**ГЛАВА 1. Постановка задачи.**

**§1. Введение.**

 Новая шкала ценностных приоритетов, отражающая государственную политику и отношение педагогической науки к образованию, является на сегодняшний день главным фактором, определяющим необходимость реформирования школьной системы образования и перехода к 12-летней школе. Ожидаемые в связи с этим преобразования носят достаточно существенный характер, поскольку предполагают «осуществление принципиально другой направленности образования, связанной не с подготовкой «обезличенных» квалифицированных кадров, а с общим, социально-нравственным и профессиональным развитием личности».

Радикальность предстоящих перемен, в процессе которых во главу угла предполагается поставить создание условий для максимально полной самореализации каждого учащегося и свободного развития его личности, делает весьма актуальным вопрос о порядке реформирования традиционной системы образования, базирующейся в основном на «знаниевой» парадигме. Совершенно очевидно, что режим «шоковой терапии» в данном случае абсолютно неуместен.

Единственно верным в создавшейся ситуации представляется путь последовательного и щадящего реформирования, предполагающий не безоглядную ломку сложившейся системы образования, а ее приспособление к решению новых задач с сохранением всего ценного, что она накопила. При таком подходе большую значимость приобретает проблема педагогического моделирования, результаты которого могут служить аргументированным основанием как для сохранения накопленного потенциала традиционной системы образования, так и для выбора форм и методов ее реформирования. Особый интерес в этой связи приобретают случаи, когда педагогическое моделирование ведется в количественном виде и сопровождается установлением функциональных и корреляционных соотношений, связывающих конечные педагогические показатели с параметрами образовательного процесса и исходными характеристиками учебно-воспитательного коллектива. Именно они способны обеспечить доказательность и оптимальность выбираемого пути реформирования педагогического процесса и его приспособления к решению изменившихся задач.

В настоящей работе приводятся результаты исследований, посвященных проблеме педагогического моделирования интеллектуального испытания школьников. В арсенале педагогических методов и средств интеллектуальному испытанию принадлежит одно из важнейших мест. В режиме интеллектуального испытания, например, проходит большинство способов контроля уровня знаний учащихся (опросы, контрольные и самостоятельные работы, экзамены, тесты). Интеллектуальное испытание лежит в основе мероприятий соревновательного характера олимпиад, викторин, конкурсов. Без интеллекту­ального испытания учащихся невозможно представить себе не только проблем­ное, но и традиционное обучение, поскольку сам процесс обучения, если го­ворить по большому счету, ведется в форме распределенного во времени ин­теллектуального испытания учащихся. При этом студенты, не выдерживающие этого испытания, просто отчисляются из вуза, а школьники переводятся на более щадящий режим обучения, например, в классы коррекции.

Очевиден и воспитательный аспект интеллектуального испытания, кото­рое можно рассматривать как определенную форму воздействия на испытуе­мого школьника. Тот факт, что режим этого воздействия задается непосред­ственно педагогом, превращает интеллектуальное испытание в инструмент формирования личности учащегося, его характера, способности к самоорга­низации и концентрации усилий на преодоление трудностей. С этой точки зрения, интеллектуальное испытание являет собой пример управляемого тре­нинга, подготовки школьника к будущей «взрослой» жизни, представляющей собой, как известно, бесконечную цепь весьма непростых испытаний.

Выбор олимпиады школьников в качестве предметной базы для отработки педагогической модели интеллектуального испытания обусловлен целым ря­дом обстоятельств. Здесь, в первую очередь, следует отметить простоту и про­зрачность олимпиады как педагогического мероприятия с четко определен­ным регламентом, в рамках которого многие педагогические проблемы при­обретают смысл, доступный для описания на языке количественных соотно­шений. Вторым обстоятельством, выделяющим олимпиаду в качестве опти­мального объекта педагогических исследований, является уникальность ан­самбля ее участников, представляющего простейшую педагогическую систе­му, образованную «механическим» соединением школьников. Данная систе­ма действительно уникальна. Она характеризуется заведомой аддитивностью своих свойств и соответствует наиболее простой (если не сказать самой примитивной) форме взаимоотношения личности и коллектива, выражающейся в элементарном сложении.

Простота олимпиады заключается еще и в небольшом разбросе ее участ­ников по уровням подготовки (все они в большинстве своем хорошо успева­ющие школьники). Это создает условия для использования линейных при­ближений, что значительно упрощает математическое описание. Моделиро­вание итогов олимпиады облегчается тем, что распределение участников по способностям известно априори. В силу многоэтапного характера олим­пиады оно соответствует распределению отобранного ансамбля, в котором основную массу испытуемых составляют именно «способные» учащиеся, поскольку малая доля «истинно талантливых» школьников определяется чис­то объективными причинами, а незначительное представительство в ансамб­ле «откровенно слабых» учащихся - их отсевом на предыдущих этапах.

Олимпиада школьников в дополнение ко всему является чрезвычайно удоб­ным объектом не только для теоретических, но и для экспериментальных пе­дагогических исследований. По отношению к проблеме интеллектуального испытания она является готовым экспериментальным полигоном. С одной стороны, циклический характер олимпиады и практически неизменный поря­док ее проведения обеспечивают благоприятные условия для долговременно­го констатирующего эксперимента по изучению параметров интеллектуаль­ного испытания, необходимых при формулировке исходных позиций модели­рования. С другой стороны, автономия отдельных этапов олимпиады предос­тавляет составителям заданий и организаторам олимпиад достаточно широ­кие возможности для формирующего этапа эксперимента, связанного с апро­бацией модели и внедрением модельных разработок в практику проведения олимпиад. Многоуровневая структура олимпиады в сочетании с иерархичес­кой взаимосвязью отдельных этапов обеспечивает при этом широкомасштаб­ный характер исследований как на пассивной, так и на активной стадиях экс­перимента. Она позволяет работать с большими статистическими ансамбля­ми, представляющими в то же самое время соединение весьма разнообраз­ных выборок учащихся. Это обеспечивает необходимую репрезентативность и достоверность получаемых экспериментальных результатов.

Непосредственную опытную базу настоящего исследования составили региональные физические олимпиады школьников, проходившие в Рязани в 2003 г., а также ведомости успеваемости студентов физико-математического факультета по разным предметам. Это дало возможность судить о гуманности преподавания на тех или иных кафедрах Рязанского педагогического университета им. С. А. Есенина. Кроме того, в настоящем исследовании были использованы материалы, взятые во время прохождения педагогической практики в средней школе №43 г. Рязани.

**§2. Цель работы.**

Работа полностью опирается на теоретические исследования Б. С. Кирьякова, и была призвана дополнить их. С самого начала передо мной ставилась задача превратить эти исследования, а также накопленную в них математическую базу, в нечто осязаемое, то есть попросту упростить тот процесс обработки экспериментальных результатов, который предлагает сам автор теории. Таким образом, целью данной работы можно считать разработку автоматизированной системы распределения мест и оценки уровня качества олимпиадных задач по физике. При выполнении работы, мною была разработана специальная программа, которая инкапсулирует в себе ту математическую теорию, которую разработал Б. С. Кирьяков. Совместно с ним была произведена проверка данной программы на примере городской олимпиады по физике в 11 классах. Кроме этого, в качестве эксперимента, через программу «прогнали» и ведомости студентов физмата по некоторым дисциплинам. При этом были получены очень интересные результаты, о которых речь пойдет ниже.

 Вообще говоря, разработанная программа может оказаться полезной не только на олимпиадах. Она может помочь и на простых уроках, причем по любым предметам.

Математическая теория, лежащая в основе программы, оперирует достаточно простыми понятиями, и, в принципе, может быть понятна рядовому учителю. Однако необходимости в изучении азов нет, так как не каждому педагогу интересна начинка какого-либо сложного с первого взгляда объекта, а большую важность здесь имеет результат. Собственно говоря, программа и призвана для получения конкретного результата без акцентирования на деталях расчета, а если этот результат представлен визуально, то это дополнительный плюс всей системе.

**Глава 2. Проблема распределения мест на олимпиаде и ее решение. Оценка уровня качества олимпиадных заданий.**

**§1. Теория распределения мест. Проблема дифференцированного подхода.**

 Проблема автоматизированного распределения мест на олимпиадах не нова. Существуют определенные системы распределения мест во многих странах мира (например, в США), и все они имеют ряд очевидных преимуществ по сравнению со стандартной схемой.

Первое (и самое главное) преимущество – отсутствие «человеческого фактора» при этой процедуре. Машине чужды эмоции, она бесстрастна, а что еще нужно для грамотной постановки вопроса. К тому же, в связи с широким, в последние 5 лет, распространением компьютерной техники в России, разработка таких систем является достаточно перспективной областью.

Второе преимущество – это так называемый «фактор времени». Всем известно, что любая школьная (городская, областная и т.д.) олимпиада – это дело долгое. Сначала участники выполняют задания, потом жюри оценивает их, а далее следует процесс сортировки работ по местам, причем, чем больше участников на олимпиаде, тем больше времени этот процесс занимает. В школе это время небольшое, но в масштабах области или страны это может занять очень много времени. Машина же выполняет этот процесс гораздо быстрее, и время на сортировку можно сократить на порядок, а то и два.

Скажем сразу – полностью автоматизированной системы для проведения олимпиад, их оценки, распределения мест нет, хотя проекты такие существуют. Машина пока может лишь работать с данными, которые в нее вводит человек. В будущем, возможно, будут созданы системы, которые сами будут проверять задания, оценивать их, распределять места и т.д., а человек будет лишь контролировать эту деятельность и пожинать ее плоды.

Вот к чему на данном этапе все стремятся, однако это не так просто как кажется. Поэтому мы остановились на обычной системе, работающей с протоколом, который вводится оператором. Исходя из данных, которые содержатся в этом протоколе, программа получает конечный результат и визуализирует его, но об этом ниже.

Теперь немного теории.

Распределение участников олимпиады по занимаемым местам происхо­дит на заключительной стадии олимпиады. Именно здесь определяются при­зеры, представляемые к награждению, и участники, допускаемые к выходу на следующий этап олимпиады. Отвечает за распределение мест обычно пред­седатель предметного жюри.

Фактическую базу, определяющую распределение мест, образуют итоги олимпиады, отражающие успехи школьников в решении олимпиадных задач. Обычно их представляют в виде (1):

***x1, x2, x3, …,xi, …, xn,*** (1)

где xi = 0, 1, 2, …, ***m*** – баллы, набранные участником за задачу с номером ***i***.

Распределение мест непосредственно проводят не по итогам решения от­дельных задач (1), а по некоторым показателям **ή**1, **ή**2, **ή**3, ..., характеризу­ющим выполнение олимпиадного задания в целом:

(**ή**1, **ή**2, **ή**3, ...)=║**П**║(**x**1**, x**2**, x**3**, …**) (2)

где ║**П**║ − некоторые преобразования, переводящие описание итогов олимпиа­ды с языка переменных ***х****1,****х****2,****х****3,…* (равных набранным баллам за отдельно взятые задачи), на язык показателей **ή**1, **ή**2, **ή**3, ..., характеризующих выпол­нение всего олимпиадного задания.

Показатели **ή**1, **ή**2, **ή**3, ..., определяющие распределение мест, удобно называть *показателями приоритета.* Одним из таких показателей, как изве­стно, является суммарный балл:

***S****=****х****1+****х****2+****х****3 + ... +* ***х****i+... +* ***х****n* (3)

В общем, порядок распределения участников соревнования по мес­там при множественном числе показателей приоритета определяется выбо­ром самих показателей **ή**1, **ή**2, **ή**3, ..., их числом ***l*** и логикой приоритета, определяющей место участника олимпиады в соответствии с численными значениями показателей **ή**1, **ή**2, **ή**3, ... . С формальной стороны использова­ние нескольких показателей при выстраивании какой-либо одномерной оче­редности объектов не создает больших сложностей. Для этого достаточно один показателей считать «главным», второй − «второстепенным», третий − «третьестепенным» и т.д. При распределении мест главный показатель **ή**1 следует принимать во внимание в первую очередь, второстепенный **ή**2 при равенстве главных, а третьестепенный **ή**3 при одновременном равенстве главных и второстепенных показателей и т.д.

Подобное распределение очень часто используется в спорте. Примером того может служить распределение футбольных команд по итогам чемпионата, которое проводят по двум показателям − по числу набранных очков (главный показатель) и по разнице между забитыми и пропущенными мячами (второстепенный показатель).

Однако это только формальная сторона дела. Вся сложность проблемы заключается в том, что ввести отмеченную иерархию показателей приоритета («главный», «второстепенный» и т.д.) достаточно непросто. Особенность ситуации состоит в том, что формальная логика распределения мест при множе­ственном числе показателей

l≥2(4)

оказывается внутренне противоречивой. Данное противоречие кроется в равноправной возможности двух подходов к распределению мест между участниками олимпиады − одного с ориентацией на большее удаление от «абсо­лютного аутсайдера» (участника, не набравшего ни одного балла), другого с ориентацией на наибольшее приближение к «абсолютному лидеру» (участни­ку, давшему исчерпывающее решение всех задач),

Отмеченное противоречие не имеет места при одном показателе приори­тета **ή**1. В этом случае каждый участник, набирая баллы по задачам и удаляясь от аутсайдера, неминуемо приближается к лидеру.

Подобная однозначность, как это ни странно, не является достоинством. Достаточно вспомнить, что распределению подвергаются не абстрактныеобъекты, а школьники. Распределение по местам подростков и юношей, отя­гощенных комплексом проблем своего возраста, можно проводить лишь с учетом соображений психолого-педагогического характера, которые по сво­ей сути являются вариативными, зависящими от конкретной ситуации. При одном показателе приоритета условий для подобной вариативности, а соот­ветственно и для дифференцированного подхода нет. Все однозначно опреде­ляется формальной логикой, а соображения психолого-педагогического ха­рактера просто некуда включить.

 Однако руководствоваться соображениями только формальной логики нельзя. Данная ситуация представляется чрезвычайно интересной. Ее уникальность заключается в том, что она соответствует условиям, когда необходимо привлечение педагогических соображений к распределению мест. Понятна и роль, отводимая при этом педагогике. Это роль «третейского суда», который в рамках сложившегося противоречия может стать на одну из двух взаимоисключающих точек зре­ния, руководствуясь соображениями педагогической целесообразности.

Ситуация соответствует случаю, когда возможный порядок распределения мест таков, что приоритет численных значений пока­зателя **ή**1, определяется формальной логикой, а приоритет значений показате­ля **ή**2 − педагогической целесообразностью. В силу вариативного характера педагогических соображений данное распределение можно провести диффе­ренцированно, меняя точку зрения на приоритет значений **ή**2 по отношению к каким-то выделенным группам школьников.

