каждой фамилией преподавателя в квадратных скобках указана его недельная нагрузка. Кроме того, внимательный читатель должно быть обратил внимание на то, что в окне, представленном на рисунках 78 и 79 точно так же, как и в предыдущем окне, имеется кнопка <-- Назад --.

Назначение кнопки **Продолжить распределение**, представленной на рисунке 80 и располагающейся на панели управления вкладки **Расписание преподавателей**, тоже весьма очевидно — если пользователь прервет свою работу во время процесса распределения преподавателей в спецпомещениях (составления расписания для спецпреподавателей), то приступив к этой работе заново ему не придется снова вводить все исходные данные, а перед ним откроется окно на котором работа была прервана.

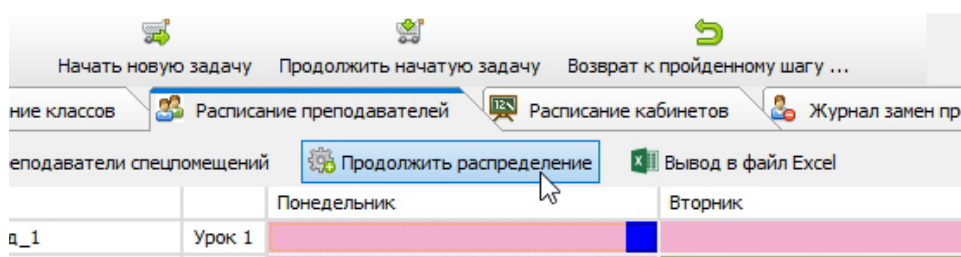


Рис. 80: Кнопка Продолжить распределение

И наконец, отметим еще одну деталь, которая может облегчить работу пользователя при вводе исходных данных. На рисунке 81, изображающим окно **Распределение нагрузки среди спецпреподавателей** показана раскраска колонки подлежащей редактированию после нажатия кнопки **Очистить**.

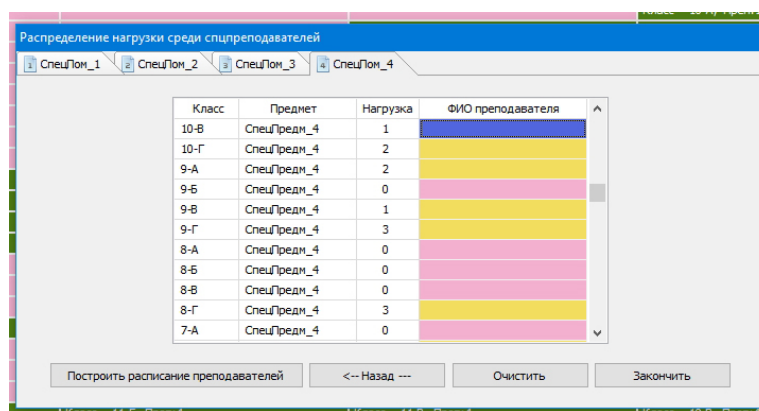


Рис. 81: Раскраска колонки ФИО преподавателя

Из этого рисунка видно, что колонка окрашена тремя цветами. Темно синий цвет, как и обычно используется для отображения текущий ячейки (имеющий фокус ввода),

розовый цвет, как было отмечено ранее, используется для окраски ячеек в которые недопустимо помещать фамилии преподавателей, а желтым цветом окрашены ячейки в которые нужно поместить фамилии преподавателей, но пользователь этого еще не сделал. После того как пользователь это сделает ячейки будут окрашены зеленым цветом, что можно видеть на рисунке 78.

После окончания распределения преподавателей по классам (после окончания распределения нагрузки среди спецпреподавателей) пользователь может нажать на кнопку **Построить расписание преподавателей** и после проверки программой правильности введенных данных во вкладке с расписанием преподавателей появятся новые строчки. При этом эти строчки будут располагаться начиная с самого верха таблицы. Соответствующий рисунок, с целью экономии бумаги, мы здесь приводить не станем, тем более потому, что пользователь может попробовать самостоятельно в этом убедиться.

17.2 Вкладка Расписание кабинетов

Как говорилось ранее, бесплатная версия программы «АФМ: Аннушка» предназначена только для демонстрации возможностей «Древнеегипетского» алгоритма по составлению школьного расписания. Задача сопоставления занятиям (набору: время, преподаватель, класс, предмет) кабинетов, хотя и требует значительного времени если решать эту задачу в «ручном» режиме, является весьма банальной. По этой причине пользователь нажав на ярлык вкладки **Расписание кабинетов** увидит текст изображенный на рисунке 82.

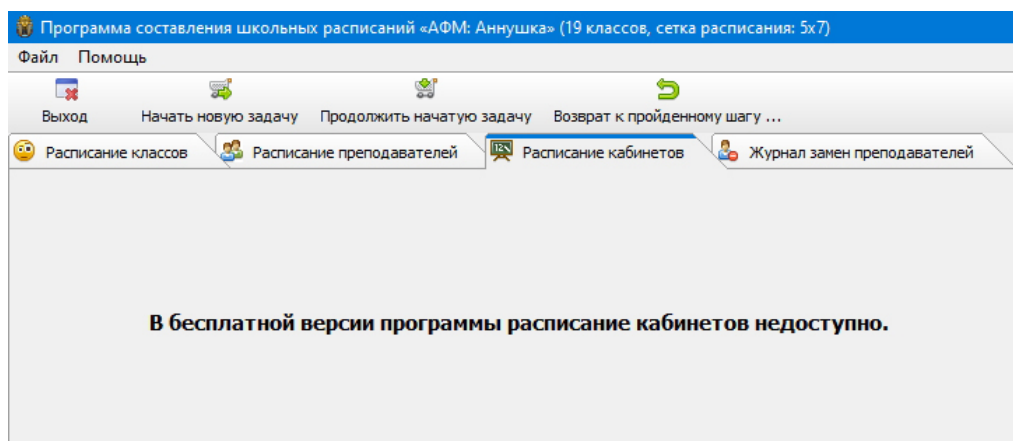


Рис. 82: Вкладка Распределение кабинетов

Вместе с тем, для начинающего разработчика программы по составлению школьных расписаний эта задача должна представлять определенный интерес. В автоматизации работы по сопоставлению занятиям кабинетов существенную помощь может оказать решение, так называемой, задачи о назначениях. Именно для таких разработчиков мы здесь приведем (см., например, [8]) ее строгую формулировку.

Эта задача состоит в следующем. Пусть имеется n работ и n кандидатов для выполнения этих работ. Назначение кандидата i на работу j связано с затратами c_{ij} ($i, j = 1, 2, \dots, n$). Требуется найти назначение кандидатов на все работы, дающие минимальные суммарные затраты; при этом каждого кандидата можно назначать только на одну работу и каждая работа может быть занята только одним кандидатом.

Иначе говоря, решение этой задачи представляет собой перестановку $(p_1, p_2, \dots, p_n)$ чисел $(1, 2, \dots, n)$; каждое из производимых назначений описывается соответствием $i \rightarrow$

p_i ($i = 1, 2, \dots, n$). Условия единственности при этом автоматически выполняются, и нашей целью является минимизация суммы

$$\sum_{i=1}^n c_i p_i,$$

по всем перестановкам $(p_1, p_2, \dots, p_n)$.

Эта задача имеет замечательное свойство. А именно. Известно огромное количество способов (алгоритмов) ее решения. Знакомство хотя бы с частью из таких алгоритмов, будет чрезвычайно полезным для начинающего разработчика программы по составлению школьных расписаний.

18 Дополнительные возможности программы

«АФМ: Аннушка»

18.1 Кнопки окон для ввода данных Закончить и кнопка панели управления Продолжить начатую задачу

Вполне прогнозируемой ситуацией является то, что в процессе работы с программой, прежде чем перейти к следующему окну для формирования очередной порции исходных данных, пользователю захочется более внимательно ознакомиться с уже полученными (промежуточными) результатами, решаемой общей задачи путем внимательного разглядывания вкладок **Расписание классов** и **Расписание преподавателей**. Такую возможность программа конечно же должна ему предоставить. Кроме того не исключен случай перерыва на обед или прекращения работы пользователем до следующего дня (следующего раза).

В таких случаях у пользователя, при работе с любым окном программы, есть право и возможность нажать на кнопку **Закончить**, находящуюся в нижнем правом углу окна. После такого нажатия текущее окно исчезнет, а пользователь может либо нажать на кнопку панели управления **Выход**, либо покрутить сетки уже сформированных расписаний вверх и вниз внимательно разглядывая полученный результат.

В качестве примера, рисунок 83 представляет сетку расписания для шестых и седьмых классов после расчета расписания для специальных помещений и формирования санитарных ограничений, в которой одна часть ячеек окрашена салатovým цветом, а другая красным.

Если пользователь выйдет из программы и далее после некоторого перерыва, например, на следующий день, снова ее запустит, то сетка расписания окажется в том состоянии при котором работа последнего сеанса была прекращена.

Как уже читатель наверное догадался, для продолжения решения задачи составления расписания занятий, пользователю следует нажать на кнопку панели управления **Продолжить начатую задачу**, изображенную на рисунке 84.

После такого нажатия появится окно, являющееся очередным окном для ввода исходных данных. Естественно, что если пользователь не собирается покидать программу и нажмет на кнопку **Продолжить начатую задачу** после ознакомления с сеткой расписания, результат будет точно таким же.

Теперь еще раз внимательно посмотрим на рисунок 72. Проницательный читатель уже наверное догадался, что кнопки **Преподаватели спецпомещений** и **Начать новую задачу**

Рис. 83: Часть сетки расписания для sixth и seventh классов

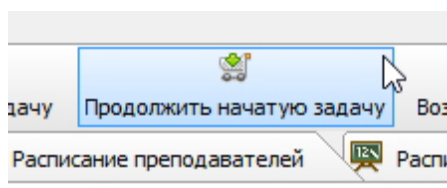


Рис. 84: Кнопка панели управления Продолжить начатую задачу

являются полными аналогами. Кроме того, кнопка Продолжить распределение, изображенная на рисунке 72 и кнопка Продолжить начатую задачу, изображенная на рисунке 84, так же являются полными аналогами.

В добавок кнопки Закончить, появляющиеся в окнах ввода данных основного алгоритма (составления расписания для классов) и, появляющиеся в окнах ввода данных при составлении расписания для преподавателей в специальных помещениях, имеют не только одинаковые названия, но и одинаковый смысл.

18.2 Кнопка панели управления Возврат к пройденному шагу...

Дело в том, что практически все алгоритмы, задействованные в программе, для построения учебного расписания очень активно пользуются генератором случайных чисел. То есть, осуществляют случайный поиск. Это приводит к тому, что если пользователь попробует решить ту же задачу с точно такими же исходными данными, то практически наверняка он получит от программы другое решение.

Трудно сказать, является ли это недостатком программы или ее достоинством. Тем не менее, при таких обстоятельствах у пользователя появляется возможность решать одну и ту же задачу, с одними и теми же исходными данными, ранее сформированными им для решения этой задачи, неограниченное число раз. Другими словами до тех пор пока

решение задачи, по тем или иным критериям, пользователю не придется по душе или не станет любо. Но кнопка **Продолжить** в любом из окон, обязывает пользователя двигаться только вперед (к следующему по счету окну).

Как же вернуться пользователю к уже пройденному окну, не начиная новой задачи?

Понятно, что прежде всего следует избавиться от очередного окна на экране ПК, поскольку это окно монопольно захватывает на себя все действия пользователя. Для этого жмем на кнопку **Завершить**, после чего окно исчезнет, и поэтому кнопка панели управления **Возврат к пройденному шагу...** станет доступна.

Для тех кто был не очень внимателен или забыл о существовании такой кнопки панели управления, покажем ее на рисунке 85.

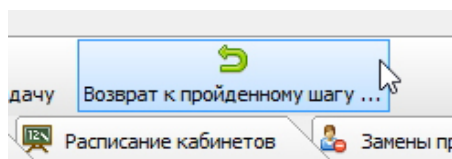


Рис. 85: Кнопка панели управления **Возврат к пройденному шагу...**

После нажатия на эту кнопку на экране ПК появиться окно, изображенное на рисунке 86, с помощью которого можно выбрать шаг (окно) к которому пользователь хочет вернуться по тем или иным причинам.

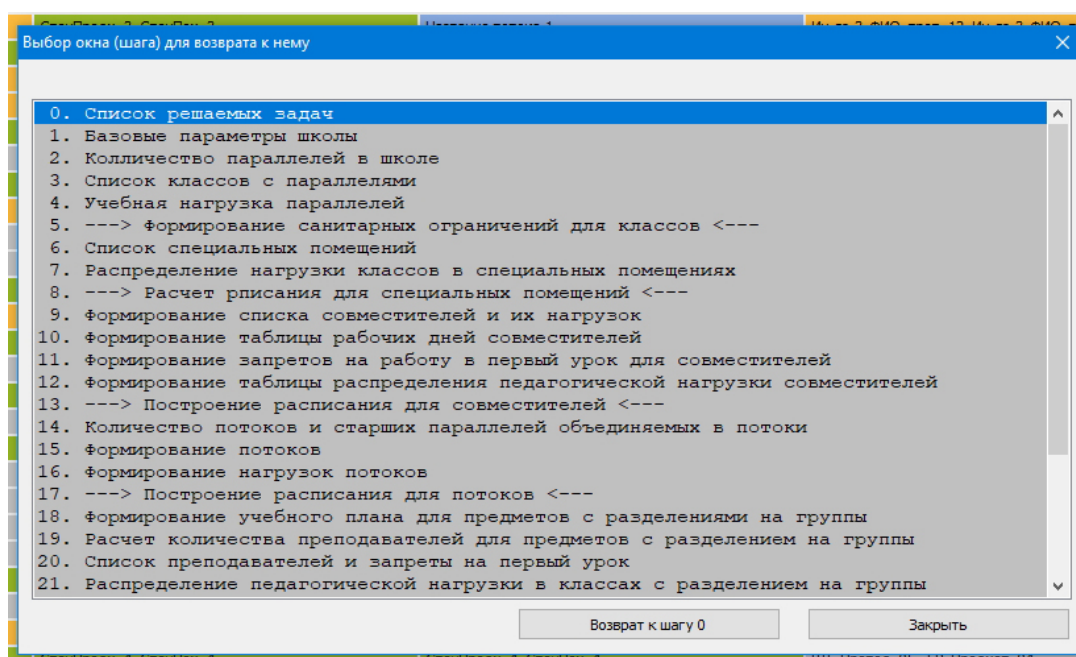


Рис. 86: Окно **Выбор окна (шага) для возврата к нему**

Каждым из пунктов этого окна является название окна (шага), только из тех шагов, которые пользователем уже были пройдены. Для удобства эти пункты пронумерованы. Выбор шага осуществляется стандартным для программы способом — двойной клик мышкой по нужной строчке, либо после подведения курсора к нужной строчке, с помощью нажатия клавиши **Enter**, либо после подведения курсора к нужной строчке,

нажатием на кнопку **Возврат к шагу XX**. На место XX программа, естественно, подставит выбираемый номер.

Перед тем как программа позволит пользователю осуществить свой выбор, она, по причине своей «бестолковости», попросит у пользователя подтверждения его намерений с помощью вопроса, изображенного на рисунке 87 и предупредит его о том, что все исходные данные следующие за выбранным шагом придется вводить заново.

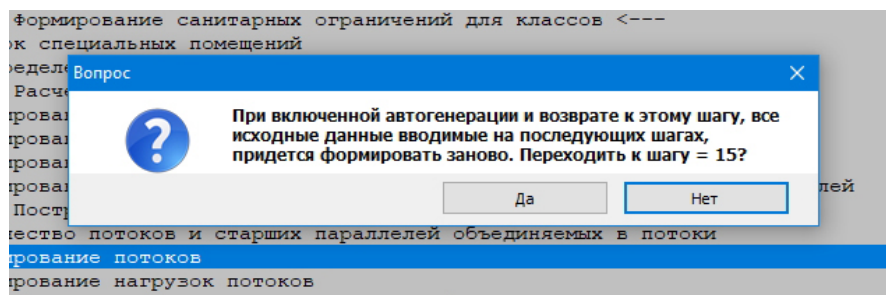


Рис. 87: Окошко с вопросом

При выборе им одного из предлагаемых пунктов, перед пользователем появиться окно на которое он указал, и у него появится возможность продолжить решение общей задачи начиная с выбранного места. В частности, пользователь может подправить ранее введенные исходные данные или еще раз решить (пересчитать) какую-то отдельную задачу без исправления исходных данных, из списка отмеченных в окне — **Список решаемых задач**. При этом, как ранее было отмечено, решение, скорее всего, окажется другим (будет значительно отличаться от ранее полученного).