Отмеченные «взаимоотношения» показателей **ή**1 и **ή**2говорят о логическом главенстве **ή**1*.* При распределении мест его необходимо рассматривать в качестве главного показателя и принимать во внимание в первую очередь, а показатель **ή**2 − в качестве второстепенного и учитывать лишь при равенстве значений **ή**1.

Приведенные выше соображения говорят о том, что дифференцирован­ный подход к участникам олимпиады в рамках ее регламента вполне возмо­жен. Он может быть реализован лишь на стадии распределения мест, но толь­ко в том случае, когда оно проводится по нескольким показателям приоритета (4). Одного главного показателя **ή**1, определяющего приоритет выполнен­ного задания с позиций формальной логики, для этого недостаточно. Педаго­гические соображения, обеспечивающие дифференцированный характер рас­пределения мест, могут быть учтены лишь с помощью второго, третьего и других показателей более высокой степени.

Смысл главного показателя приоритета **ή**1 вполне ясен. Суммарный балл (3) способен испол­нять роль лишь главного показателя приоритета **ή**1, и в принципе не может служить предметной базой для дифференцированного подхода.

Возможность использования величины **ή**2= ***x***1−***x***2 (5) в качестве второстепенного показателя приоритета, дополняющего суммарный балл **ή**1 (4), достаточно очевидна. Если суммарный балл **ή**1 определяет выполнение задания с количественной стороны, то показатель **ή**2 (5) характеризует качество выполнения задания. Он показывает, в решении какой из задач (простой или сложной) участник больше преуспел.

Множественный характер показателей приоритета является свидетельством самой возможности дифференцированного подхода. С этой точки зрения соотношение (4) можно рассматривать как необходимое условие, определяющее соответствие используемой системы распределения мест требованиям дифференцированного подхода. Следует отметить, что в условиях рязанских региональных олимпиад условие (4) никогда не выполнялось. Места тради­ционно распределялись с использованием лишь одного показателя приорите­та - суммарного балла ***S*** (3), что не дает никаких оснований даже говорить о дифференцированном подходе.

В общепедагогическом плане пренебрежение дифференцированным подходом может вызывать лишь глубокое сожаление. Олимпиада, являясь педа­гогическим мероприятием, должна заниматься не только констатацией спо­собностей участников на момент ее проведения, но и заботиться о создании мотивационной базы для развития скрытых потенциальных возможностей учащихся. В первую очередь, здесь следует обращать внимание на участников, которые выступили на олимпиаде пока еще не совсем удачно. Этих школьни­ков необходимо поддержать и отметить хотя бы самые малые их успехи на олимпиаде, подкрепив все соответствующим поощрением по соображениям педагогического характера. Дифференцированный подход к распределению мест, возможный при выполнении соотношения (4), создает для этого все необходимые условия.

Следует отметить, что введение множественного числа показателей при­оритета, определяющих саму возможность дифференцированного подхода, не может быть произвольным. Для этого необходимы различаемые этапы ре­шения задач или различаемые задачи (что несколько предпочтительнее). Имен­но по этой причине для олимпиады должны быть использованы разноуровневые задачи (2). Только различие этих за­дач сделало понятным смысл **ή**2 (5) как показателя поляризации способ­ностей школьника. Для одноуровневых неразличимых задач показатель **ή**2 (в отличие от **ή**1, характеризующий выполнение задания с количественной стороны) потерял бы всякий смысл, что сделало бы невозможным его использование как показателя приоритета.

В нашем случае мы ограничиваемся лишь тремя показателями приоритета **ή**1, **ή**2 и **ή**3 при распределении мест, чего вполне достаточно для нашей задачи. Смысл этих показателей достаточно прозрачен. Показатель **ή**1, как показано выше, тождественен суммарному баллу и сам по себе не может быть использован в качестве критерия для распределения мест. Показатель **ή**2 характеризует успехи школьника в репродуктивно-продуктивной деятельности по сравнению со средним арифметическим значением его успе­хов за отдельно взятые испытания репродуктивного и продуктивного харак­тера. Он показывает, насколько соединение способностей школьника отлича­ется от их простого арифметического сложения. Показатель же **ή**3 характеризует поляризацию способностей школьника, представляя его достижения в решении творческих задач, рассчитанных на продуктивную деятельность, в сравнении с успехами в решении типовых задач, носящих репродуктивный характер. Все три показателя являются целыми числами, что существенно облегчает процесс расчета.

 Таким образом, имея результаты олимпиады (или, например, сессии), можно точно подсчитать эти три показателя, исходя из них, можно с большой точностью говорить о распределении мест. Здесь возникает еще один вопрос: какой из показателей главный, а какие второстепенный и третьестепенный? Частично эта проблема решена выше, но там описывались только два параметра. Решение здесь может быть таким. Необходимо вводить несколько «дифференцированных подходов» на базе значений показателя **ή**1 (так как он является основным и главным для других). Если значения **ή**1 для большей части (или для всех) участников отрицательны (это говорит о потенциальной слабости испытуемого коллектива), то имеет смысл за второстепенный показатель принять **ή**2, а за третьестепенный – **ή**3. Проще говоря, в этом случае мы акцентируем внимание на репродуктивные (типовые) задачи, которые, по логике вещей, участники должны решить. Продуктивные (творческие) же задачи мы как бы не учитываем вообще в силу того, что такой коллектив может их не решить вообще. Например, таким ансамблем является коллектив школьников, представленный в программе в базе dbolymp1. Это условно первый вариант дифференцированного подхода.

 Возможен вариант, что значения **ή**1 для всех участников только равны нулю или положительны (это признак сильного коллектива). В этом случае за второстепенный показатель приоритета имеет смысл принять **ή**3, а за третьестепенный – **ή**2. Другими словами, здесь мы делаем упор именно на продуктивные задачи (они обычно сложнее), а решение типовых задач считаем саморазумеющимся. Этот подход можно назвать вторым методом дифференцированного подхода.

 И, наконец, самый интересный случай – **ή**1 для всех участников принимает и нулевые, и положительные, и отрицательные значения. Здесь процесс распределения мест несколько усложняется, так как во всем количестве участников присутствуют и потенциально сильные ученики, и слабые. Понятно, что всех их сортировать только одним из способов нельзя (исчезает главный принцип дифференцированного подхода), поэтому мы прибегаем к комбинационному методу. Суть метода такова. Все многообразие участников делится пополам, исходя из значений **ή**1. Тех участников, у которых **ή**1≥0, относят к условно «сильной» группе и для сортировки используют метод **ή**1→ **ή**3→ **ή**2. Те же участники, у которых **ή**1<0, попадают в условно «слабую» группу, и для этой группы используют метод **ή**1→ **ή**2→ **ή**3. Таким образом достигается полная реализация принципов дифференцированного подхода. Реально, олимпиадных коллективов с такой комбинацией значений параметра **ή**1, практически не встречается. Это можно отнести к минусу составителей олимпиадных заданий, а можно – к учителям, которые готовят школьников к олимпиадам. Это самый общий принцип дифференцированного подхода. Мы назовем его условно третьим методом. Этот метод, вообще говоря, применим всегда, так как видно, что он является сочетанием первых двух методов. Поэтому, всегда рекомендуется использовать именно его. В частности, разработанная система не требует вмешательства пользователя в процесс выбора типа метода, сама выбирает необходимый и сортирует, придерживаясь этого типа.

Сложно сказать, что должно быть в идеальном случае. С одной стороны, если сильных участников будет много – это хорошо. С другой стороны – можно с полной уверенностью сказать о том, что всегда будут и сильные, и слабые ученики. Единственное, о чем можно точно говорить – модель, которая использовалась при построении теории, базируется на последнем варианте распределения.

Это было краткое введение в теорию распределения мест, которая использовалась при создании автоматизированной системы. Теперь, опять же с точки зрения теории, рассмотрим проблему оценки уровня качества олимпиадных заданий, что тоже в дальнейшем понадобится.

**§2. О проблеме оценки уровня качества олимпиадных заданий.**

 Проблема оценки уровня качества олимпиадных заданий является достаточно интересной областью исследования на данном этапе. Понятно, что сейчас есть смысл говорить о качестве заданий, предлагаемых на олимпиадах по различным тематикам. Подобно тому, как любой продукт питания или элемент домашней техники должен удовлетворять каким-то определенным требованиям, олимпиадное задание должно также характеризоваться набором каких-либо параметров, которые, в свою очередь, должны характеризовать его качество и класс его составителя. Однако такие параметры для конкретного олимпиадного задания найти достаточно сложно или правильнее сказать практически невозможно. В этом случае реально можно использовать только один очевидный параметр – сложность задачи. Но, с другой стороны, одна и та же задача может быть «сложной по-разному» для разных учеников. Здесь подразумевается то, что у задачи может быть разный ход решения, приводящий к правильному результату, и этот ход по-разному воспринимается разными учениками. Проще говоря, для одного ученика данная задача окажется очень легкой, а для другого – нерешаемой, и говорить о сложности нет смысла. Однако в контексте данной теории все задачи условно делят на три группы: продуктивные (творческие), репродуктивные (типовые) и продуктивно-репродуктивные (типовые задачи с «изюминкой» или творческие с элементарным смыслом). При этом полагается, что решение продуктивной задачи вызовет у любого ученика большую сложность, чем решение репродуктивной.

 Вообще говоря, необходимо понять то, что нужно оценивать качество не какой-то отдельной олимпиадной задачи, а пытаться оценить блок заданий на олимпиаде и всю ее целиком. Эта задача менее трудна, но тут тоже может встретиться ряд трудностей, главная из которых заключается в поиске адекватных педагогической теории параметров, при помощи которых эти самые задания и оцениваются. То есть необходимо вывести такие показатели, которые будут полностью объясняемы с точки зрения педагогики. После того, как эти параметры будут известны, смысл их с педагогической точки зрения понятен, необходимо попытаться определить оптимальный вид комплектации олимпиадных заданий по типам задач, при котором будет достигнут максимальный эффект. В этом, собственно говоря, и заключается смысл всей теории.

**§3. Виды задач. Краткое описание каждого вида.**

 Как уже отмечено выше, в рамках теории существует некое деление задач по видам деятельности учащихся. Это необходимо для нашего подхода к оценке их уровня качества. Опишем каждый вид.

1. ***Продуктивные задачи.***

Данный тип задач, как видно из названия, учитывает творческую деятельность учащихся, то есть при решении таких задач необходимо провести маленькое исследование или поставить небольшой мысленный эксперимент. Это необходимо для полного и верного решения. К такому типу задач относят, например, качественные задачи. Естественно, что данный тип задач является достаточно сложным для решения, и поэтому часто используется на олимпиадах.

Рис. 1. Пример продуктивной задачи.

Рис. 2. Распределение по баллам для этой задачи.

1. ***Репродуктивные задачи.***

Этот тип задач дает возможность учитывать репродуктивную деятельность учащихся. При решении задач такого типа необходимо либо знать определенную формулу, либо вспомнить ее. По сути, данные задачи – это просто набор определенных формул, связанных общими неизвестными (найдем данную величину из этой формулы, подставим вот в эту и получим искомый результат). Такие задачи обычно очень легкие, буквально в одно действие. Из-за их простоты, достойного применения на олимпиадах они не нашли. Однако, как потом выяснится, зря. К такому типу задач можно отнести задачи учебника на повторение (особенно, 11 класс Мякишева), а также большая часть задач из сборника Рымкевича.

Рис. 3. Пример репродуктивной задачи.

Рис.4. Распределение для этой задачи.

1. ***Продуктивно-репродуктивные задачи.***

Это – самый интересный тип задач. Он представляет собой смесь первых двух видов, что делает его привлекательным для большинства составителей олимпиадных заданий. В принципе, это верно, ведь данный тип позволяет проверить знания учащихся сразу в нескольких аспектах. Задачи такого типа, очевидно, могут иметь различную структуру (см. рис. 5 и рис. 6). На рисунке ниже представлено два интересных варианта такой структуры.

Рис. 5. Две структуры продуктивно-репродуктивных задач.

Рис. 6. Распределения для этих задач.

Ход решения таких задач во многом зависит от их подвида. Например, задача типа «додуматься, а потом вспомнить» относится как раз к этому классу. Ясно, что задачи такого типа получили наибольшее распространение на олимпиадах, а также в задачниках для поступающих в ВУЗы.

**§4. Понятие о сбалансированном комплекте олимпиадных заданий. Шкала сложности.**

 Введение данного понятия необходимо по нескольким причинам: первая причина заключается в том, что для построения такой педагогической модели, которую мы используем в данной работе, необходим какой-то определенный идеализированный подход к олимпиадным заданиям. То есть нужно представить себе идеальный случай, при помощи которого можно описать (математически и, главное, педагогически) все реально встречающиеся варианты. Вторая причина состоит в том, что для построения шкалы сложности задач, нужно иметь какой-то базовый элемент, относительно которого и происходит построение этой шкалы.

 В данном параграфе описывается лишь формальное введение основного понятия данной теории. В полном описании математического вывода и доказательства педагогической оправданности сбалансированного комплекта задач нет необходимости в силу того, что сама по себе автоматизированная система не использует этого понятия, а использует только математические выводы, которые сделаны на его основе.

 Под *сбалансированным комплектом олимпиадных заданий*, в контексте данной работы, будем понимать такой комплект заданий, в котором максимально равномерно воссоединены жесткий, естественный и щадящий режимы испытания для вывода серии всех испытаний школьников на гуманное отношение к личности школьников и бережное отношение к их талантам. В рамках представлений обсуждаемой модели, исходят из двух видов учебной деятельности учащихся и объясняют разный уровень сложности задач разным насыщением их решений формальными и творческими моментами. Эти требования к комплекту тождественны требованиям сбалансированности и полноты этого комплекта по отношению к репродуктивному, продуктивно-репродуктивному и продуктивному видам деятельности учащихся.

 Хочется обратить внимание на то, что сбалансированный комплект представляет собой лишь идеализированную модель педагогического испытания школьников на олимпиадах. Ясно, что такой комплект в реальных условиях подобрать крайне сложно, однако он позволяет судить о том, какими должны быть олимпиадные задания, чтобы, в результате, можно было максимально приблизится к идеалу.

 Вопрос об уровне сложности задач носит в рамках рассматриваемой теории достаточно важный характер. Наиболее исчерпывающий ответ на него может дать *шкала сложности задач*. Основные особенности подобной шкалы непосредственно оговариваются в исходных положениях теории. В связи с этим, следует упомянуть два момента. Первый момент заключается в том, что для полного анализа задач достаточен учет *двух* различных и несводимых друг к другу видов учебно-познавательной деятельности школьника – репродуктивной и продуктивной. Второй момент изначально оговаривает большие способности каждого школьника к репродуктивному виду деятельности по сравнению с продуктивным. Этот момент условно выразим неравенством: .