Обратим внимание на то, что в окне **Выбор окна (шага)** для возврата к нему некоторые шаги слева и справа выделены стрелочками, что можно увидеть на рисунке 88. Это сделано для того, что бы пользователю было удобней искать окошки в которых программа предлагает ему только произвести некоторый расчет без исправления исходных данных.

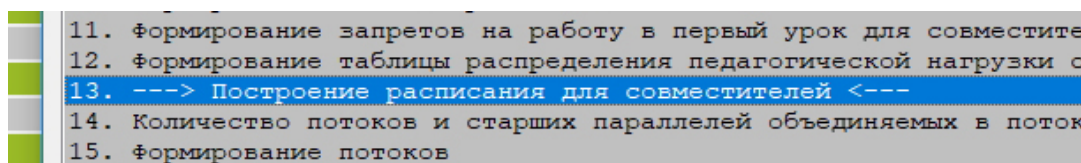


Рис. 88: Шаг алгоритма выделенный стрелочками

Есть еще и другой способ возврата к уже пройденным шагам алгоритма, без повторного ввода уже сформированных исходных данных (если они были введены пользователем), но о нем будет рассказано чуть позже.

18.3 Пункты главного меню **Запомнить/Вспомнить** текущее расписание

При возврате пользователя к уже пройденному шагу решения задачи или пересчете расписания, нет никакой гарантии, что полученное новое расписание окажется лучше (милее), по тем или иным критериям пользователя, предыдущего (ранее рассчитанного).

Таким образом, пользователь нуждается в некой «страховке» во время его борьбы хорошего с еще лучшим.

Одним из способов такой «страховки» может оказаться полезным пункт главного меню программы — **Запомнить текущее расписание**, изображенный на рисунке 89.

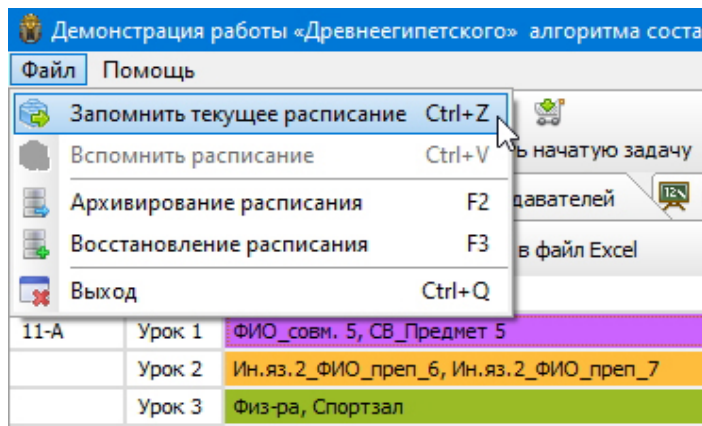


Рис. 89: Пункт главного меню **Запомнить текущее расписание**

Как видно из рисунка 89, следующий пункт меню — **Вспомнить расписание** находится пока в неактивном состоянии. Причина этому простая — пользователь еще ни разу не запоминал текущее расписание. Как только он это сделает, пункт **Вспомнить расписание** станет активным (ярким) и пользователь при желании сможет восстановить расписание в том состоянии (на том шаге), как оно было запомнено. Отметим, что «запоминание» и «вспоминание» расписания будет предварять вопрос программы на подтверждение таких действий. При «вспоминании» расписания окошко с вопросом будет выглядеть так, как изображено на рисунке 90.

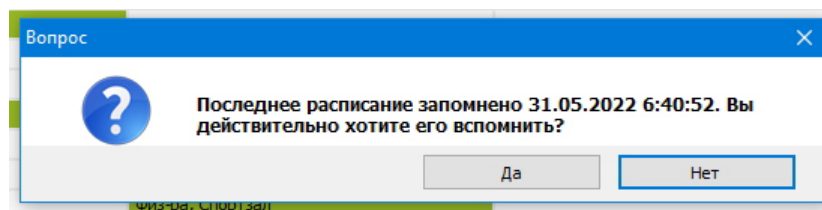


Рис. 90: Окошко с вопросом

18.4 Кнопка панели управления Автогенерация данных

Для не очень внимательных читателей и тех людей кто не обладает феноменальной памятью, на рисунке 91 еще раз представим изображение этой кнопки.

После обсуждения назначения кнопки панели управления **Возврат к пройденному шагу...** нами было упомянуто о том, что есть еще один способ возврата к пересчету уже решенных задач.

Скептически настроенный читатель может сказать: — «Вот еще один пример борьбы хорошего с еще лучшим». Но не стоит делать поспешных выводов. Обратим внимание читателя на то, что в случае возврата к некоторому шагу (окну) алгоритма с помощью

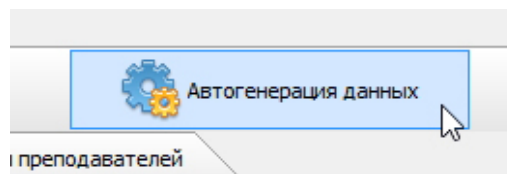


Рис. 91: Кнопка панели управления Автогенерация данных

кнопки **Возврат к пройденному шагу...**, исходные данные сформированные пользователем на этом шаге, не считая окошек с предложением решить очередную задачу, сохраняются. Другими словами программа оставляет их не тронутыми. Но на следующем шаге (в следующем по счету окне) программа тупо начинает заменять данные введенные пользователем, своими — автоматически генерируемыми, что может совершенно не совпадать с желанием этого пользователя. Если такими данными являются всего-навсего одно или два числа, то с подобной ситуацией еще можно было бы смириться, но если программа уничтожит, например, таблицу распределения педагогической нагрузки на заполнение которой пользователь потратил целый день работы, то...

При этом ситуация, когда у пользователя может возникнуть желание пересчитать все ранее полученные расписания с абсолютно теми же данными, является вполне рабочей и встречающейся на практике достаточно часто. Как же тогда быть?

Как быть, как быть... да запретить... и дело с концом.

Вот для такого запрета программе — не менять уже введенные данные пользователя, и предназначена кнопка **Автогенерация данных**. Поскольку программа весьма «бестолкова» и с первого раза не понимает чего от нее хотят, после того, как пользователь кликнет мышкой по кнопке **Автогенерация данных**, она симулируя слабоумие, переспросит его с помощью окошка, изображенного на рисунке 92.

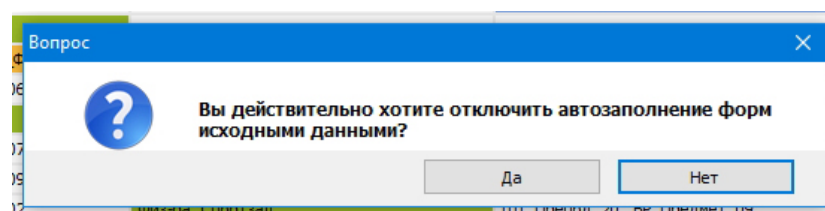


Рис. 92: Окошко с вопросом

Если пользователь подтвердит свое желание, кнопка **Автогенерация данных**, затоив тайную обиду на пользователя потухнет, а окошко с вопросом исчезнет с экрана ПК. Далее пользователь может нажать на кнопку **Возврат к пройденному шагу...** и выбрать из окна — **Выбор окна (шага) для возврата к нему**, желаемый для него шаг алгоритма. Здесь сразу стоит отметить, что подлая кнопка — **Автогенерация данных**, вспомнив о своей обиде и желая отомстить пользователю, запретит ему использование любых управляющих элементов во всех окнах программы, предназначенных для формирования исходных данных, кроме кнопок **Продолжить** и **Закончить**.

Находясь под такими наложенными санкциями, которые, как известно, идут только на пользу и ни на что не влияют, работающему с программой имеет смысл выбирать для возврата только те шаги (пункты) алгоритма, с помощью которых производится расчет той или иной задачи. Как можно видеть из рисунка 93 такие пункты отмечены в

окне выбора текстовыми стрелочками, а для большей наглядности и привлечения к ним внимания, на рисунок нанесены еще и красные изгибающиеся стрелки.

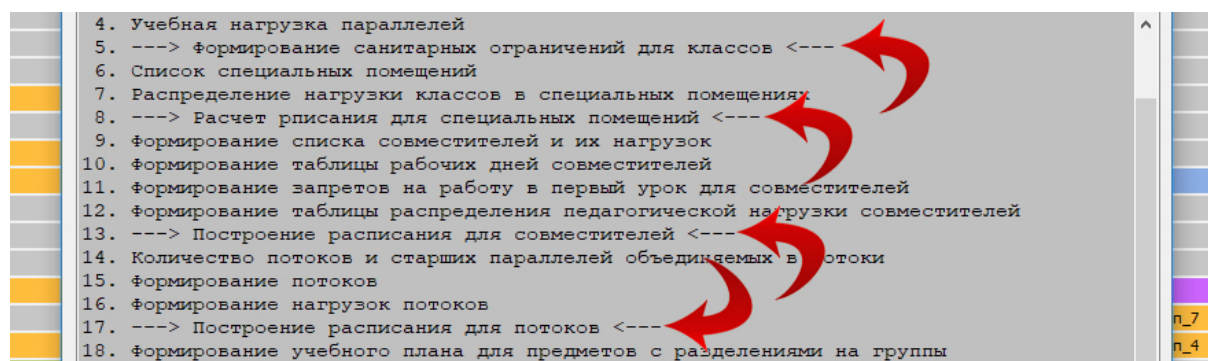


Рис. 93: Часть окна Выбор окна (шага) для возврата к нему

Действительно, какой смысл выбирать окно в котором практически все запрещено, хотя формально никаких запретов на выбор таких окон — нет. Например, такое окно можно выбрать только для того, что бы вспомнить (еще раз посмотреть) введенные пользователем данные и убедиться в правильности этих данных.

Далее подумаем над возможной ситуацией, связанной с запретом автогенерации исходных данных. Предположим, что пользователь начал решать свою задачу, дошел (имея ввиду шаги алгоритма) до некой середины, например до формы (окна) номер 15 из всех 29-ти, и не долго думая, осуществляет запрет автогенерации данных. Потом нажимает кнопку **Продолжить начатую задачу**. И что дальше? А дальше не будет не только автогенерации данных, но и будет запрет пользователю вводить свои данные. Что может содержаться в появившемся окне? Либо ничего, если пользователь в первый раз решает задачу, либо мусор, оставшийся от ранее решаемой задачи не совместимый с измененными (переделанными) пользователем исходными данными. В следствии чего пользователю, могла бы представиться возможность понаблюдать за поведением спятившей программы.

Таким образом, мы приходим к выводу о том, что *осуществлять запрет автогенерации можно только в том случае, если пользователем были сформированы (введены) все без исключения исходные данные*. А когда, все без исключения, такие данные введет пользователь? Ответ прост — после того как он нажмет кнопку **Продолжить** на форме **Запреты на работу в первый урок** под номером 28, а последней показанной формой окажется форма **Построение расписания для классов без разделения на группы** под номером 29. Следовательно, в случае несоблюдения пользователем выше описанного правила, быть ему обруганным программой с помощью окошка, изображенного на рисунке 94 и показанного вопреки нашего установленного правила — не приводить рисунков с руганью программы.

На этом наше повествование о кнопке **Автогенерация данных** и борьбе хорошего с еще лучшим можно было бы закончить. Но хотелось бы, а точнее, совершенно необходимо обратить внимание потенциального пользователя программы «АФМ: Аннушка» на еще один аспект расчета школьных расписаний.

Как ранее уже было отмечено, самым, если так можно выразиться, тяжелым (трудным) шагом расчета расписания является шаг — **Финальное расписание**. Известно, и строго математически доказано, что финальное расписание, при условии отсутствия каких-либо ограничений существует всегда и рассчитывается практически мгновенно для любых разумных размерностей (любого разумного количества классов в школе).

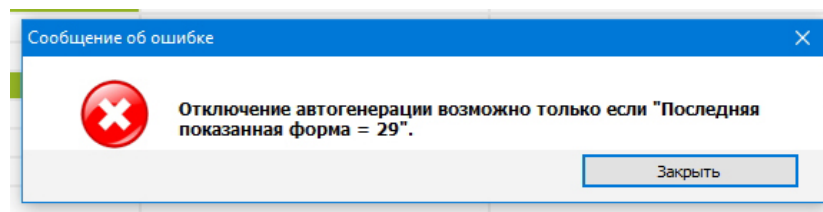


Рис. 94: Сообщение об ошибке

Так почему же расчет финального расписания отмечается, как самый трудный? Все дело в дополнительных ограничениях. Эти ограничения условно можно разделить на две группы. Во-первых, до начала расчета уже на четверть или хуже того, на половину сетка расписания занята расставленными занятиями. Во-вторых, преподавателям с невысокой нагрузкой необходимо предоставлять пресловутые методические дни (выходные). При этом нет ответа на вопрос — а существует ли вообще расписание при названных ограничениях? Если быть совсем точным, ответ на поставленный вопрос относится к категории, так называемых NP-трудных задач.

Таким образом, приступая к решению задачи о финальном расписании остается только надеется на то, что искомое решение задачи вообще существует.

И ежику понятно, что невозможно найти того, чего не существует.

А как можно оценить процент везения (вероятность того, что занятые ячейки сетки расписания, которые получены, в некотором смысле, случайным образом) являются теми ограничениями при которых решение задачи существует? Кроме того (при этом), известно, что бездумное и оголтелое предоставление всем желающим методических дней по их выбору, просто обрушивает такую вероятность.

Теоретические выкладки по этому поводу авторам программы «АФМ: Аннушка» не известны. Остается только обратиться к результатам вычислительного эксперимента. А что нам показывает эксперимент?

Эксперимент показывает, что подозрение на несуществование решения при выше перечисленных ограничениях, оправданы примерно в нескольких случаях из тысячи случайно сформированных тестовых заданий, поскольку решение остальных 999 — 997 задач было получено (найдено).

Из этого, полученного с помощью эксперимента факта, можно сделать важный практический вывод. А именно. Если уж пользователю достались ограничения при которых решение задачи не существует, то такие ограничения без особого ущерба нужно просто поменять. И вот здесь-то, самое время вспомнить о кнопке **Автогенерация данных**. При отключенной автогенерации, т.е. при сохранении всех введенных пользователем исходных данных, следует возвратиться к ранее пройденному шагу, например, к шагу 8 — **Расчет расписания для специальных помещений** или к какому-либо еще другому, изменив путем пересчета, ранее полученные ограничения.

С помощью такого приемчика, вероятность успеха (получение ограничений при которых решение существует) увеличиться на три порядка.

И последнее на что стоит обратить внимание. При выходе из программы она не запоминает состояние кнопки **Автогенерация данных**³⁵, поэтому в случае отключения автогенерации пересчет расписания нужно доводить до конца. Иначе при следующем запуске программы ее состояние окажется на «полпути» к окончанию расчета (в том месте от-

³⁵Это сделано намеренно.

куда был осуществлен выход), а восстановить состояние кнопки **Автогенерация данных** не получится, поскольку номер последней показанной формы не будет равен 29-ти.

Вот теперь, наше повествование о борьбе хорошего с еще лучшим и пресловутой **Автогенерации данных** мы заканчиваем, плавно переходя к другой мультке³⁶ программы.

18.5 Кнопка Вывод в файл Excel

После получения финального расписания у пользователя может возникнуть желание самостоятельно «доделать» («допилить») его путем построения расписания для кабинетов (классов, аудиторий). Теперь, после того, как все самые сложные расчеты закончены, этот процесс не может вызвать каких-либо серьезных затруднений, если использовать возможности всеми любимой и популярной программы **Microsoft Excel**.

Кроме того в программе «АФМ: Аннушка» отсутствуют собственные средства вывода полученных результатов на печать (принтер). **Excel** же справляется с этой задачей виртуозно. Поэтому не воспользоваться его возможностями было бы весьма опрометчиво.