 Принципиальная особенность указанных моментов заключается в том, что они определяют заведомо двумерный характер шкалы сложности задач. На этой шкале каждая задача должна характеризоваться двумя индексами, учитывающими два вида деятельности учащихся. В соответствии с этим любой единый показатель уровня сложности задач должен быть двумерным объектом. Это касается всех возможных шкал, включая и простейший случай ранжированной шкалы, оперирующей лишь целочисленной нумерацией уровней сложности задач. Она должна быть также двойной. Из всего сказанного выше ясно, что каждая задача в комплекте характеризуется точкой с координатами (***k*n**, ***k*p**) на шкале. Где ***k*n** – индекс задачи, характеризующий продуктивный (творческие задачи) вид деятельности, а ***k*p** – индекс, характеризующий репродуктивный (типовые задачи) вид деятельности.

 Кроме всего прочего, для построения шкалы сложности особую значимость имеет местоположение на ней двух задач – «очевидной» и «недоступной», ограничивающих весь возможный диапозон сложности задач. «Очевидную» задачу можно определить как задачу, которую полностью решают все участники без исключения. В решении «недоступной» задачи ни один из участников не способен сделать даже одного оцениваемого шага.

Возможен еще один интересный вариант задачи. Такую задачу условно назовем «нулевой». Она соответствует равновероятному распределению участников по набираемым баллам. «Нулевую» задачу можно одновременно считать как творческой, так и типовой.

 Сама шкала сложности, согласно теории, имеет вид, представленный на рис. 1:

Рис. 1. Вид шкалы сложности.

 Крайне интересным представляется расположение на этой шкале «очевидной», «недоступной» и «нулевой» задач. Очевидно, исходя из определения задач, видно, что «очевидная» задача – это есть предельный случай самой простой типовой задачи, то есть располагается она на оси ординат в −∞. «Недоступная» задача – есть предельный самый сложный случай творческих задач, располагается на оси абсцисс в +∞. «Нулевая» же задача, в силу своей двойственности, располагается на шкале в единственно пригодном месте – точке (0,0). Данная шкала недаром называется шкалой сложности, ведь видно, что усложнение творческих задач выражается перемещением точек, соответствующих задачам, вправо, вдоль оси абсцисс, а усложнение типовых задач – вверх, вдоль оси ординат.

 Возникает вопрос: как же отображается на шкале сбалансированный комплект задач? Ответ вполне очевиден – сбалансированный комплект отображается направленными отрезками прямых, проходящих перпендикулярно к биссектрисе главного координатного угла (см. рис. 1), и, как следствие, пересекающих координатные оси под углом в 45°. При этом направление этих отрезков указывает увеличение сложности от задачи к задаче во всем комплекте. Если привести простой пример с комплектом из 2-х задач, то получим следующую шкалу:

Рис. 2. Шкала сложности для двух комплектов из 2-х задач.

Для данного примера: комплект задач 1 и 2 считаем сбалансированным (задача 2 сложнее задачи 1), а комплект 3 и 4 считаем несбалансированным (задача 4 сложнее задачи 3).

 Данная шкала имеет огромное практическое значение, так как позволяет с большой точностью определить, является ли данный комплект задач сбалансированным или нет. Поэтому она используется в разработанной программе в качестве одного из показателей качества задач.

**§5. Требования к олимпиадным заданиям. Основные показатели качества.**

Введенное в §4 понятие сбалансированного комплекта олимпиадных заданий является краеугольным, и на его основе строятся основные требования к составителям этих заданий. Из данного понятия следуют следующие требования:

1. Все задания, которые предлагаются участникам олимпиады, должны быть **разноуровневыми**. Это необходимое условие для проведения олимпиад. При полной реализации этого требования осуществляется первый шаг к возможности дифференцированного подхода. Задачи должны быть разной сложности. При этом необязательно различие максимального балла за сложные и простые задачи. На мой взгляд, это является отпугивающим фактором для слабых учеников (эта задача сложная, я ее все равно не решу, а поэтому решать не буду) и заманивающим для сильных (за эту задачу дают большой балл, поэтому лучше решить две задачи по 10 баллов, чем четыре по 5). Учащиеся заранее видят сложность (или простоту) задачи, что крайне нежелательно. Если же все задачи имеют одинаковую балльную стоимость, то есть вариант, что потенциально слабый участник додумается до сложной задачи, а это поднимет его самооценку. В этом выражается гуманистический подход к олимпиаде.
2. Второе требование к олимпиадным заданиям – они должны быть максимально приближены к идеальному сбалансированному комплекту, то есть должны в равной степени затрагивать продуктивную и репродуктивную деятельность школьников. Это выражается симметрией точек на шкале сложности (см. выше) относительно биссектрисы главного координатного угла. На распределении по суммарному баллу (***S***, **ή**1) приближение к сбалансированному комплекту выражается в колоколообразном виде этого распределения.

Рис. 1. Идеальный вид распределения по **ή**1**.**

1. Третье требование заключается в том, что после проверки заданий и распределения участников по местам должно иметь место однозначное расположение участников на местах. То есть не должно быть несколько мест одного «достоинства». Если это требование выполнено, то можно говорить о максимальной реализации дифференцированного подхода и сбалансированного комплекта заданий.

Кроме вышеописанных требований, можно выделить еще достаточно много других, но мы ограничимся тем, что есть. Эти три требования в математическом виде реализованы в разработанной системе в виде трех параметров качества заданий.

Очевидно, что заранее невозможно предугадать о том, как будет разворачиваться обстановка при решении тех или иных заданий. В итоге, мы оперируем с протоколом результатов олимпиады и поэтому не можем точно направить ее ход в нужное русло. Однако, при помощи системы, можно оценить прошедшую олимпиаду и сделать выводы относительно следующей. В этом нам помогают три парамера качества заданий, которые полностью базируются на 3-х основных требованиях. Параметры эти таковы.

1. *Процент реализации сложности задач.* Этот параметр представляет собой математической выражение первого требования. Выражается он в процентах (%). В идеале должен быть, очевидно, равен 100%. Реально, такое значение получить крайне сложно, поэтому нормальным результатом можно считать 80-95%. Параметр зависит от количества блоков (для разного количества блоков – разный расчет). Если блок один, то параметр равен нулю и смысла, с точки зрения теории не имеет. Рассчитывается он следующим образом. В контексте данной теории этот параметр может быть использован применительно к каким-либо двум блокам заданий, то есть позволяет оценить, удалось ли реализовать большую сложность для одного блока задач относительно другого. Отсюда исходит принцип разного расчета для разного количества блоков. Практически, смысл расчета этого показателя сводится к следующему. При составлении олимпиадных заданий мы заранее знаем о том, какой блок является более сложным с точки зрения его решения, а какой – более легким. После решения этих блоков участниками, у нас есть реальные результаты для каждого блока. Далее, берется общий балл для более сложного блока (***x***1) и общий балл для более легкого блока (***x***2) (для каждого участника) и подсчитывается их разница (***x***1-***x***2). После проведения данных расчетов, строится гистограмма, подобная той, что изображена на рис. 2.

Рис. 2. Надежность реализации неравенства *x*1≥*x*2.

После построения такой гистограммы необходимо подсчитать число участников, для которых эта разница оказалась положительной (для данной гистограммы: общее количество участников равно 32, и разница ***x***1-***x***2 положительна для всех, то есть надежность реализации – 100%). Далее, берется процент этого количества от общего количества участников.

1. *Сбалансированность комплекта.* Этот параметр представляет собой второе требование, выраженное в графической форме. В идеале, точки на графике должны быть максимально симметричны относительно биссектрисы угла (об этом читайте в §4). Расчет этого параметра требует дополнительных введений и кардинально отличается для комплектов с разным количеством блоков. Стоит заметить, что в случае, если все задания помещены в один блок, параметр не имеет смысла.
2. *Коэффициент распределения по местам.* Данный параметр представляет собой, очевидно, третье требование. Диапазон значений параметра [0..1]. В идеальном случае должен быть равен 1 (каждый участник находится на своем заслуженном месте), в самом худшем случае равен 0 (все участники заняли 1 место). Расчет этой величины прост: , где ∆N - количество мест, N – общее количество участников.

Таким образом, рассчитав и визуализировав эти три параметра, можно с большой точностью сказать о реализации приведенных выше требований, а исходя из требований – сделать вывод об олимпиаде в целом.

Этими двумя задачами (распределение мест и оценка качества) занимается специальная программа, которая будет полностью описана в следующей главе.

**Глава 3. Автоматизированная система распределения мест и оценки уровня качества олимпиадных заданий.**

**§1. Общее описание. Системные требования.**

Программа **OLYMPS** разработана для ускорения процесса распределения мест на олимпиадах разных уровней, а также для оценки уровня качества олимпиадных заданий, предлагаемых на этих олимпиадах. Как уже отмечено выше (см. **Глава 1**), данный программный продукт имеет определенную теоретическую и математическую базу, которая кратко написана в **Главе 2**. Теперь, когда основные теоретические понятия, которыми оперирует ситема, были введены, приступим к описанию принципа ее работы. Первый вопрос, к которому мы обратимся – что необходимо для работы с продуктом.

Для ответа на данный вопрос ниже приведены две таблицы. Первая характеризует аппаратное обеспечение вычислительной машины, пригодной для работы с программой, вторая – программное обеспечение, которое **строго необходимо**  для корректной работы.

Таблица 1. Аппаратное обеспечение.

|  |  |  |
| --- | --- | --- |
|  | **Минимальное** | **Рекомендуемое** |
| Процессор (CPU) | 500MHz−1GHz | 1,3GHz и выше |
| Оперативная память (RAM) | 64Mb−128Mb | 256Mb |
| Видеокарта (SVGA) | Любая, поддерживающая 16−32bpp и 800х600@75Hz | – |
| Свободное место на ЖД | 20Mb | 50Mb |

Таблица 2. Программное обеспечение.

|  |  |
| --- | --- |
| Операционная система | Windows 98SEWindows 2000 (+SP3) |
| Дополнительное ПО | Borland Database Engine (BDE) v. 5.01 |

Приведенные в Таблице 1 требования не являются строго обязательными. Однако чем мощнее установлен процессор и больше оперативной памяти, тем быстрее будет работать программа. Приведенная в Таблице 1 рекомендуемая конфигурация будет работать относительно быстро со средним набором данных (до 150 человек). Если количество участников исчисляется сотнями (200-500), то возможно общее снижение производительности системы. Для решения проблемы необходим мощный процессор с большим объемом внутренней кэш-памяти (например, Intel Pentium 4) и 512Mb ОЗУ.

Наличие BDE является **обязательным** условием для работы программы. Установить его можно, например, вместе с Borland Delphi 6, при помощи которой была написана вся система.

Опишем комплект данного программного продукта. При установке данной системы в папку, которую выберет пользователь, будут скопированы следующие файлы: OLYMPS.EXE (это выполняемый файл программы), CONFIG.INI (это конфигурационный файл программы). Кроме этого, в директории программы будет создана папка BASES, в которой хранятся все созданные БД и файлы-описания к ним. Эти файлы имеют имя как у БД и расширение .OLP.

**§2. Описание главного окна программы.**

Теперь, после того как изложены основные требования, рассмотрим главное окно программы. Оно имеет вид, представленный на рис. 1.

Рис. 1. Главное окно программы.

 Как видно на рисунке, главное окно состоит из 6 частей.

 Первая часть – это главное меню программы. Рассмотрим его содержание. Первая опция меню – «Файл». При наведении указателя мыши на эту опцию всплывает меню, представленное на рис. 2:

Рис. 2. Меню «Файл».

Опция «Создать базу данных» ответственна за вызов диалогового окна создания локальной базы данных (далее БД). Может быть создано любое количество БД. Опция «Открыть базу данных» отвечает за вызов стандартного диалогового окна открытия файла. При помощи этой опции осуществляется открытие существующей локальной БД. Опция «Выход» отвечает за окончание работы с программой.

 Вторая опция главного меню – «Конфигурирование». Опции этого пункта меню представлены на рис. 3:

Рис. 3. Меню «Конфигурирование».

Опция «Общие настройки» вызывает окно конфигурации программы. Опция «Вид БД» сохраняет текущее расположение колонок БД в конфигурационный файл. Если эти настройки сохранены, то далее отображение БД строится исходя из них. Если настройки вида БД не сохранялись, то она отображается по умолчанию.

Следующий пункт главного меню – «Сервис». Опции этого пункта представлены на рис. 4:

Рис. 4. Меню «Сервис».

Опция «Распределение мест» отвечает за вызов окна, в котором осуществляется, собственно говоря, сам процесс распределения мест. Опция «Уровень качества» вызывает окно, в котором осуществляется оценка уровня качества заданий.

И, наконец, последний пункт главного меню – «Помощь». Опции данного пункта представлены на рис. 5:

Рис. 5. Меню «Помощь».

Опция «О программе» вызывает стандартное окно ОС «О программе». Опция «Справка» загружает этот документ.

На этом мы заканчиваем описание главного меню программы и переходим к описанию остальных пяти частей главного окна.

Вторая часть окна – это, собственно говоря, то место, в котором отображается сама текущая БД. На рис. 6 представлено главное окно с открытой БД.

Рис. 6. Главное окно с открытой БД.

 Следующие три части окна очень похожи друг на друга. Это так называемые панели инструментальных кнопок. Кнопки первой панели «Редактирование» отвечают за физическое редактирование записей в БД. Кнопка «Добавить участника» вызывает диалоговое окно добавления участника, а кнопка «Удалить участника» – окно удаления. Кнопки второй панели «Сервисы» по своим функциям аналогичны опциям главного меню «Сервис». Особенный интерес представляет кнопки группы «Отчеты». Эти кнопки ответственны за вызов окон построения печатной формы отчета и сохранения этого отчета в файл соответственно.

 Последняя, пятая часть главного окна – информационная панель. Она располагается под главным меню. На этой панели отображается дополнительная информация, а если точнее, то имя БД, количество (записей) и текущее системное время.

 После того, как мы рассмотрели главное окно программы, переходим к описанию основных действий в ней.

**§3. Основные действия в системе.**

 В данном параграфе описаны основные действия, которые позволяет осуществлять программа.

**Создание новой базы данных.**

 Разработанная система позволяет создавать сколь угодно много баз данных. Сразу необходимо отметить, что все базы, которые могут быть созданы при помощи данной программы, имеют **одинаковую структуру**, а отличаться могут лишь данными, которые в них находятся. Возможность создания нескольких баз введена не зря. Предполагается, что пользователь системы может вести накопительную статистику работы, создавая новую базу для новой олимпиады. Этим обеспечивается одно из главных правил разработки прикладного ПО – удобство пользователя.

 Создание базы данных в программе осуществляется при помощи пункта главного меню «Файл» «Создать базу данных». При выборе этого пункта на экране появляется диалоговое окно создания базы данных, представленное на рис. 1:

Рис. 1. Диалоговое окно «Создание базы данных».

В этом окне и осуществляется процесс подготовки к созданию базы. При этом необходимо заполнить два поля – «Имя БД» и «Комментарий». В поле «Имя БД» вводится произвольное имя базы, состоящее **только** из символов английского алфавита и цифр, причем цифра **не может** быть первым символом в имени базы. Такие жесткие требования к имени базы обусловлены тем, что введенный в поле «Имя БД» текст является физическим именем файла, который после создания будет располагаться в папке BASES. Поэтому рекомендуется не вкладывать большого смысла в это имя, называть базу просто (например, DBOLYMP1). Для более полного описания создаваемой базы предусмотрено поле «Комментарий». В этом поле может быть введен абсолютно произвольный текст, отражающий смысл базы. Текст может вводиться символами любого алфавита.

 Непосредственно создание базы данных осуществляется нажатием кнопки «Создать». При этом создается и открывается локальная БД, а также записывается файл комментария (как отмечено выше, такой файл имеет расширение .OLP и хранится в той же папке, что и базы), который в дальнейшем используется для информирования пользователя. Создание БД сопровождается появлением сообщения (см. рис. 2).

Рис. 2. Сообщение об успешном создании БД.

 Программа защищена от неправильного ввода, но все же для исключения проблем рекомендуется до этого дело не доводить.

 Кнопка «Отмена» предназначена для выхода из окна создания БД. При этом никакие файлы не создаются.

**Удаление базы данных.**

 Удалить ненужную локальную базу данных непосредственно из программы **нельзя**. Если это все-таки необходимо, придется удалять эту БД вручную (как это делается, можно посмотреть в любой книге по ОС Windows в разделе «Удаление файлов»).

**Открытие существующей базы данных.**

 Если файл нужной БД уже существует, то необходимо открыть его для записи или чтения. Эта возможность реализована в пункте главного меню «Файл» «Открыть базу данных». При выборе этого пункта на экране появляется стандартное окно открытия файла, которое представлено на рис. 3:

Рис. 3. Окно открытия базы данных.

В этом окне необходимо выбрать файл базы, который необходимо открыть, а потом нажать кнопку «Открыть». Для выхода нужно нажать кнопку «Отмена».

 Открытие базы данных сопровождается появлением сообщения (см. рис. 4).

Рис. 4. Сообщение об успешном открытии БД.

**Запись данных в базу.**

 Для реализации данного действия, необходимо, чтобы база сначала была открыта (см. «Открытие существующей базы данных»). Если это условие выполнено, то нужно воспользоваться кнопкой инструментальной панели «Редактирование» «Добавить участника». При этом на экране появится диалоговое окно, представленное на рис. 5:

Рис. 5. Диалоговое окно добавления нового участника.

В этом окне необходимо заполнить **все** поля. Поле «ФИО участника» есть фамилия, имя, отчество добавляемого участника. Поле «Школа (адрес, №)» – адрес и номер школы, в которой обучается участник. Поля «Баллы за задания» − это набранные участником баллы за 6 заданий соответственно. Для записи данного участника в базу необходимо нажать кнопку «Выполнить». При этом суммарный балл участника программа подсчитает автоматически. Кнопка «Отмена» предназначена для выхода из диалогового окна.

**Удаление записи из базы.**

 Для удаления какого-либо участника из БД воспользуйтесь кнопкой инструментальной панели «Редактирование» «Удалить участника». При этом на экране появится диалоговое окно, представленное на рис. 6:

Рис. 6. Диалоговое окно удаления участника.

Для удаления участника из БД необходимо выбрать его ФИО в раскрывающемся списке и нажать кнопку «Удалить». Нажмите кнопку «Отмена» для выхода из окна без удаления. В этом процессе есть одно «но». Если база не имеет записей (т.е. база пустая), то программа известит об этом сообщением, а кнопка «Удалить» и раскрывающийся список станут неактивными.

 На этом мы закончили описание основных действий в системе и переходим к описанию основной программной части, которая является практическим воплощением рассмотренной выше теории.

**§4. Распределение мест и оценка качества заданий в автоматизированной системе.**

 Перед тем, как непосредственно перейти к реализации теоретического аспекта на практике, необходимо сконфигурировать программу под конкретные задачи. Конфигурация системы осуществляется при помощи диалогового окна, которое вызывается пунктом главного меню «Конфигурирование» «Общие настройки». Это окно представлено ниже на рис. 1:

Рис. 1. Окно конфигурации программы.

При помощи данного окна осуществляется общее конфигурирование всей системы, поэтому рекомендуется основательно ознакомиться с каждым из параметров.

 Первый параметр – количество блоков задач. Этот параметр реализован при помощи панели «Количество блоков». Можно выбрать три различных варианта: 1 блок, 2 блока и 3 блока. Стоит иметь в виду, что если выбран вариант с 1 блоком задач, то два из трех параметров качества заданий лишаются смысла. Это приводит к очень неточному результату. Кроме этого, нужно знать, что под блоком, в контексте программы, подразумевается последовательный набор заданий. Это означает, что если выбирается вариант с тремя блоками, то эти блоки представляют собой следующую комбинацию: первый блок – задание №1,задание №2 и задание №3, второй блок – задание №4, задание №5 и задание №6. Если же выбран вариант с двумя блоками, то каждый блок – это: задание №1 и задание №2, задание №3 и задание №4, задание №5 и задание №6. Эту особенность необходимо иметь в виду при вводе участников в базу. По умолчанию установлен вариант с тремя блоками.

 Второй параметр – максимальный балл за каждой задание. Этот параметр реализован в панели «Задания». Здесь необходимо ввести бальную стоимость каждой задачи. Подразумевается, что для всех заданий бальная стоимость одинакова. Причина такого подхода описана выше. Очевидно, что эта бальная стоимость не должна быть меньше, чем соответствующий балл у какого-либо участника. За этим моментом необходимо проследить.

 Третий параметр – порядок распределения. Этот параметр программно реализует принцип дифференцированного подхода к распределению мест на олимпиадах. Возможно три варианта этого порядка: для слабого коллектива учащихся (1→2→3), для сильного коллектива (1→3→2) и для смешанного коллектива (1→2→3, если **ή**1<0, и 1→3→2, если **ή**1≥0). По умолчанию установлен последний вариант.

 После конфигурирования этих параметров необходимо нажать кнопку «Записать» для записи конфигурации или кнопку «Отмена» для отказа от записи и выхода из окна.

 После записи конфигурации необходимо открыть необходимую базу данных (если она еще не открыта). Для автоматического распределения участников по местам необходимо воспользоваться кнопкой инструментальной панели «Сервисы» «Распределение мест» или пунктом главного меню «Сервис» «Распределение мест». При этом на экране появится диалоговое окно, отвечающее за процесс распределения мест, которое изображено на рис. 2. В этом окне можно получить полностью автоматизированное распределение участников по местам. В верхней его части представлена текущая база данных, в которой рассчитаны значения трех параметров **ή**1, **ή**2, **ή**3 (этим параметрам соответствуют поля Параметр1, Параметр2 и Параметр3 соответственно). По умолчанию (то есть при вызове этого окна) база сортируется таким образом, каким указано в конфигурации. Однако при желании, можно отсортировать базу по-другому. Эта возможность реализована при помощи группы переключателей «Сортировать по». Предусмотрена возможность сортировки только по **ή**1, только по **ή**2, только по **ή**3. Также введена возможность автоматической сортировки базы. При этом способе программа сама сортирует базу так, как считает правильным.

 Расположенная под группой переключателей информационная панель предоставляет пользователю информацию об установленной конфигурации системы.

Рис. 2. Окно распределения мест.

 Левее этой панели расположена область построения диаграмм. При первом открытии окна в этой области построений нет. Для того чтобы построить распределение по какому-либо из параметров нужно воспользоваться всплывающим меню, которое появляется при нажатии правой кнопки мыши на области построения диаграммы. Это меню представлено на рис. 3:

Рис. 3. Всплывающее меню построения распределений.

Первая опция строит распределение по **ή**1, вторая – по **ή**2, третья – по **ή**3, четвертая – по суммарному баллу. Следующие три опции имеют дополнительные функции: первая меняет цвет диаграммы, вторая меняет цвет фона диаграммы, а третья печатает диаграмму на принтере.

 Теперь рассмотрим, как осуществляется оценка качества заданий в контексте данной автоматизированной системы. Этот процесс происходит при нажатии либо кнопки инструментальной панели «Сервисы» «Оценка качества», либо пункта главного меню «Сервис» «Оценка качества». Локальная БД должна быть открыта перед совершением этих действий.

 Стоит сразу заметить одну важную особенность: никакой дополнительной конфигурации программы для данной операции нет. Эта особенность связана с тем, что оценка уровня качества заданий в программе ведется на основе значений трех параметров, которые полностью описаны в §5 Главы 2. Все данные, которые необходимы для расчета этих параметров, берутся из того протокола олимпиады, который вводит оператор.

 Окно, отвечающее за процесс распределения мест, представлено ниже на рис. 4. Как видно из рисунка, данное окно состоит из четырех частей. Первая часть окна – диаграмма «Сбалансированность комплекта». Эта диаграмма фактически представляет собой шкалу сложности задач. Круглые точки красного цвета на шкале представляют собой положения на шкале сложности блоков задач (на рис. 5 представлен комплект из 3 блоков). Зеленая линия, проходящая по области диаграммы, представляет собой биссектрису главного координатного угла шкалы сложности.

Рис. 4. Окно «Оценка уровня качества заданий».

Как следует из теории, эта линия необходима для оценки сбалансированности блоков задач или комплекта в целом. В системе можно построить данную диаграмму для всех заданий олимпиады. Для этого сконфигурировать систему на работу с одним блоком задач.

 Вторая часть окна – это диаграмма распределения участников по суммарному баллу. Эта диаграмма необходима для сравнения получившегося экспериментального распределения с идеальным. Как известно, идеальное распределение представляет собой «колокол».

 Третья часть окна – это диаграмма надежности реализации сложности. Она необходима для визуализации процесса расчета этого параметра.

 Четвертая часть окна – это информационная панель, на которой отображается текущая конфигурация системы, а также численные значения каждого параметра (если эти значения есть).

 Видно, что система не требует вмешательства пользователя при оценке уровня качества заданий. Она только демонстрирует полученный результат. Никаких дополнительных возможностей данное окно не предоставляет.

 Выше представлено полное руководство по работе с разработанной системой. Таким образом, на этом можно завершить. В следующей главе вниманию читателей будет представлено описание результатов работы системы с реальными наборами данных.

**Глава 4. Результаты, полученные в ходе исследования.**

**§1. Результаты, полученные на основе олимпиад.**

 В комплекте с программой поставляется БД, которая представляет собой протокол *областной олимпиады по физике в 11 классах*. Данная БД размещается в файле DBOLYMP1.DB. Общее количество записей в этой БД равно 32. Ниже будут представлены результаты, для этого набора, которые были получены при помощи разработанной системы. Непосредственно сами фамилии участников можно посмотреть при помощи программы, а мы опишем только результаты по двум операциям: *распределение участников по местам* и *оценка уровня качества заданий*.

**Распределение по местам.**

Ниже, на рис. 1, представлено окно распределения мест для этой БД. Конфигурация системы для этой БД следующая: количество блоков задач – 3, максимальный балл за задачу – 6, установлен второй порядок сортировки. На рисунке видно, что данный набор является потенциально слабым, так как значения поля «Параметр1» (**ή**1) почти все отрицательны. Поэтому, можно сортировать по второму методу сортировки (сортировка для слабого коллектива). Очевидно, что победителем олимпиады является Панкратов С. А., который набрал максимальный суммарный балл (18). При желании, можно отсортировать участников по одному из трех параметров или автоматически.

Рис. 1. Распределение по местам для данной БД.

Ниже места участникам присуждаются исходя из положения их результатов в таблице. На следующем рисунке представлены распределения участников по 4 параметрам (**ή**1, **ή**2, **ή**3, ***S***).

Рис. 2. Распределения по параметрам.

Из этого рисунка видно, что до идеального вида этим распределениям далеко. Полученный вид распределений указывает на то, что задания, которые предлагались в качестве базового блока, были слишком простыми (об этом говорит и сам протокол), а задания, которые предлагались в качестве квалификационного блока, оказались практически нерешаемыми. Можно сделать вывод о том, что, скорее всего, комплект заданий, предложенный этим участникам, сбалансированным не будет. Верно ли это предположение или нет, можно узнать лишь в результатах оценки качества этих заданий.

**Оценка уровня качества заданий.**

После того, как распределение участников по местам закончено, можно попытаться оценить качество заданий. Окно оценки качества приведено ниже на рис. 3:

Рис. 3. Оценка качества заданий.

Диаграмма «Сбалансированность комплекта» показывает нам, что наше предположение оказалось верным. Данный комплект не является сбалансированным. Однако два других параметра указывают на достаточно хорошие показатели. Видно, что составителям удалось реализовать большую сложность 2 блока по сравнению с 1, а также большую сложность 3 блока по сравнению с 1, на 100%. Большая же сложность 3 блока по сравнению со 2 реализована лишь на 68,75%, что тоже считается хорошим результатом. Коэффициент мест для данного коллектива равен 0,71875. Это тоже достаточно хороший результат.

**Описание в целом.**

В целом об олимпиаде можно сказать следующее: составителям удалось реализовать основные требования к проведению такого мероприятия как олимпиада, однако участники оказались достаточно слабыми для такого класса заданий, которые предлагались им для решения.

**§2. Результаты, полученные на основе педагогической практики.**

Кроме БД, которая описана выше в §1, в комплекте поставляется еще одна локальная БД (содержится в файле DBOLYMP5.DB). Она сформирована мной на педагогической практике и включает результаты учащихся моего класса по нескольким работам. Для демонстрации мультифункциональности системы решено было использовать данную БД для демонстрации. При проверке использовалась следующая конфигурация системы: количество блоков задач – 3, использовался смешанный метод сортировки, максимальный балл за задания – 5.

**Распределение по местам.**

 Окно распределения мест для данного набора данных представлено ниже на рис. 1. Из этого рисунка видно, что лидером по успеваемости по физике в 11Б классе является Нечаева Е., которая набрала самый большой суммарный балл. Этот вывод подтверждается классным журналом. Распределения по четырем параметрам представлены на рис. 2. Из вида этих распределений можно сделать вывод о том, что предложенный мною комплект задач не является сбалансированным.

Рис. 1. Распределение мест для этой БД.

Более того, если обратить внимание на распределение по суммарному баллу, то видно, что «колокол» имеет максимум не в зоне среднестатистических результатов (так должно быть в идеале), а в зоне, приближенной к максимально возможному баллу.

Рис. 2. Распределения по четырем параметрам.