Процесс переноса полученного расписания из программы «АФМ: Аннушка» в программу **Microsoft Excel** осуществляется с помощью кнопки **Вывод в файл Excel** панели управления вкладки **Расписание классов**. После нажатия на эту кнопку на экране ПК появится окошко с вопросом, изображенное на рисунке 95.

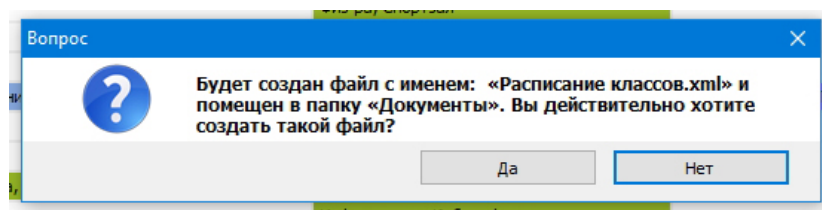


Рис. 95: Окошко с вопросом

Ответив положительно на этот вопрос, пользователю остается только воспользоваться этим файлом, расположенном в папке **Документы**. Поскольку потенциальный пользователь программы «АФМ: Аннушка» наверняка хорошо знаком с программой **Microsoft Excel**, мы здесь не станем комментировать ее использование. Действительно — такого учить, только портить.

18.6 Кнопка Найти подстроку

При формировании дополнительных расписаний в программе **Microsoft Excel**, может оказаться очень полезной кнопка **Найти подстроку**, находящаяся на той же панели управления вкладки **Расписание классов**.

Окошко, которое появится после нажатия на эту кнопку, изображено на рисунке 96.

После ввода пользователем подстроки в поле ввода **Текст**, например, фамилии преподавателя, и нажатия кнопки **Найти**, окошко примет другой вид (см. рис. 97), и будет оставаться таким, до тех пор пока не отыщутся все ячейки с введенным текстом. При этом каждая найденная ячейка будет постоянно переходить в состояние — текущая, и следовательно будет отмечаться синим квадратиком.

³⁶Мультка — то же, что и цапка. По английски — фича. Функциональная возможность.

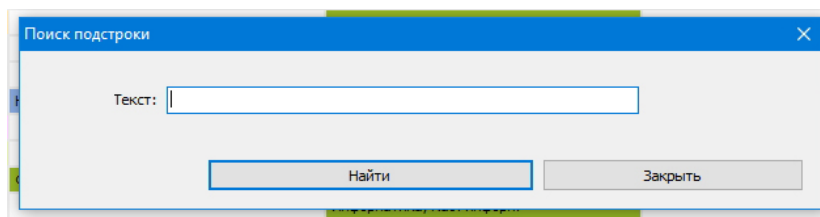


Рис. 96: Окошко Поиск подстроки

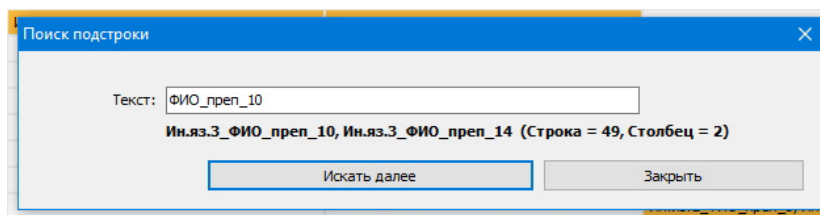


Рис. 97: Окошко Поиск подстроки

18.7 Просмотр длинного содержания ячеек

Если содержание ячейки вкладки Расписание классов окажется слишком длинным и поэтому такое содержание не будет видно целиком непосредственно в сетке расписания, то для более удобного рассмотрения содержания ячейки можно воспользоваться специальным окошком, изображенном на рисунке 98.

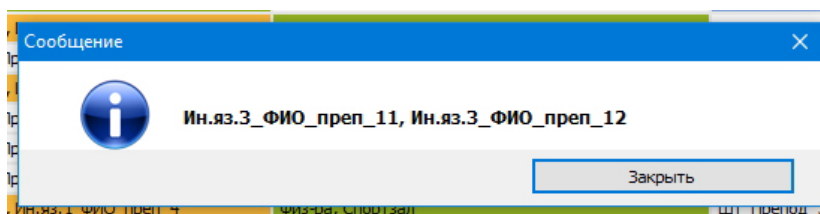


Рис. 98: Окошко Сообщение с содержанием ячейки

Как наверное уже догадался проницательный пользователь, вызов окошка осуществляется с помощью стандартного приемчика — двойной клик мышкой по ячейке или после подведения курсора к нужной ячейке, нажатие клавиши **Enter**.

Аналогичное окошко и с помощью того же приемчика будет появляться и во вкладке Расписание преподавателей.

Далее обсудим еще одну важную тему, на которую пользователи иногда не обращают должного внимания.

18.8 Архивирование и восстановление данных

В самом начале 90-х годов прошлого века у автора этих строк состоялся поучительный диалог с одним из своих коллег.

- А если персональный компьютер сломается?
- Персональные компьютеры не ломаются!

Именно так тогда оценивалась перспектива длительного, а главное, надежного хранения информации на тогдашних ПК³⁷. Теперь такого пожалуй никто не скажет. Пользователи ПК с длительным опытом работы на них, знают, что выход почти подряд из строя двух жестких дисков — дело вполне обычное, не говоря уже о переустановке операционной системы. Из чего, неотвратимо следует вывод.

Дублируйте важные данные на разные носители!

Именно для такого дублирования и предназначены пункты главного меню программы Архивирование расписания F2 и Восстановление расписания F3, изображенные на рисунке 99.

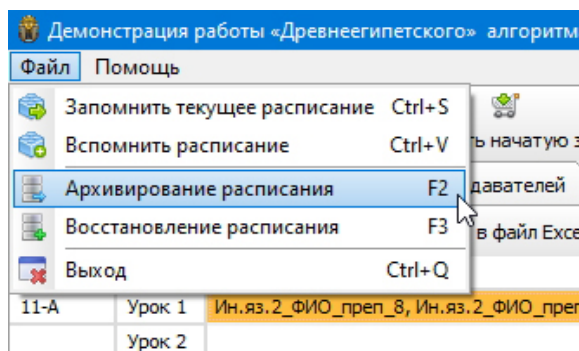


Рис. 99: Пункты главного меню Архивирование и Восстановление

18.8.1 Пункт главного меню Архивирование расписания

После нажатия клавиши F2 или соответствующего выбора пункта главного меню программы появиться окно, изображенное на рисунке 100. Точнее говоря, окно аналогичное такому окну, в зависимости от подключенных к ПК носителей информации.

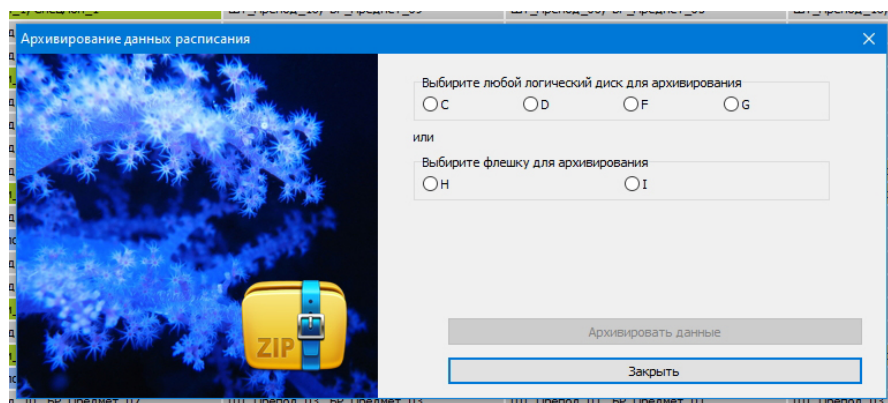


Рис. 100: Окно Архивирование данных расписания

В этом окне пользователю предстоит выбрать физический носитель, имеющийся на его ПК, для архивирования. Подчеркнем, что по смыслу всей этой затеи с архивированием оно должно производиться именно на другой физический, а не логический, носитель. Удобнее всего для этой цели использовать флешку.

³⁷Скорее всего под впечатлением постоянных и регулярных поломок (сбоев) тогдашних супер-ЭВМ.

После того как выбор носителя будет сделан, кнопка **Архивировать данные** перейдет в активное состояние и ей можно будет воспользоваться. Состояние окна архивирования после выбора носителя изображено на рисунке 101.

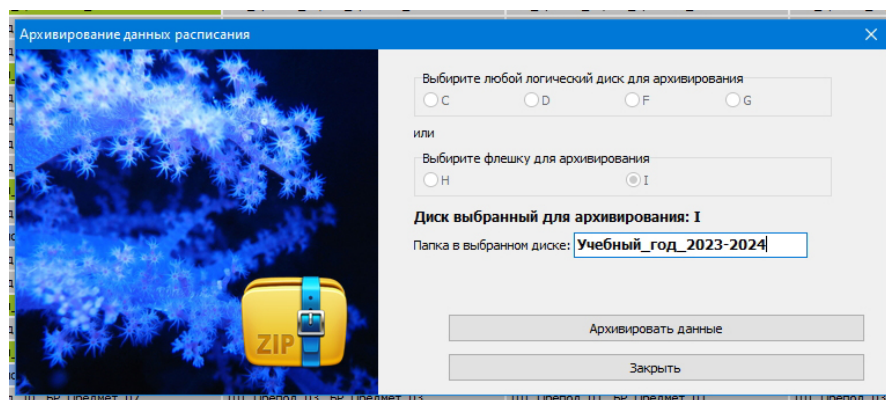


Рис. 101: Окно Архивирование данных расписания

Как видно из этого рисунка, на нем отражен отдельной строкой не только выбранный носитель информации (в данном случае флешка I), но и поле ввода папки (каталога) в которую может быть помещен файл архива. Если поле ввода для папки оставить не заполненным, то файл с архивом будет помещен в корень каталога. При этом следует меть ввиду, что операционные системы Windows версий старше чем 7 запрещают копирование файлов в корневой каталог диска C:. Таким образом, если у пользователя все же возникнет желание разместить архив именно на этом диске, то ему придется создавать папку в обязательной порядке.

Кроме того, помещение архива в папку может преследовать и другие цели. В качестве примера на рисунке указана папка: **Учебный_год_2023-2024**. Ее имя (название) содержит некую дополнительную информацию о содержании архива, что согласится достаточно удобно.

По окончании архивирования в корне выбранного носителя или в соответствующей папке появится файл **AFM_annushka_025_XX.zip** в котором и будут содержаться все необходимые данные для восстановления расписания. Точнее — для восстановления данных, хранящихся на ПК пользователя. Окончание — **_XX** соответствует третьей серии цифр из версии программы «АФМ: Аннушка».

По мнению разработчиков программы у возможности архивирования данных есть еще одно полезное назначение — это обмен данными между пользователями программы. Один пользователь может прикрепить полученный файл к письму электронной почты, а другой пользователь, получив такой файл имеет возможность его использовать в качестве восстановленных данных. Как говориться — простенько и со вкусом.

18.8.2 Пункт главного меню Восстановление расписания

В случае необходимости восстановления данных программы, пользователь после нажатия клавиши **F3** или соответствующего выбора пункта главного меню, увидит на экране стандартное окно выбора файла, изображенное на рисунке 102.

Поскольку окно стандартное мы здесь не делаем каких-либо комментариев по поводу работы с этим окном. Просто, с его помощью нужно найти ранее сохраненный файл на

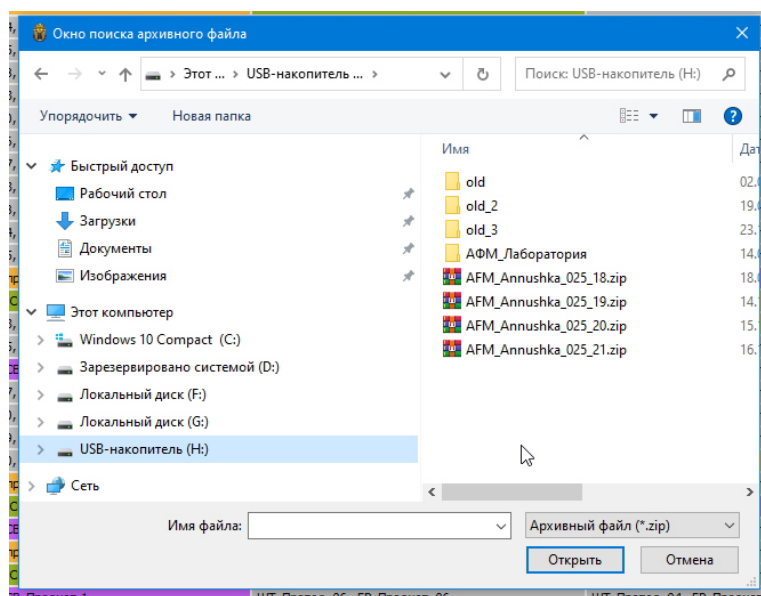


Рис. 102: Окно поиска архивного файла

некотором носителе информации, который скорее всего будет являться флешкой. После выбора в этом окне архивного файла появится окно, изображенное на рисунке 103.

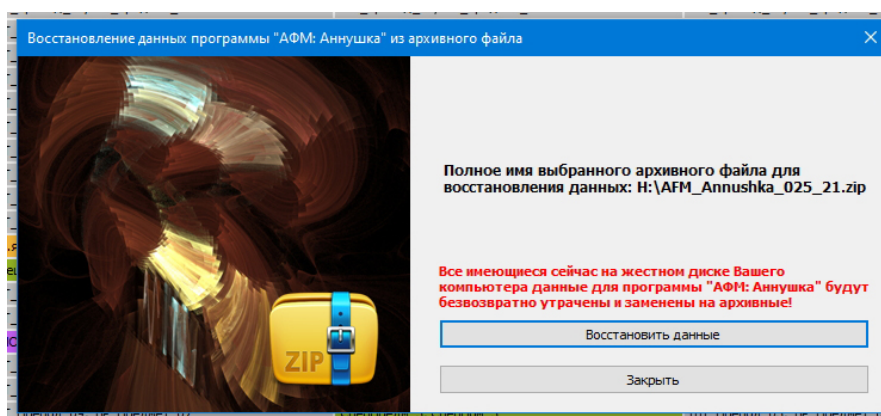


Рис. 103: Окно Восстановление данных программы

Далее пользователю останется только нажать кнопку **Восстановить данные**, немного перед этим подумав — все ли правильно он делает, и дожидаться сообщения в этом окне об успешном их восстановлении.

Проницательный и внимательный читатель уже должно быть обратил внимание на то, что на рисунке 102 можно заметить несколько файлов с разным окончанием последних двух цифр. Это архивы от предыдущих версий программы. Как нетрудно догадаться выбирать нужно именно тот файл который соответствует версии программы установленной на ПК пользователя. Данные хранящиеся на ПК пользователя от разных версий программы «АФМ: Аннушка» не совместимы друг с другом. Именно поэтому если пользователь попытается открыть не тот файл, быть ему грязно обруганным программой.

19 Расчет расписания для школ, работающих в две смены

В разделе 6.3 было отмечено, что программу «АФМ: Аннушка» можно обмануть с целью расчета (составления) расписания для школ, работающих в две смены. Но прежде чем подробно объяснить на чем основывается такой обман и как именно обмануть бестолковую программу отметим, что «Древнеегипетский» алгоритм расчета школьного расписания предназначен для составления расписаний только в одну смену.

Не будем долго останавливаться на вопросе о том, что одно дело алгоритм и совсем другое дело его реализация, т.е. программа.

Так, на чем же основывается обман программы «АФМ: Аннушка»?

А основывается он, на тупости и бестолковости этой программы.

Если бы программа могла отвечать на вопросы и у нее спросили: — «Ты вообще для какой смены составляешь расписание»? Ответ был бы простой: — «Я не знаю ...».

Действительно:

- Во сколько начинается первый урок?
- Когда заканчивается последний урок?
- Сколько минут длится урок?
- Какова продолжительность перемен?
- Когда начинается и когда заканчивается в школе обед, между какими уроками?

Ничего этого программа «не знает». Такую просто грех не обмануть.