Это говорит о том, что задания, предложенные на самостоятельных работах, оказались очень простыми для учащихся и, поэтому, большая их часть набрала высокий балл. Вид других распределений только подтверждает это обстоятельство.

**Оценка уровня качества.**

В результате оценки качества для этой БД были получены следующие результаты. Диаграмма сбалансированности комплекта представлена на рис. 3. Она подтверждает наше предположение о несбалансированности комплекта. Точки, каждая из которых характеризует отдельный блок, находятся практически «друг на друге». Для сбалансированности комплекта это неприемлемо.

Рис. 3. Диаграмма сбалансированности комплекта.

Следующая диаграмма, необходимая для оценки качества, это диаграмма надежности реализации сложности. Она представлена на рис. 4. При помощи данной диаграммы получены следующие численные результаты. Надежность реализации неравенств:

* ***x***1≥***x***2 – 50%;
* ***x***2≥***x***3 – 64,29%;
* ***x***1≥***x***3 – 64,29%.

Эти значения говорят о том, что в целом удалось реализовать разную сложность для разных задач. Однако сами по себе эти показатели имеют достаточно маленькие значения.

Рис. 4. Диаграмма надежности реализации сложности.

И последний параметр, это коэффициент мест. Для данной БД его значение составляет 0,93. Это говорит о том, что 93% всех участников занимают одно заслуженное место, и только 7% делят свое место с другими. Такой результат считается достаточно высоким.

**Описание в целом.**

 Все параметры, описанные выше, позволяют сделать несколько выводов касательно данной БД. Во-первых, параметры качества заданий говорят о достаточно высоком классе подбора заданий, однако этот подбор не явился в результате сбалансированным, а это большой минус. Во-вторых, необходимо подбирать (для этого коллектива) задачи с большим уровнем сложности. Это может помочь в выявлении скрытых талантов учеников.

**§3. Результаты, полученные на основе ведомостей студентов.**

Самым интересным из проделанной работы явились результаты, полученные на основе зимней сессии студентов физико-математического факультета нашего университета. В качестве тестируемых были взяты результаты экзаменов трех групп по трем предметам. Естественно, для чистоты эксперимента было необходимо полное различие в фактической сложности предметов. Поэтому, были выбраны три следующих: возрастная психология, общая физика и математический анализ. Причем последние два предмета были взяты у двух групп отделения «Физика». Сдавались эти экзамены в зимнюю сессию одним и тем же преподавателям.

Не секрет, что в студенческих кругах психология считается легким экзаменом, общая физика – средним, а математический анализ – сложным. Однако такое деление предметов во многом зависит от преподавателя. Например, если преподаватель принимает экзамен по математическому анализу очень мягко, то, очевидно, такой экзамен автоматически становится простым, а если экзамен принимается очень жестко, то он становится сложным.

Кроме этого сложность предмета зависит еще и от его фактической сути. Например, тот же математический анализ является точной наукой, а это накладывает определенный отпечаток на его сложность. Психология же во многом является очевидным предметом (это многим только кажется). Эти факты склоняют его сложность в более простую сторону. Физика же есть эмпирическая наука. Это значит, что практически все ее законы можно проверить на опыте. Это обстоятельство вызывает определенный интерес у студентов, а это способствует лучшему пониманию предмета.

Из этих соображений вытекает суть той идеи, которую мы используем для описания конкретно этих результатов. А идея проста: если есть набор оценок по каким-либо предметам, то, построив распределение по этим оценкам, можно оценить жесткость экзаменаторов по конкретным дисциплинам.

Первая БД (содержится в файле DBOLYMP2.DB) представляет собой протокол экзамена по математическому анализу у групп 15 и 16 ФМФ. В результате обработки данного протокола было получено следующее распределение по суммарному баллу:

Рис. 1. Распределение по суммарному баллу для DBOLYMP2.

Очевидно, что максимально возможный балл за экзамен равен 5, а минимально возможный балл условно принимаем равным 2. Из вида диаграммы можно сделать вывод о том, что самое оценка, которую получила большая часть студентов, это 2. А оценка, которую получила меньшая часть студентов, это 5. На симметричный относительно среднестатистического балла «колокол» полученное распределение явно не похоже. Причина этого может быть объяснена при помощи нескольких вариантов. Первый вариант заключается в чрезвычайной жесткости экзаменатора, то есть, либо он спрашивает со студентов то, чего не рассказывал на лекциях, либо просто спрашивает очень жестко. Второй вариант говорит о том, что сами студенты пришли на экзамен абсолютно не готовыми.

Вторая БД (содержится в файле DBOLYMP3.DB) представляет собой протокол экзамена тех же групп, но по общей физике. В результате обработки этого протокола получено следующее распределение.

Рис. 2. Распределение по суммарному баллу для DBOLYMP3.

Аппроксимируя данное распределение, мы получаем практически идеальный «колокол». Это говорит о сбаланированном подходе экзаменатора к приему экзамена. То есть, при таком подходе осуществляется гуманистический подход к личности студента, а также равномерное соединение всех трех режимов испытания.

Третья БД (содержится в файле DBOLYMP4.DB) представляет собой протокол экзамена по возрастной психологии у 11 группы ФМФ. В результате обработки данного протокола получено следующее распределение. На рис. 3 мы видим полную противоположность распределению, полученному при анализе экзамена по математическому анализу. Здесь большая часть студентов получила 5, а совсем малая часть получила 3.

Рис. 3. Распределение по суммарному баллу для DBOLYMP4.

Двоек нет вообще. Такое распределение характерно для щадящего режима испытания. То есть экзаменационные вопросы, которые предлагались студентам, были очень простыми.

Исходя из всех протоколов, описанных выше, можно сделать один очень важный вывод. На экзамене необходимо придерживаться такого варианта испытания, который представлен в описании экзамена по физике. Нет смысла в очень жестком испытании (как на математическом анализе), однако нет смысла и в очень щадящем режиме (как на психологии). Необходимо выбрать «золотую середину». Тогда преподаватель будет на 100% уверен в своей рациональности.

**Глава 5. Заключение.**

**§1. Итоги исследования.**

Подводя итоги проведенных исследований, можно выделить ряд момен­тов. Главный из них заключается в том, что на примере интеллектуального испытания удалось продемонстрировать саму возможность количественного моделирования педагогического процесса, создания автоматизированной системы и показать практическую значи­мость получаемых результатов. При разработке модели была реализована достаточно последовательная схема педагогического моделирования, в про­цессе которой была сформулирована замкнутая система исходных педагоги­ческих положений; введены понятия идеализированного испытания и идеа­лизированного ансамбля испытуемых школьников; выбран и изучен матема­тический объект, адекватный оптимальным педагогическим итогам испыта­ний идеализированного ансамбля; дана педагогическая интерпретация свойств этого объекта; определены оптимальный педагогический и математический формат интеллектуального испытания и взаимосвязь его исходных и итого­вых показателей; исходя из теоретических предпосылок, разработана автоматизированная система, позволяющая визуализировать смысл всей теории.

Разработка модели и системы велась по отношению к конкретным вариантам интел­лектуального испытания, в качестве которых были выбраны испытания учас­тников олимпиад по физике и студентов физико-математического факультета РГПУ на различных учебных дисциплинах.

Разработанную схему отличает доказательность и обоснованность, опре­деляемые достоверным характером исходных педагогических положений модели и строгостью ее математического аппарата. Этот аппарат дает воз­можность вести проектирование итогов интеллектуального испытания, мо­жет составить основу инструментария педагогической экспертизы испыта­ния на предмет его соответствия критериям общепедагогического характера. Математический аппарат модели позволил построить шкалу сложности за­дач, ввести понятие сбалансированного комплекта задач, обеспечивающего оптимальное соответствие интеллектуального испытания новой шкале цен­ностных приоритетов образования. Был рассмот­рен вопрос о выборе показателей приоритета, о дифференцированном и не­дифференцированном порядке распределения испытуемых школьников по занимаемым местам.

Интерес представляют выявленные свойства идеализированного ансамбля как «механического» объединения школьников, на уровне которого реализуется наи­более простая, но в то же время и самая фундаментальная форма взаимоотноше­ния личности и коллектива, выражающаяся в элементарном сложении. Подобное объединение школьников, лишенное выраженных межличностных взаимоотно­шений, хорошо соотносится с ансамблем участников олимпиады. Идеализиро­ванный ансамбль отличается заведомой аддитивностью своих свойств. Исследо­вание подобных систем в научном плане имеет большой интерес. История науки дает основания надеяться, что они могут составить простейшую базу для постро­ения количественных шкал, необходимых для измерения педагогических харак­теристик. Объединение школьников, лишенное выраженных межличностных и групповых взаимоотношений, интересно и по отношению к проблемам учебно-воспитательного коллектива. Оно может сыграть роль первичной матрицы, на фоне которой влияние на коллектив межличностных и групповых взаимоотно­шений должно прослеживаться наиболее рельефно.

Практическую значимость имеют разработанные в рамках модели 2-блочные и 3-блочные макеты олимпиадных заданий, характеризуемые оптимальным соответствием педагогической модели. Макеты были отрабо­таны в условиях реального эксперимента. Итоги этого эксперимента показа­ли, что педагогическая модель подтверждается опытом и имеет хорошие пер­спективы для практического использования. Она решает большинство нако­пившихся на уровне региональных олимпиад проблем, способствуя переводу этих олимпиад в режим «талантосбережения».

Отмечая положительные моменты проведенного исследования, можно остановиться в заключение на перспективах его практического и теоретичес­кого использования. Полученные результаты интересны тем, что являются, строго говоря, лишь первым шагом на пути построения полномасштабной педагогической модели интеллектуального испытания. Характеризуя этот шаг в теоретическом плане, можно сказать, что он был сделан в правильном на­правлении, поскольку вывел нас на исходные позиции теории вероятности, способной дать вероятностное истолкование различия репродуктивного и продуктивного видов деятельности школьников и процесса их соединения. Полученные данные важны тем, что делают контуры вероятностной модели интеллектуального испытания просматриваемыми. Залогом этого является математический аппарат модели, сформированный в процессе настоящего исследования.

Что касается перспектив практического использования системы, то они достаточно понятны. Разработанная система и модель интеллектуального испытания в ее сегодняшнем виде пригодна для определения формата не только олимпи­ад, но и большинства способов оценки и контроля уровня знаний учащихся (контрольных и самостоятельных работ, экзаменов, тестов). В рамках моде­ли, например, можно определить возможный формат единого экзамена школь­ников, к которому должна перейти отечественная система образования. Дос­таточно лишь задать его выходные параметры (разрешение, уровень интел­лектуальной нагрузки школьников и т.д.).

Модель может быть использована и для решения ряда других задач. Она позволяет исследовать структуру ансамбля испытуемых школьников, что мо­жет быть интересным, скажем, для оценки влияния дифференциации систе­мы образования на уровень подготовки школьников. С помощью модели, на­пример, можно оценивать не только уровень сложности задач, но и уровень профессиональной подготовки учителя, через его знание класса и его способ­ность подготовить разноуровневое контрольное задание, сбалансированное по видам учебной деятельности.

**Приложение. Исходный код системы (по модулям).**

**Модуль 1.**

Код данного модуля отвечает за формирование главного окна программы.

unit Unit1;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, Menus, ToolWin, ComCtrls, Grids, DBGrids, ExtCtrls, StdCtrls,

 Buttons, DB, DBTables, INIFiles, ShellAPI;

type

 TForm1 = class(TForm)

 MainMenu1: TMainMenu;

 N1: TMenuItem;

 N2: TMenuItem;

 N3: TMenuItem;

 N4: TMenuItem;

 N5: TMenuItem;

 N6: TMenuItem;

 N7: TMenuItem;

 N10: TMenuItem;

 N12: TMenuItem;

 N13: TMenuItem;

 N14: TMenuItem;

 N15: TMenuItem;

 N16: TMenuItem;

 N17: TMenuItem;

 N18: TMenuItem;

 N20: TMenuItem;

 ToolBar1: TToolBar;

 DBGrid1: TDBGrid;

 Panel1: TPanel;

 GroupBox1: TGroupBox;

 BitBtn1: TBitBtn;

 BitBtn2: TBitBtn;

 GroupBox2: TGroupBox;

 BitBtn3: TBitBtn;

 BitBtn4: TBitBtn;

 GroupBox3: TGroupBox;

 BitBtn5: TBitBtn;

 BitBtn6: TBitBtn;

 dbOlymp: TDatabase;

 dsOlymp: TDataSource;

 Table1: TTable;

 OpenDialog1: TOpenDialog;

 Label1: TLabel;

 procedure N5Click(Sender: TObject);

 procedure FormClose(Sender: TObject; var Action: TCloseAction);

 procedure N16Click(Sender: TObject);

 procedure N2Click(Sender: TObject);

 procedure FormCreate(Sender: TObject);

 procedure BitBtn1Click(Sender: TObject);

 procedure BitBtn2Click(Sender: TObject);

 procedure N7Click(Sender: TObject);

 procedure N10Click(Sender: TObject);

 procedure BitBtn3Click(Sender: TObject);

 procedure N3Click(Sender: TObject);

 procedure BitBtn4Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

 Config:TIniFile;

implementation

uses Unit2, Unit3, Unit4, Unit5, Unit6, Unit7;

{$R \*.dfm}

procedure TForm1.N5Click(Sender: TObject);

begin

 if Application.MessageBox('Закончить работу?','Выход',mb\_YesNo+mb\_IconAsterisk)=idYes

 then Application.Terminate

 else Exit;

end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);

begin

 if Application.MessageBox('Закончить работу?','Выход',mb\_YesNo+mb\_IconAsterisk)=idYes

 then Action:=caFree

 else Action:=caNone;

end;

procedure TForm1.N16Click(Sender: TObject);

begin

 ShellAbout(Form1.Handle,'FoxSoft Olymps v1.0 (beta)','Programm by Chernetckiy Kirill aka Fox',Application.Icon.Handle);

end;

procedure TForm1.N2Click(Sender: TObject);

begin

 Form6.ShowModal;

end;

procedure TForm1.FormCreate(Sender: TObject);

begin

 try

 with dbOlymp do

 begin

 Connected:=False;

 Params.Clear;

 Params.Add('PATH='+GetCurrentDir+'\Bases');

 Params.Add('DEFAULT DRIVER=PARADOX');

 Params.Add('ENABLE BCD=FALSE');

 Connected:=True;

 end;

 except

 Application.MessageBox('Ошибка инициализации баз данных. Возможно не установлен BDE.','Ошибка',mb\_Ok+mb\_IconHand);