В связи с этим возникает незатейливая мысль. Что если одно (отдельное) расписание составить для первой смены, а второе (отдельное) для второй. На первый взгляд ничего этому не препятствует. Коллектив учителей разделяется на три группы:

1. Учителя, работающие только в первую смену;
2. Учителя, работающие только во вторую смену;
3. Учителя, работающие и в первую и во вторую смену.

С первой и второй группой все ясно. Эти группы ничем не отличаются от стандарта (школ работающих в одну смену). С третьей группой могут возникнуть некоторые проблемы. Пользователю нужно будет внимательно, причем самостоятельно, отследить нагрузку учителей в первой смене (в первом расписании) и нагрузку во второй (втором расписании), что бы не возникло ситуации когда такому учителю придется работать по 48 часов в неделю. Программа ничего не подозревая о составлении расписания для двух смен отследить данную ситуацию не сможет.

Теперь покажем технологию обмана во всех деталях на конкретном примере. Хотя прежде отметим, что программа хоть и тупая но, как говорится: дура душой, а мыло не ест. И поэтому программа по мере своих сил и возможностей будет брыкаться, кусаться и сопротивляться ее обману. Все подробности на этот счет мы здесь опускаем.

Итак, возьмем к примеру сетку расписания шесть на шесть. Пусть пользователю захотелось отправить во вторую смену седьмые, шестые и пятые классы. При этом, седьмых — один класс, шестых — четыре класса, пятых — четыре класса. Всего девять классов.

После того как пользователь в соответствующем окне выберет сетку расписания, введет количество классов равное девяти и нажмет кнопку **Продолжить** появится окошко, изображенное на рисунке 104. Ранее мы это окошко уже показывали.

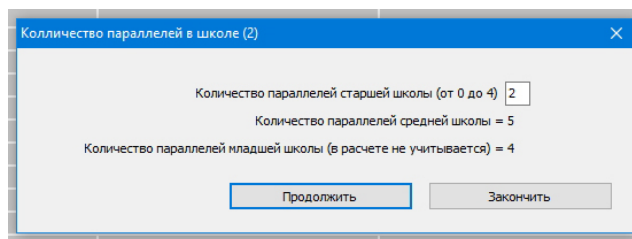


Рис. 104: Окошко Количество параллелей в школе

И оно может довести пользователя до тяжелого ступора. Самого старшего класса — седьмого, нет. Нужного количества параллелей — трех, нет. Ну да ладно, для нашего обмана заменим настоящее количество параллелей равное трем, фиктивным количеством.

Скептически настроенный пользователь сразу же спросит: — «Фиктивное количество, это сколько»?

Правильным ответом на этот вопрос будет следующий:

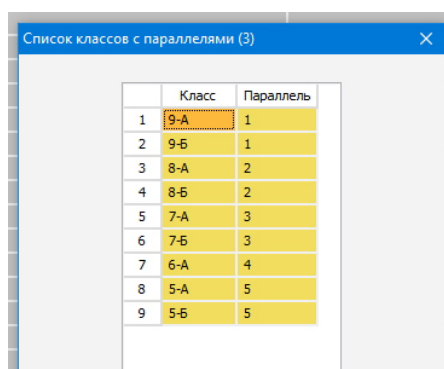
Любое количество, которое примет программа, не превышающее количества классов для которого рассчитывается расписание.

Для приводимого примера расклад такой. Если мы в поле **Количество параллелей старшей школы** введем число 0, то программа будет считать, что количество параллелей 5. Если в это поле мы введем число 4, то программа будет считать, что количество параллелей 9.

Отсюда следует, что все числа от 5 до 9, которые программа принимает, к тому же, все эти числа не превышают количество классов для которых мы в данном примере составляем расписание, нам могут подойти.

Для определенности, в качестве примера, остановимся на пяти фиктивных параллелях. В поле **Количество параллелей старшей школы** введем число 0.

Далее жмем кнопку **Продолжить** и начнем обманывать программу в следующем окне, изображенном на рисунке 105.



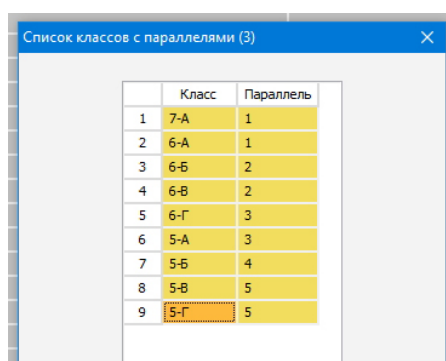
	Класс	Параллель
1	9-А	1
2	9-Б	1
3	8-А	2
4	8-Б	2
5	7-А	3
6	7-Б	3
7	6-А	4
8	5-А	5
9	5-Б	5

Рис. 105: Окно Список классов с параллелями

Программа тупо формирует в этом окне тестовые данные по своему собственному разумению. Тем не менее, не смотря на свою тупость, формирует такие данные правильно. В частности, правильно указывает номера параллелей.

Для начала вспомним, что значения в колонке **Класс** представляют из себя обычный текст, который пользователь может легко исправить. Что мы и сделаем. На рисунке 106

изображено то же самое окно после внесенных в нем исправлений.



	Класс	Параллель
1	7-А	1
2	6-А	1
3	6-Б	2
4	6-В	2
5	6-Г	3
6	5-А	3
7	5-Б	4
8	5-В	5
9	5-Г	5

Рис. 106: Окно Список классов с параллелями после исправлений

Глядя на рисунок 106 нетрудно догадаться о том, что мы в колонку Класс занесли (ввели) все названия классов для которых собрались составить расписание.

Теперь обратим свой взор на колонку Параллель. В этой колонке остались правильные параллели. Но нам для обмана программы нужны не правильные, а фиктивные параллели. Другими словами параллели — неправильные.

Вспоминаем, что программа формирует колонку «правильных» параллелей исходя из четырех законов:

1. В самой верхней строчке должна стоять единица;
2. В самой нижней строчке должна стоять пятерка;
3. После (ниже) любой строчки должна стоять цифра либо та же, что в верхней строчке, либо цифра увеличенная на единицу;
4. В колонке должны присутствовать все цифры (для нашего примера) от 1 до 5.

Для обмана программы мы дополним эти четыре закона пятым:

У классов с разной по санитарным нормам учебной нагрузкой номера параллелей должны быть разные.

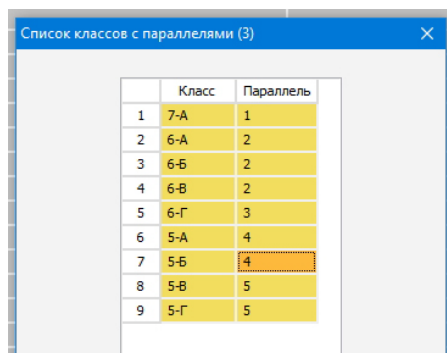
Что же мы видим на рисунке 106? У 7-А класса с учебной нагрузкой 35 часов в неделю номер параллели 1, и у 6-А класса с учебной нагрузкой 33 часа в неделю номер параллели то же 1. Непорядок. Параллель для 6-А придется исправить с 1 на 2.

Смотрим дальше. В пятой строчке у 6-Г с учебной нагрузкой 33 часа в неделю номер параллели 3, и у 5-А с учебной нагрузкой 32 часа номер параллели 3. То же непорядок. Придется исправить номер параллели для 5-А с цифры 3 на цифру 4.

После внесенных исправлений наша таблица приобретет вид, изображенный на рисунке 107.

И чего же мы этим добились? А добились мы вот чего. Седьмому классу мы поставили с соответствие 1-ю фиктивную параллель, всем шестым классам поставили в соответствие фиктивные параллели со 2-й по 3-ю, всем пятым классам поставили в соответствие фиктивные параллели с 4-й по 5-ю. И программа это запомнит после того как мы нажмем кнопку Продолжить.

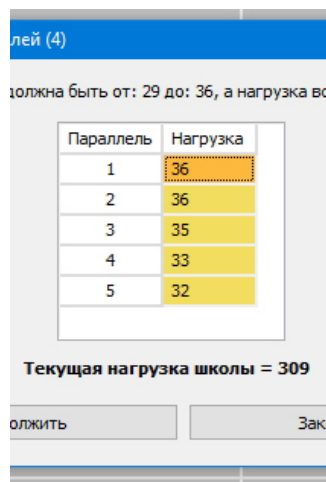
В появившемся следующим окне Учебная нагрузка параллелей, часть которого изображена на рисунке 108, программа продолжает тупить сопоставляя, как она «думает»



	Класс	Параллель
1	7-А	1
2	6-А	2
3	6-Б	2
4	6-В	2
5	6-Г	3
6	5-А	4
7	5-Б	4
8	5-В	5
9	5-Г	5

Рис. 107: Окно Список классов с параллелями после исправлений

правильным параллелям правильные с ее точки зрения учебные нагрузки классов в соответствии с санитарными нормами.



Параллель	Нагрузка
1	36
2	36
3	35
4	33
5	32

Текущая нагрузка школы = 309

Продолжить Закрыть

Рис. 108: Часть окна Учебная нагрузка параллелей

Наша задача заключается в сопоставлении фиктивным параллелям, правильных с нашей точки зрения, учебных нагрузок для классов. Таким образом для 1-й параллели (седьмой класс) мы сопоставим учебную нагрузку равную 35, для 2-й по 3-ю параллели (шестые классы) сопоставим учебную нагрузку равную 33, для 4-й по 5-ю параллели (пятые классы) сопоставим учебную нагрузку равную 32.

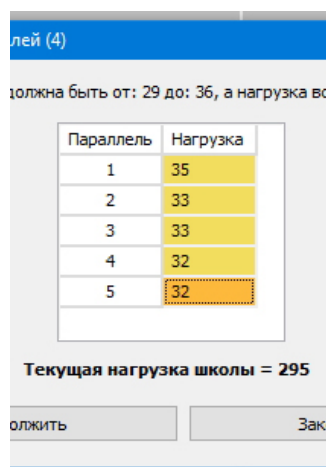
Исправленная таблица изображена на рисунке 109.

Теперь жмем кнопку **Продолжить** и как говориться — далее по списку, со всеми остановками.

Программа вела себя смирно, тихо, не брыкалась, не ругалась. Просто паенька.

Вот так мы и обманули бестолковую программу при помощи сопоставления классам для которых составляется расписание правильных (нужных нам) учебных нагрузок с помощью введения фиктивных параллелей.

Конечно, справедливости ради, нужно сказать, что такой обман не самое лучшее средство для составления расписания при двух сменах работы школы. В результате для того что бы поработать с другим расписанием нужно будет предварительно сохранить на флешку текущее расписание, а потом что бы к нему вернуться нужно будет восстанав-



Параллель	Нагрузка
1	35
2	33
3	33
4	32
5	32

Текущая нагрузка школы = 295

Очистить Закрыть

Рис. 109: Часть окна Учебная нагрузка параллелей после исправлений

ливать это расписание с той же флешки. В результате чего можно легко запутаться.

Но что делать. По видимому, в древнем Египте не было школ, обучающих будущих фараонов, жрецов и придворных мальков в две смены. Из-за этого и не позаботилась жрица Анушер предусмотреть в своем «Древнеегипетском» алгоритме составления школьного расписания две смены.

Нам остается пользоваться только тем, что есть, не взирая на трудности, прочие неудобства, непристойности и излишества.

20 Заключение

То, что изготовление (написание) текстов является очень непростым занятием (искусством) было известно давно. Более того, по нашему мнению это занятие является более затейливым, чем программирование. При составлении данного текста в голове все время крутились такие слова, как:

- Вычесывание;
- Вылизывание;
- Вырубание;
- Выскребание;
- Выделывание.

А так же:

- Шлифовка;
- Рихтовка;
- Заделка;
- Обрезка;
- Пристройка;
- Обточка;
- Подчистка;
- Строгать;
- Тесать;
- Утюжить;
- Пользовательский интерфейс;

- Повторное тестирование;
- Переделка.

Авторы этого текста далеки от мысли, что ими был произведен идеальный продукт (как сам предлагаемый текст, так и собственно программа). Мысль о том, что совершенству предела нет, для авторов проста и понятна, как дважды два — четыре.

Основные принципы, которых пытались придерживаться авторы при подготовки текста это:

- Понятность (и отсутствие двусмысленности);
- Подробность;
- Полнота.

Нет сомнений в том, что достичь своей цели им так и не удалось. Поэтому мы с удовольствием примем все замечания заинтересованных читателей, которые пришлют нам свои вопросы, пожелания, комментарии, примеры, которые вызывают у них затруднения и которые не рассматривались здесь.

Написание (разработка) данного повествования носит ярко выраженный итерационный характер. Не редкость, когда работа начиналась не сначала, а с конца или середины. Все время приходилось восклицать: — «Как же мы забыли об этом и пропустили!». В следующей редакции этого текста, не рассмотренные вопросы будут непременно рассмотрены. Непонятные места будут разъяснены. Понятия описанные не подробно, будут описаны подробно. Все ошибки и несуразицы пользовательского интерфейса устранены.

Ждем Ваших отзывов! Как о данном тексте (включая о замеченных в нем опечатках), так и о смой программе (особенно о замеченных в ней ошибках или неточностях). И удачи в работе при тестировании «Древнеегипетского» алгоритма составления школьных расписаний.

Список литературы

- [1] Амосов А.А., Дубинский Ю.А., Копченкова Н.В. Вычислительные методы для инженеров: Учебное пособие — 2-е изд. доп. — М: Издательство МЭИ. 2003. — 596 с.
- [2] Балтак С.В., Сотсков Ю.Н. Построение расписания учебных занятий на основе раскраски вешин графа // Информатика, 2006, № 3, с. 58 – 69.
- [3] Воронина Е.В. Профильное обучение: модели организации, управленческое и методическое сопровождение. — М.: «5 за знания», 2006. — 256 с.
- [4] Галкина Т.И., Сухенко Н.В. Организация профильного обучения в школе. — Ростов н/Д: Феникс, 2007. — 288 с.
- [5] Гафаров Е.Р., Лазарев А.А. Математические методы оптимизации при составлении учебного расписания // Новые информационные технологии в образовании. Сборник научных трудов. — М.: 1С-Пабблишинг, 2013, Часть 2, с. 51 – 55.
- [6] Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. — М.: Мир, 1982. — 416 с.

- [7] Ичбана Д., Кнеппер С. Бил Гейтс и сотворение Microsoft. — Ростов-на-Дону: Издательство «Феникс», 1997. — 352 с.
- [8] Костюкова Н.И. Графы и их применение. Комбинаторные алгоритмы для программистов: Учебное пособие — М: Интернет-Университет Информационных Технологий; БИНОМ Лаборатория знаний, 2007. — 311 с.
- [9] Математика: Школьный курс. — М.: АСТ-ПРЕСС, 2001. — 608 с.
- [10] Романовский И.В. Дискретный анализ. Учебное пособие для студентов, специализирующихся по прикладной математике и информатике. — Издание 2-е, исправленное. — СПб.: Невский диалект, 2000. — 240 с.
- [11] Свами М., Тхуласираман К. Графы, сети и алгоритмы. — М.: УЦ Мир, 1984. — 455 с.
- [12] Сергеева В.П., Ляпко И.Л. Профильное обучение в общеобразовательном учреждении. — М.: УЦ Перспектива, 2011. — 164 с.
- [13] Смирнов В.В. Перербургские школы и школьные здания. История школьного строительства в Санкт-Петербурге — Петрограде — Ленинграде 1703 – 2003 гг. — СПб.: Издательство «Русско-Балтийский информационный центр "БЛИЦ"», 2003. — 144 с.
- [14] Профильное обучение: программы элективных курсов здоровьесберегающей направленности: Учебно-методическое пособие / Под ред. Т.В. Черниковой — М.: ТЦ Сфера, 2006. — 304 с.
- [15] Урнов В.А. Расписание — наиболее востребованный АРМ в образовании // Информатика и образование. 2001, № 4, с. 47 – 52.
- [16] Филипенко А.С., Овсянников К.В., Болотов С.В. Анализ требований к школьному расписанию с учетом специфики белорусского законодательства // ПФМТ. 2016, выпуск 4(29), с. 103 – 106.
- [17] Харари Ф. Теория графов. — М.: Мир, 1973. — 302 с.
- [18] Even S., Itai A., Shamir A. On the complexity of timetable and multicommodity flow problems // SIAM J: Comput. Vol. 5, No. 4, December 1976, 691-703