 Application.Terminate;

 end;

end;

procedure TForm1.BitBtn1Click(Sender: TObject);

begin

 Form2.ShowModal;

end;

procedure TForm1.BitBtn2Click(Sender: TObject);

begin

 Form3.ShowModal;

end;

procedure TForm1.N7Click(Sender: TObject);

begin

 Form4.ShowModal;

end;

procedure TForm1.N10Click(Sender: TObject);

begin

try

 if Application.MessageBox('Хотите сохранить текущий вид таблицы?','Подтвердите',mb\_YesNo+mb\_IconAsterisk)=idNo

 then Exit

 else

 begin

 Config:=TIniFile.Create(GetCurrentDir+'\Config.ini');

 with Config do

 begin

 WriteString('GRID\_CONFIG','FIO',IntToStr(DBGrid1.Columns[0].Width));

 WriteString('GRID\_CONFIG','MARK1',IntToStr(DBGrid1.Columns[1].Width));

 WriteString('GRID\_CONFIG','MARK2',IntToStr(DBGrid1.Columns[2].Width));

 WriteString('GRID\_CONFIG','MARK3',IntToStr(DBGrid1.Columns[3].Width));

 WriteString('GRID\_CONFIG','MARK4',IntToStr(DBGrid1.Columns[4].Width));

 WriteString('GRID\_CONFIG','MARK5',IntToStr(DBGrid1.Columns[5].Width));

 WriteString('GRID\_CONFIG','MARK6',IntToStr(DBGrid1.Columns[6].Width));

 WriteString('GRID\_CONFIG','SUMMARK',IntToStr(DBGrid1.Columns[7].Width));

 WriteString('GRID\_CONFIG','SCHOOL',IntToStr(DBGrid1.Columns[8].Width));

 UpdateFile;

 end;

 Config.Free;

 Application.MessageBox('Настройки успешно сохранены!','Сообщение',mb\_Ok+mb\_IconAsterisk);

 end;

except

 Application.MessageBox('Ошибка сохранения настроек!','Ошибка',mb\_Ok+mb\_IconHand);

 Exit;

end;

end;

procedure TForm1.BitBtn3Click(Sender: TObject);

begin

 Form5.ShowModal;

end;

procedure TForm1.N3Click(Sender: TObject);

 var

 i:integer;

begin

OpenDialog1.InitialDir:=GetCurrentDir+'\Bases';

if OpenDialog1.Execute then

 begin

 try

 with Table1 do

 begin

 Active := False;

 DatabaseName := 'Olymp';

 TableType := ttParadox;

 TableName := ExtractFileName(OpenDialog1.FileName);

 Active:=True;

 end;

 dsOlymp.DataSet:=Table1;

 DBGrid1.DataSource:=dsOlymp;

 Application.MessageBox('База успешно открыта.','Сообщение',mb\_Ok+MB\_ICONASTERISK);

 except

 Application.MessageBox('Ошибка открытия базы. Возможно она не существует.','Ошибка',mb\_Ok+MB\_ICONHAND);

 Exit;

 end;

end;

end;

procedure TForm1.BitBtn4Click(Sender: TObject);

begin

 Form7.ShowModal;

end;

end.

**Модуль 2.**

Код этого модуля отвечает за формирование баз данных, то есть за запись участника.

unit Unit2;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, ExtCtrls, StdCtrls, Buttons;

type

 TForm2 = class(TForm)

 Panel1: TPanel;

 GroupBox1: TGroupBox;

 BitBtn1: TBitBtn;

 BitBtn2: TBitBtn;

 Edit1: TEdit;

 Edit2: TEdit;

 Label1: TLabel;

 Label2: TLabel;

 Edit3: TEdit;

 Edit4: TEdit;

 Edit5: TEdit;

 Edit6: TEdit;

 Edit7: TEdit;

 Edit8: TEdit;

 Label3: TLabel;

 Label4: TLabel;

 Label5: TLabel;

 Label6: TLabel;

 Label7: TLabel;

 Label8: TLabel;

 Label9: TLabel;

 procedure BitBtn1Click(Sender: TObject);

 procedure BitBtn2Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form2: TForm2;

implementation

uses Unit1, DB, DBTables;

{$R \*.dfm}

procedure TForm2.BitBtn1Click(Sender: TObject);

begin

 ModalResult:=mrCancel;

end;

procedure TForm2.BitBtn2Click(Sender: TObject);

 var

 Fio,Sch:string;

 n1,n2,n3,n4,n5,n6,snum:double;

begin

 Fio:=Edit1.Text;

 Sch:=Edit2.Text;

 if (Fio='') or (Sch='') then

 begin

 Application.MessageBox('Не введено ФИО или Школа!','Ошибка',mb\_Ok+mb\_IconHand);

 Edit1.SetFocus;

 Exit;

 end;

 try

 n1:=StrToFloat(Edit3.Text);

 n2:=StrToFloat(Edit4.Text);

 n3:=StrToFloat(Edit5.Text);

 n4:=StrToFloat(Edit6.Text);

 n5:=StrToFloat(Edit7.Text);

 n6:=StrToFloat(Edit8.Text);

 except

 Application.MessageBox('Одно или несколько текстовых полей заполнены неверно!','Ошибка',mb\_Ok+mb\_IconHand);

 Edit1.SetFocus;

 Exit;

 end;

 snum:=n1+n2+n3+n4+n5+n6;

 try

 with Form1.Table1 do

 begin

 Active:=True;

 Append;

 FieldValues['COUNTER']:=RecordCount+1;;

 FieldValues['FIO']:=Fio;

 FieldValues['MARK1']:=n1;

 FieldValues['MARK2']:=n2;

 FieldValues['MARK3']:=n3;

 FieldValues['MARK4']:=n4;

 FieldValues['MARK5']:=n5;

 FieldValues['MARK6']:=n6;

 FieldValues['SUMMARK']:=snum;

 FieldValues['SCHOOL']:=Sch;

 Post;

 end;

 Edit1.Text:='';

 Edit2.Text:='';

 Edit3.Text:='';

 Edit4.Text:='';

 Edit5.Text:='';

 Edit6.Text:='';

 Edit7.Text:='';

 Edit8.Text:='';

 Edit1.SetFocus;

 Application.MessageBox(PChar('Добавлен участник: "'+Fio+'". Суммарный балл '+FloatToStr(snum)+'.'),'Сообщение',mb\_Ok+mb\_IconAsterisk);

 except

 Application.MessageBox('Ошибка записи участника в БД. Загрузите базу.','Ошибка',mb\_Ok+mb\_IconHand);

 Edit1.SetFocus;

 Exit;

 end;

end;

end.

**Модуль 3.**

Данный модуль отвечает за удаление участника из базы данных.

unit Unit3;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, ExtCtrls, StdCtrls, Buttons;

type

 TForm3 = class(TForm)

 Panel1: TPanel;

 Label1: TLabel;

 ComboBox1: TComboBox;

 BitBtn1: TBitBtn;

 BitBtn2: TBitBtn;

 procedure BitBtn1Click(Sender: TObject);

 procedure FormShow(Sender: TObject);

 procedure BitBtn2Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form3: TForm3;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm3.BitBtn1Click(Sender: TObject);

begin

 ModalResult:=mrCancel;

end;

procedure TForm3.FormShow(Sender: TObject);

 var i:integer;

begin

 try

 if Form1.Table1.RecordCount=0 then

 begin

 Application.MessageBox('В БД нет ни одной записи!','Ошибка',mb\_Ok+mb\_IconHand);

 ComboBox1.Color:=clMenu;

 ComboBox1.Enabled:=False;

 BitBtn2.Enabled:=False;

 end

 else

 begin

 ComboBox1.Items.Clear;

 ComboBox1.Color:=clWhite;

 ComboBox1.Enabled:=True;

 BitBtn2.Enabled:=True;

 Form1.Table1.First;

 for i:=0 to Form1.Table1.RecordCount-1 do

 begin

 ComboBox1.Items.Add(Form1.Table1.FieldValues['FIO']);

 Form1.Table1.Next;

 end;

 end;

 ComboBox1.ItemIndex:=0;

 Form1.Table1.First;

 except

 Application.MessageBox('Ошибка БД. Возможно база не открыта!','Ошибка',mb\_Ok+MB\_ICONHAND);

 ComboBox1.Color:=clMenu;

 ComboBox1.Enabled:=False;

 BitBtn2.Enabled:=False;

 end;

end;

procedure TForm3.BitBtn2Click(Sender: TObject);

 var

 i:integer;

begin

 try

 Form1.Table1.First;

 For i:=0 to Form1.Table1.RecordCount-1 do

 begin

 if ComboBox1.Text=Form1.Table1.FieldValues['FIO'] then

 begin

 Form1.Table1.Delete;

 Application.MessageBox(PChar('Запись участника "'+ComboBox1.Text+'" успешно удалена'),'Удаление',mb\_Ok+mb\_IconAsterisk);

 ModalResult:=mrCancel;

 Exit;

 end;

 Form1.Table1.Next;

 end;

 except

 Application.MessageBox('Ошибка удаления записи!','Ошибка',mb\_Ok+mb\_IconHand);

 Exit;

 end;

end;

end.

**Модуль 4.**

Код этого модуля отвечает за формирование окна конфигурации программы, а также за запись настроек этой конфигурации.

unit Unit4;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, ExtCtrls, StdCtrls, Buttons, INIFiles;

type

 TForm4 = class(TForm)

 Panel1: TPanel;

 Label1: TLabel;

 BitBtn1: TBitBtn;

 BitBtn2: TBitBtn;

 GroupBox1: TGroupBox;

 RadioGroup1: TRadioGroup;

 GroupBox2: TGroupBox;

 Label2: TLabel;

 Edit1: TEdit;

 RadioGroup2: TRadioGroup;

 procedure BitBtn1Click(Sender: TObject);

 procedure BitBtn2Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form4: TForm4;

 Conf:TIniFile;

implementation

{$R \*.dfm}

procedure TForm4.BitBtn1Click(Sender: TObject);

begin

 ModalResult:=mrCancel;

end;

procedure TForm4.BitBtn2Click(Sender: TObject);

begin

 try

 Conf:=TIniFile.Create(GetCurrentDir+'\Config.ini');

 with Conf do

 begin

 case RadioGroup1.ItemIndex of

 0: WriteString('BLOCK\_CONFIG','NUM\_BLOCKS','1');

 1: WriteString('BLOCK\_CONFIG','NUM\_BLOCKS','2');

 2: WriteString('BLOCK\_CONFIG','NUM\_BLOCKS','3');

 end;

 case RadioGroup2.ItemIndex of

 0: WriteString('SORT\_CONFIG','SORT\_TYPE','1');

 1: WriteString('SORT\_CONFIG','SORT\_TYPE','2');

 2: WriteString('SORT\_CONFIG','SORT\_TYPE','3');

 end;

 WriteString('MARK\_CONFIG','MARKS\_TYPE','ONE\_IN\_ONE');

 WriteString('MARK\_CONFIG','MARKS\_VALUE',Edit1.Text);

 UpdateFile;

 end;

 Conf.Free;

 Application.MessageBox('Настройки успешно сохранены!','Сообщение',mb\_Ok+mb\_IconAsterisk);

 ModalResult:=mrCancel;

 except

 Application.MessageBox('Ошибка записи настроек в INI-файл!','Ошибка',mb\_Ok+mb\_IconHand);

 Exit;

 end;

end;

end.

**Модуль 5.**

Код этого модуля представляет собой программный вариант распределения участников по местам, а также построение распределений по указанным параметрам. Проще говоря, модуль отвечает за окно распределения по местам и все действия, производимые в нем.

unit Unit5;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, ExtCtrls, StdCtrls, Grids, DBGrids, Buttons, DB, DBTables, INIFiles,

 TeeProcs, TeEngine, Chart, TeeFunci, Series, Menus;

type

 TForm5 = class(TForm)

 Panel1: TPanel;

 Label1: TLabel;

 DBGrid1: TDBGrid;

 GroupBox1: TGroupBox;

 BitBtn1: TBitBtn;

 Label2: TLabel;

 Label3: TLabel;

 Label4: TLabel;

 RadioGroup1: TRadioGroup;

 DataSource1: TDataSource;

 Query1: TQuery;

 PopupMenu1: TPopupMenu;

 N11: TMenuItem;

 N21: TMenuItem;

 N31: TMenuItem;

 N1: TMenuItem;

 N2: TMenuItem;

 N3: TMenuItem;

 N4: TMenuItem;

 N5: TMenuItem;

 ColorDialog1: TColorDialog;

 Chart1: TChart;

 Series1: TBarSeries;

 procedure FormShow(Sender: TObject);

 procedure BitBtn1Click(Sender: TObject);

 procedure RadioGroup1Click(Sender: TObject);

 procedure N11Click(Sender: TObject);

 procedure N3Click(Sender: TObject);

 procedure N4Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form5: TForm5;

 Conf1:TIniFile;

 SortType:String;

 NumMarks:double;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm5.FormShow(Sender: TObject);

 var

 Default, BCon, MCon:string;

 i:integer;

 blocks, param:array [1..3] of double;

begin

 NumMarks:=0;

 RadioGroup1.ItemIndex:=3;

 try

 Conf1:=TIniFile.Create(GetCurrentDir+'\Config.ini');

 with Conf1 do

 begin

 BCon:=ReadString('BLOCK\_CONFIG','NUM\_BLOCKS',Default);

 Label2.Caption:='Количество блоков задач - '+Bcon;

 MCon:=ReadString('MARK\_CONFIG','MARKS\_TYPE',Default);

 SortType:=ReadString('SORT\_CONFIG','SORT\_TYPE',Default);

 if MCon='ONE\_IN\_ONE' then

 begin

 NumMarks:=StrToFloat(ReadString('MARK\_CONFIG','MARKS\_VALUE',Default));

 Label3.Caption:='Общий балл для всех задач равен '+FloatToStr(NumMarks);

 end;

 if SortType='1' then Label4.Caption:='Слабый коллектив (1,2,3)';

 if SortType='2' then Label4.Caption:='Сильный коллектив (1,3,2)';

 if SortType='3' then Label4.Caption:='Смешанный коллектив';

 end;

 Conf1.Free;

 except

 Application.MessageBox('Ошибка чтения из INI-файла. Проверьте конфигурацию.','Ошибка',mb\_Ok+mb\_IconHand);

 Exit;

 end;

 try

 with Query1 do

 begin

 Active:=False;

 SQL.Clear;

 SQL.Add('Select \* from "'+Form1.Table1.TableName+'"');

 ExecSQL;

 end;

 Query1.Active:=True;

 Chart1.Enabled:=True;

 DBGrid1.Enabled:=True;

 RadioGroup1.Enabled:=True;

 except

 Application.MessageBox('Ошибка инициализации БД. Возможно не установлен BDE, или база не открыта.','Ошибка',mb\_Ok+MB\_ICONHAND);

 Chart1.Enabled:=False;

 DBGrid1.Enabled:=False;

 RadioGroup1.Enabled:=False;

 Exit;

 end;

 if Query1.RecordCount=0 then

 begin

 Application.MessageBox('В БД нет ни одной записи.','Сообщение',mb\_Ok+MB\_ICONASTERISK);

 RadioGroup1.Enabled:=False;

 Exit;

 end;

 try

 with Query1 do

 begin

 First;

 for i:=1 to RecordCount do

 begin

 if BCon='3' then

 begin

 blocks[1]:=FieldValues['MARK1']+FieldValues['MARK2'];

 blocks[2]:=FieldValues['MARK3']+FieldValues['MARK4'];

 blocks[3]:=FieldValues['MARK5']+FieldValues['MARK6'];

 param[1]:=(blocks[1]+blocks[2]+blocks[3])-3\*NumMarks;

 param[2]:=2\*(blocks[2]-0.5\*(blocks[1]+blocks[3]));

 param[3]:=(blocks[3]-blocks[1])+NumMarks;

 end;

 if Bcon='2' then

 begin

 blocks[1]:=FieldValues['MARK1']+FieldValues['MARK2']+FieldValues['MARK3'];

 blocks[2]:=FieldValues['MARK4']+FieldValues['MARK5']+FieldValues['MARK6'];

 blocks[3]:=0;

 param[1]:=(blocks[1]+blocks[2])-3\*NumMarks;

 param[2]:=(blocks[2]-blocks[1])+NumMarks;

 param[3]:=0;

 end;

 if Bcon='1' then

 begin

 blocks[1]:=FieldValues['MARK1']+FieldValues['MARK2']+FieldValues['MARK3']+FieldValues['MARK4']+FieldValues['MARK5']+FieldValues['MARK6'];

 blocks[2]:=0;

 blocks[3]:=0;

 param[1]:=blocks[1]-NumMarks;

 param[2]:=0;

 param[3]:=0;

 end;

 Edit;

 FieldValues['PARAM1']:=param[1];

 FieldValues['PARAM2']:=param[2];

 FieldValues['PARAM3']:=param[3];

 Next;

 end;

 end;

 except

 Application.MessageBox('Неизвестная ошибка!','Ошибка',mb\_Ok+MB\_ICONHAND);

 Exit;

 end;

 try

 with Query1 do

 begin

 Active:=False;

 SQL.Clear;

 SQL.Add('select \* from "'+Form1.Table1.TableName+'"');

 SQL.Add('order by PARAM1 desc, PARAM2 desc, PARAM3 desc');

 ExecSQL;

 end;

 DataSource1.DataSet:=Query1;

 DataSource1.Enabled:=True;

 DBGrid1.DataSource:=DataSource1;

 DBGrid1.Enabled:=True;

 Query1.Active:=True;

 except

 Application.MessageBox('Ошибка выполнения сортировки участников. Проверьте конфигурацию.','Ошибка',mb\_Ok+MB\_ICONHAND);

 Exit;

 end;

end;

procedure TForm5.BitBtn1Click(Sender: TObject);

begin

 ModalResult:=mrOk;

end;

procedure TForm5.RadioGroup1Click(Sender: TObject);

 var

 SortStr:string;

begin

 case RadioGroup1.ItemIndex of

 0: SortStr:='order by PARAM1 desc';

 1: SortStr:='order by PARAM2 desc';

 2: SortStr:='order by PARAM3 desc';

 3: SortStr:='order by PARAM1 desc,PARAM2 desc,PARAM3 desc';

 end;

 if (SortType='1') and (RadioGroup1.ItemIndex=3) then SortStr:='order by PARAM1 desc, PARAM3 desc, PARAM3 desc';

 with Query1 do

 begin

 Active:=False;

 SQL.Clear;

 SQL.Add('select \* from "'+Form1.Table1.TableName+'"');

 SQL.Add(SortStr);

 ExecSQL;

 Active:=True;

 end;

end;

procedure TForm5.N11Click(Sender: TObject);

 var

 i,j,NumI:integer;

 ind,per:double;

 GrPar, GrPercent, parm:array[1..200] of double;

begin

 NumI:=StrToInt(FloatToStr(NumMarks));

 for i:=0 to 200 do

 begin

 GrPar[i]:=0;

 GrPercent[i]:=0;

 parm[i]:=0;

 end;

 per:=0;

 ind:=0;

 Query1.First;

 if Sender=N11 then

 begin

 for j:=1 to Query1.RecordCount do

 begin

 parm[j]:=Query1.FieldValues['PARAM1'];

 Query1.Next;

 end;

 ind:=-3\*NumMarks;

 for i:=1 to 6\*NumI+1 do

 begin

 for j:=1 to Query1.RecordCount do

 begin

 if ind=parm[j] then per:=per+1;

 end;

 GrPar[i]:=ind;

 GrPercent[i]:=per/Query1.RecordCount;

 per:=0;

 ind:=ind+1;

 end;

 Series1.Clear;

 Chart1.BottomAxis.Title.Caption:='Параметр 1';

 Chart1.BottomAxis.Minimum:=-3\*NumMarks;

 Chart1.BottomAxis.Maximum:=ind-1;

 For i:=0 to 6\*NumI+1 do

 begin

 Series1.AddXY(GrPar[i],GrPercent[i]);

 end;

 end;

 if Sender=N21 then

 begin

 for j:=1 to Query1.RecordCount do

 begin

 parm[j]:=Query1.FieldValues['PARAM2'];

 Query1.Next;

 end;

 ind:=-2\*NumMarks;

 for i:=1 to 4\*NumI+1 do

 begin

 for j:=1 to Query1.RecordCount do

 begin

 if ind=parm[j] then per:=per+1;

 end;

 GrPar[i]:=ind;

 GrPercent[i]:=per/Query1.RecordCount;

 per:=0;

 ind:=ind+1;

 end;

 Series1.Clear;

 Chart1.BottomAxis.Title.Caption:='Параметр 2';

 Chart1.BottomAxis.Minimum:=-2\*NumMarks;

 Chart1.BottomAxis.Maximum:=ind-1;

 For i:=0 to 4\*NumI+1 do

 begin

 Series1.AddXY(GrPar[i],GrPercent[i]);

 end;

 end;

 if Sender=N31 then

 begin

 for j:=1 to Query1.RecordCount do

 begin

 parm[j]:=Query1.FieldValues['PARAM3'];

 Query1.Next;

 end;

 ind:=-1\*NumMarks;

 for i:=1 to 2\*NumI+1 do

 begin

 for j:=1 to Query1.RecordCount do

 begin

 if ind=parm[j] then per:=per+1;

 end;

 GrPar[i]:=ind;

 GrPercent[i]:=per/Query1.RecordCount;

 per:=0;

 ind:=ind+1;

 end;

 Series1.Clear;

 Chart1.BottomAxis.Title.Caption:='Параметр 3';

 Chart1.BottomAxis.Minimum:=-1\*NumMarks;

 Chart1.BottomAxis.Maximum:=ind-1;

 For i:=0 to 2\*NumI+1 do

 begin

 Series1.AddXY(GrPar[i],GrPercent[i]);

 end;

 end;

 if Sender=N1 then

 begin

 for j:=1 to Query1.RecordCount do

 begin

 parm[j]:=Query1.FieldValues['SUMMARK'];

 Query1.Next;

 end;

 ind:=0;

 for i:=1 to 6\*NumI+1 do

 begin

 for j:=1 to Query1.RecordCount do

 begin

 if ind=parm[j] then per:=per+1;

 end;

 GrPar[i]:=ind;

 GrPercent[i]:=per/Query1.RecordCount;

 per:=0;

 ind:=ind+1;

 end;

 Series1.Clear;

 Chart1.BottomAxis.Title.Caption:='Суммарный балл';

 Chart1.BottomAxis.Minimum:=0\*NumMarks;

 Chart1.BottomAxis.Maximum:=ind-1;

 For i:=0 to 6\*NumI+1 do

 begin

 Series1.AddXY(GrPar[i],GrPercent[i]);

 end;

 end;

 Query1.First;

end;

procedure TForm5.N3Click(Sender: TObject);

begin

 if ColorDialog1.Execute then

 begin

 Series1.SeriesColor:=ColorDialog1.Color;

 end;

end;

procedure TForm5.N4Click(Sender: TObject);

begin

 if ColorDialog1.Execute then

 begin

 Chart1.Gradient.EndColor:=ColorDialog1.Color;

 Chart1.Gradient.StartColor:=clWhite;

 end;

end;

end.

**Модуль 6.**

Код этого модуля отвечает за окно и сам процесс создания локальной базы данных.

unit Unit6;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, ExtCtrls, StdCtrls, Buttons, DB, DBTables;

type

 TForm6 = class(TForm)

 Panel1: TPanel;

 BitBtn1: TBitBtn;

 BitBtn2: TBitBtn;

 Label1: TLabel;

 Edit1: TEdit;

 Label2: TLabel;

 Edit2: TEdit;

 Label3: TLabel;

 procedure BitBtn1Click(Sender: TObject);

 procedure BitBtn2Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form6: TForm6;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm6.BitBtn1Click(Sender: TObject);

begin

 ModalResult:=mrNo;

end;

procedure TForm6.BitBtn2Click(Sender: TObject);

var

 Comm:TStringList;

begin

 try

 with Form1.Table1 do begin

 Active := False;

 DatabaseName := 'Olymp';

 TableType := ttParadox;

 TableName := Edit1.Text;

 if not Form1.Table1.Exists then begin

 with FieldDefs do begin

 Clear;

 with AddFieldDef do begin

 Name := 'Counter';

 DataType := ftInteger;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'FIO';

 DataType := ftString;

 Size := 25;

 end;

 with AddFieldDef do begin

 Name := 'MARK1';

 DataType := ftFloat;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'MARK2';

 DataType := ftFloat;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'MARK3';

 DataType := ftFloat;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'MARK4';

 DataType := ftFloat;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'MARK5';

 DataType := ftFloat;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'MARK6';

 DataType := ftFloat;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'SUMMARK';

 DataType := ftFloat;

 Required := True;

 end;

 with AddFieldDef do begin

 Name := 'SCHOOL';

 DataType := ftString;

 Size := 35;

 end;

 with AddFieldDef do begin

 Name := 'PARAM1';

 DataType := ftFloat;

 Required:=False;

 end;

 with AddFieldDef do begin

 Name := 'PARAM2';

 DataType := ftFloat;

 Required:=False;

 end;

 with AddFieldDef do begin

 Name := 'PARAM3';

 DataType := ftFloat;

 Required:=False;

 end;

 with AddFieldDef do begin

 Name := 'PLACE';

 DataType := ftInteger;

 Required:=False;

 end;

 end;

 with IndexDefs do begin

 Clear;

 with AddIndexDef do begin

 Name := '';

 Fields := 'COUNTER';

 Options := [ixPrimary];

 end;

 with AddIndexDef do begin

 Name := 'VAL1';

 Fields := 'PARAM1';

 Options := [ixDescending];

 end;

 with AddIndexDef do begin

 Name := 'VAL2';

 Fields := 'PARAM2';

 Options := [ixDescending];

 end;

 with AddIndexDef do begin

 Name := 'VAL3';

 Fields := 'PARAM3';

 Options := [ixDescending];

 end;

 end;

 CreateTable;

 Application.MessageBox('Локальная база данных успешно создана!','Сообщение',mb\_OK+mb\_IconAsterisk);

 end

 else Application.MessageBox('Локальная база данных уже существует и открыта!','Сообщение',mb\_OK+mb\_IconAsterisk);

 end;

 except

 Application.MessageBox('Ошибка создания базы данных!','Внимание',mb\_OK+mb\_IconHand);

 Exit;

 end;

 Form1.Table1.Active:=True;

 Form1.dsOlymp.DataSet:=Form1.Table1;

 Form1.DBGrid1.DataSource:=Form1.dsOlymp;

 Comm:=TStringList.Create;

 Comm.Clear;

 Comm.Add(Edit2.Text);

 Comm.SaveToFile('Bases\'+Edit1.Text+'.olp');

 Comm.Free;

 Form1.Label1.Caption:='Описание БД: '+Edit2.Text;

 ModalResult:=mrNo;

end;

end.

**Модуль 6.**

Этот модуль отвечает за весь процесс оценки качества заданий. В нем рассчитываются все параметры, и строятся необходимые диаграммы.

unit Unit7;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, ExtCtrls, StdCtrls, Buttons, Grids, DBGrids, TeEngine, Series,

 TeeProcs, Chart, DB, DBTables, INIFiles;

type

 TForm7 = class(TForm)

 Panel1: TPanel;

 BitBtn1: TBitBtn;

 Chart1: TChart;

 Series1: TBarSeries;

 DataSource1: TDataSource;

 Query1: TQuery;

 GroupBox1: TGroupBox;

 Label1: TLabel;

 Label2: TLabel;

 Label4: TLabel;

 Label5: TLabel;

 Chart2: TChart;

 Series2: TBarSeries;

 Chart3: TChart;

 Series3: TLineSeries;

 Series4: TBarSeries;

 Series5: TBarSeries;

 Label3: TLabel;

 Series6: TLineSeries;

 procedure BitBtn1Click(Sender: TObject);

 procedure FormShow(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form7: TForm7;

 Cf1:TIniFile;

implementation

uses Unit1, Unit5, foxsoft;

{$R \*.dfm}

procedure TForm7.BitBtn1Click(Sender: TObject);

begin

 ModalResult:=mrOk;

end;

procedure TForm7.FormShow(Sender: TObject);

var

 i,j,NumI,kl,kz,nz,gr:integer;

 ind,per,block1,block2,block3,coun,coun1,coun2,coun3,n1,n2,n3,k:double;

 p1,p2,p3:array [1..50] of double;

 GrPar, GrPercent, parm:array[1..200] of double;

 MCon, Default,maxb:string;

 sbl1,sbl2,sbl3:double;

 kn, kp:array[1..3] of double;

 srbl1,srbl2,srbl3,sdxq1,sdxq2,sdxq3:double;

 dx1,dx2,dx3,sigm1,sigm2,sigm3,m:double;

begin

 try

 with Query1 do

 begin

 Active:=False;

 SQL.Clear;

 SQL.Add('Select \* from "'+Form1.Table1.TableName+'"');

 ExecSQL;

 end;

 Query1.Active:=True;

 Chart1.Enabled:=True;

 except

 Application.MessageBox('Ошибка инициализации БД. Возможно не установлен BDE, или база не открыта.','Ошибка',mb\_Ok+MB\_ICONHAND);

 Chart1.Enabled:=False;

 Exit;

 end;

 if Query1.RecordCount=0 then

 begin

 Application.MessageBox('В БД нет ни одной записи.','Сообщение',mb\_Ok+MB\_ICONASTERISK);

 Exit;

 end;

 Cf1:=TIniFile.Create(GetCurrentDir+'\Config.ini');

 try

 with Cf1 do

 begin

 MCon:=ReadString('MARK\_CONFIG','MARKS\_TYPE',Default);

 if MCon='ONE\_IN\_ONE' then

 begin

 maxb:=ReadString('MARK\_CONFIG','MARKS\_VALUE',Default);

 end;

 MCon:=ReadString('BLOCK\_CONFIG','NUM\_BLOCKS',Default);

 end;

 NumI:=StrToInt(maxb);

 Cf1.Free;

 except

 Application.MessageBox('Ошибка чтения из INI-файла. Проверьте конфигурацию.','Ошибка',mb\_Ok+mb\_IconHand);

 Exit;

 end;

 for i:=0 to 200 do

 begin

 GrPar[i]:=0;

 GrPercent[i]:=0;

 parm[i]:=0;

 end;

 per:=0;

 ind:=0;

 Query1.First;

 for j:=1 to Query1.RecordCount do

 begin

 parm[j]:=Query1.FieldValues['SUMMARK'];

 Query1.Next;

 end;

 ind:=0;

 for i:=1 to 6\*NumI+1 do

 begin

 for j:=1 to Query1.RecordCount do

 begin

 if ind=parm[j] then per:=per+1;

 end;

 GrPar[i]:=ind;

 GrPercent[i]:=per/Query1.RecordCount;

 per:=0;

 ind:=ind+1;

 end;

 Series1.Clear;

 Chart1.BottomAxis.Title.Caption:='Суммарный балл';

 Chart1.BottomAxis.Minimum:=0\*NumMarks;

 Chart1.BottomAxis.Maximum:=ind-1;

 For i:=0 to 6\*NumI+1 do

 begin

 Series1.AddXY(GrPar[i],GrPercent[i]);

 end;

 if MCon='2' then

 begin

 with Query1 do

 begin

 Series2.Clear;

 Chart2.BottomAxis.Maximum:=RecordCount;

 Chart2.BottomAxis.Minimum:=0;

 n1:=0;

 k:=1;

 coun:=0;

 First;

 for i:=1 to RecordCount do

 begin

 block1:=FieldValues['MARK1']+FieldValues['MARK2']+FieldValues['MARK3'];

 block2:=FieldValues['MARK4']+FieldValues['MARK5']+FieldValues['MARK6'];

 Series2.AddXY(k,block1-block2);

 if block1-block2>=0 then coun:=coun+1;

 Next;

 k:=k+1;

 end;

 end;

 n1:=coun/Query1.RecordCount;

 Label1.Caption:='Надежность реализации - '+FloatToStr(n1\*100)+'%.';

 end;

 if MCon='3' then

 begin

 with Query1 do

 begin

 Series2.Clear;

 Series4.Clear;

 Series5.Clear;

 Chart2.BottomAxis.Maximum:=RecordCount+10;

 Chart2.BottomAxis.Minimum:=0;

 n1:=0; n2:=0; n3:=0;

 k:=1;

 coun:=0; coun2:=0; coun3:=0;

 First;

 for i:=1 to RecordCount do

 begin

 block1:=FieldValues['MARK1']+FieldValues['MARK2'];

 block2:=FieldValues['MARK3']+FieldValues['MARK4'];

 block3:=FieldValues['MARK5']+FieldValues['MARK6'];

 Series2.AddXY(k,block1-block2);

 Series4.AddXY(k,block2-block3);

 Series5.AddXY(k,block1-block3);

 if block1-block2>=0 then coun1:=coun1+1;

 if block2-block3>=0 then coun2:=coun2+1;

 if block1-block3>=0 then coun3:=coun3+1;

 Next;

 k:=k+1;

 end;

 end;

 n1:=coun1/Query1.RecordCount;

 n2:=coun2/Query1.RecordCount;

 n3:=coun3/Query1.RecordCount;

 Label1.Caption:='Надежность реализации - '+FloatToStr(Okrugl(n1\*100,2))+'%, '+FloatToStr(Okrugl(n2\*100,2))+'%, '+FloatToStr(Okrugl(n3\*100,2))+'%.';

 end;

 if MCon='1' then

 begin

 Series2.Clear;

 Series4.Clear;

 Series5.Clear;

 Label1.Caption:='Параметр не имеет смысла с 1 блоком.';

 end;

 Label3.Caption:='Количество блоков задач - '+Mcon;

 Label5.Caption:='Обшее количество участников '+IntToStr(Query1.RecordCount)+'.';

 Label4.Caption:='Максимальный балл равен '+IntToStr(NumI)+'.';

 if MCon<>'1' then

 begin

 Query1.First;

 for i:=1 to Query1.RecordCount do

 begin

 p1[i]:=Query1.FieldValues['PARAM1'];

 p2[i]:=Query1.FieldValues['PARAM2'];

 p3[i]:=Query1.FieldValues['PARAM3'];

 Query1.Next;

 end;

 kl:=0;

 j:=1;

 kz:=Query1.RecordCount;

 nz:=0;

 gr:=0;

 for i:=1 to kz do

 begin

 with Query1 do

 begin

 Active:=False;

 SQL.Clear;

 SQL.Add('Select \* from "'+Form1.Table1.TableName+'"');

 SQL.Add('where (PARAM1='+FloatToStr(p1[j])+') and (PARAM2='+FloatToStr(p2[j])+') and (PARAM3='+FloatToStr(p3[j])+')');

 ExecSQL;

 Active:=True;

 kl:=Query1.RecordCount;

 if kl=1 then j:=j+1;

 if kl>1 then

 begin

 j:=j+kl;

 nz:=nz+kl;

 gr:=gr+1;

 end;

 end;

 end;

 Label2.Caption:='Коэффициент мест - '+FloatToStr(Okrugl((kz-nz+gr)/kz,2))+'.';

 end

 else Label2.Caption:='Для одного блока не рассчитывается.';

 //This is a demo-code for KN & KP

 try

 with Query1 do

 begin

 Active:=False;

 SQL.Clear;

 SQL.Add('Select \* from "'+Form1.Table1.TableName+'"');

 ExecSQL;

 end;

 Query1.Active:=True;

 Chart1.Enabled:=True;

 except

 Application.MessageBox('Ошибка инициализации БД. Возможно не установлен BDE, или база не открыта.','Ошибка',mb\_Ok+MB\_ICONHAND);

 Chart1.Enabled:=False;

 Exit;

 end;

 sbl1:=0; srbl1:=0;

 sbl2:=0; srbl2:=0;

 sbl3:=0; srbl3:=0;

 dx1:=0; dx2:=0; dx3:=0;

 sdxq1:=0; sdxq2:=0; sdxq3:=0;

 sigm1:=0; sigm2:=0; sigm3:=0;

 Query1.First;

 if MCon='2' then

 begin

 for i:=1 to Query1.RecordCount do

 begin

 block1:=Query1.FieldValues['MARK1']+Query1.FieldValues['MARK2']+Query1.FieldValues['MARK3'];

 block2:=Query1.FieldValues['MARK4']+Query1.FieldValues['MARK5']+Query1.FieldValues['MARK6'];

 sbl1:=sbl1+block1;

 sbl2:=sbl2+block2;

 Query1.Next;

 end;

 srbl1:=sbl1/Query1.RecordCount;

 srbl2:=sbl2/Query1.RecordCount;

 Query1.First;

 for i:=1 to Query1.RecordCount do

 begin

 block1:=Query1.FieldValues['MARK1']+Query1.FieldValues['MARK2']+Query1.FieldValues['MARK3'];

 block2:=Query1.FieldValues['MARK4']+Query1.FieldValues['MARK5']+Query1.FieldValues['MARK6'];

 dx1:=dx1+sqr(block1-srbl1);

 dx2:=dx2+sqr(block2-srbl2);

 Query1.Next;

 end;

 sdxq1:=dx1/Query1.RecordCount;

 sdxq2:=dx2/Query1.RecordCount;

 sigm1:=sqrt(sdxq1);

 sigm2:=sqrt(sdxq2);

 m:=StrToFloat(maxb);

 kn[1]:=((m-srbl1)\*((m-srbl1)\*srbl1-sqr(sigm1))/(m\*sqr(sigm1)-(m-srbl1)\*srbl1))-1;

 kp[1]:=1-((srbl1\*((m-srbl1)\*srbl1-sqr(sigm1)))/(m\*sqr(sigm1)-(m-srbl1)\*srbl1));

 kn[2]:=((m-srbl2)\*((m-srbl2)\*srbl2-sqr(sigm2))/(m\*sqr(sigm2)-(m-srbl2)\*srbl2))-1;

 kp[2]:=1-((srbl2\*((m-srbl2)\*srbl2-sqr(sigm2)))/(m\*sqr(sigm2)-(m-srbl2)\*srbl2));

 Series3.Clear;

 for i:=1 to 2 do

 Series3.AddXY(kn[i],kp[i]);

 end;

 if MCon='1' then

 begin

 for i:=1 to Query1.RecordCount do

 begin

 block1:=Query1.FieldValues['SUMMARK'];

 sbl1:=sbl1+block1;

 Query1.Next;

 end;

 srbl1:=sbl1/Query1.RecordCount;

 Query1.First;

 for i:=1 to Query1.RecordCount do

 begin

 block1:=Query1.FieldValues['SUMMARK'];

 dx1:=dx1+sqr(block1-srbl1);

 Query1.Next;

 end;

 sdxq1:=dx1/Query1.RecordCount;

 sigm1:=sqrt(sdxq1);

 m:=StrToFloat(maxb);

 kn[1]:=((m-srbl1)\*((m-srbl1)\*srbl1-sqr(sigm1))/(m\*sqr(sigm1)-(m-srbl1)\*srbl1))-1;

 kp[1]:=1-((srbl1\*((m-srbl1)\*srbl1-sqr(sigm1)))/(m\*sqr(sigm1)-(m-srbl1)\*srbl1));

 Series3.AddXY(kn[1],kp[1]);

 end;

 if MCon='3' then

 begin

 for i:=1 to Query1.RecordCount do

 begin

 block1:=Query1.FieldValues['MARK1']+Query1.FieldValues['MARK2'];

 block2:=Query1.FieldValues['MARK3']+Query1.FieldValues['MARK4'];

 block3:=Query1.FieldValues['MARK5']+Query1.FieldValues['MARK6'];

 sbl1:=sbl1+block1;

 sbl2:=sbl2+block2;

 sbl3:=sbl3+block3;

 Query1.Next;

 end;

 srbl1:=sbl1/Query1.RecordCount;

 srbl2:=sbl2/Query1.RecordCount;

 srbl3:=sbl3/Query1.RecordCount;

 Query1.First;

 for i:=1 to Query1.RecordCount do

 begin

 block1:=Query1.FieldValues['MARK1']+Query1.FieldValues['MARK2'];

 block2:=Query1.FieldValues['MARK3']+Query1.FieldValues['MARK4'];

 block3:=Query1.FieldValues['MARK5']+Query1.FieldValues['MARK6'];

 dx1:=dx1+sqr(block1-srbl1);

 dx2:=dx2+sqr(block2-srbl2);

 dx3:=dx3+sqr(block3-srbl3);

 Query1.Next;

 end;

 sdxq1:=dx1/Query1.RecordCount;

 sdxq2:=dx2/Query1.RecordCount;

 sdxq3:=dx3/Query1.RecordCount;

 sigm1:=sqrt(sdxq1);

 sigm2:=sqrt(sdxq2);

 sigm3:=sqrt(sdxq3);

 m:=StrToFloat(maxb);

 kn[1]:=((m-srbl1)\*((m-srbl1)\*srbl1-sqr(sigm1))/(m\*sqr(sigm1)-(m-srbl1)\*srbl1))-1;

 kp[1]:=1-((srbl1\*((m-srbl1)\*srbl1-sqr(sigm1)))/(m\*sqr(sigm1)-(m-srbl1)\*srbl1));

 kn[2]:=((m-srbl2)\*((m-srbl2)\*srbl2-sqr(sigm2))/(m\*sqr(sigm2)-(m-srbl2)\*srbl2))-1;

 kp[2]:=1-((srbl2\*((m-srbl2)\*srbl2-sqr(sigm2)))/(m\*sqr(sigm2)-(m-srbl2)\*srbl2));

 kn[3]:=((m-srbl3)\*((m-srbl3)\*srbl3-sqr(sigm3))/(m\*sqr(sigm3)-(m-srbl3)\*srbl3))-1;

 kp[3]:=1-((srbl3\*((m-srbl3)\*srbl3-sqr(sigm3)))/(m\*sqr(sigm3)-(m-srbl3)\*srbl3));

 Series3.Clear;

 for i:=1 to 3 do

 Series3.AddXY(kn[i],kp[i]);

 end;

 m:=0;

 Series6.Clear;

 for i:=0 to 20 do

 begin

 Series6.AddXY(m,-1\*m);

 m:=m+2;

 end;

end;

end.

**Модуль 7.**

Этот модуль носит вспомогательный характер. Он необходим для правильного округления десятичных чисел с указанной степенью точности.

unit foxsoft;

interface

uses SysUtils;

 function Okrugl(cs:double;numb:integer):double;

implementation

 function Okrugl(cs:Double;numb:integer):double;

 var

 db,db1,db2:double;

 i:int64;

 ii,ink,i1:integer;

 st:string;

 begin

 db:=db-int(cs);

 ink:=1;

 for ii:=1 to numb do ink:=ink\*10;

 db1:=db\*ink;

 db2:=cs\*ink\*100;

 i:=trunc(int(db2)/100);

 i1:=Trunc(db2-i\*100);

 if i1>49 then inc(i);

 Result:=i/ink;

 end;

end.

**Литература.**

1. *Кирьяков Б. С.* Педагогическая модель интеллектуального испытания школьников. – Рязань: Изд-во «Русское слово», 2002.
2. *Кирьяков Б.* *С.* Педагогическая модель интеллектуального испытания учащихся/Вестник Рязанского государственного педагогического университета. Рязань: РГПУ, 2001.
3. *Шарапков А.* *Н., Кирьяков Б. С.* Исследование гуманности режима соревнования на олимпиадах школьников. // Рязанские физические олимпиады // Рязань: «РИНФО», 2000. Выпуск 8.
4. *Кирьяков Б.* *С.* Проблемы проведения олимпиад в условиях дифференциации уровня подготовки школьников // Рязанские физические олимпиады// Выпуск 8, – Рязань: Изд-во «РИНФО», 2000.
5. *Кирьяков Б.**С.* Параметры интеллектуального испытания учащихся на олимпиаде// Рязанские физические олимпиады// Выпуск 9, – Рязань, 2001.
6. *Лишер Р.* Delphi. Справочник. – Пер. с англ. – СПб.: «Символ-Плюс», 2001.
7. *Фаронов В.* *В.* Delphi 5. Руководство программиста. – М.: «Нолидж», 2001.
8. *Озеров В.* Delphi. Советы программистов. – СПб.: «Символ-Плюс», 2003.