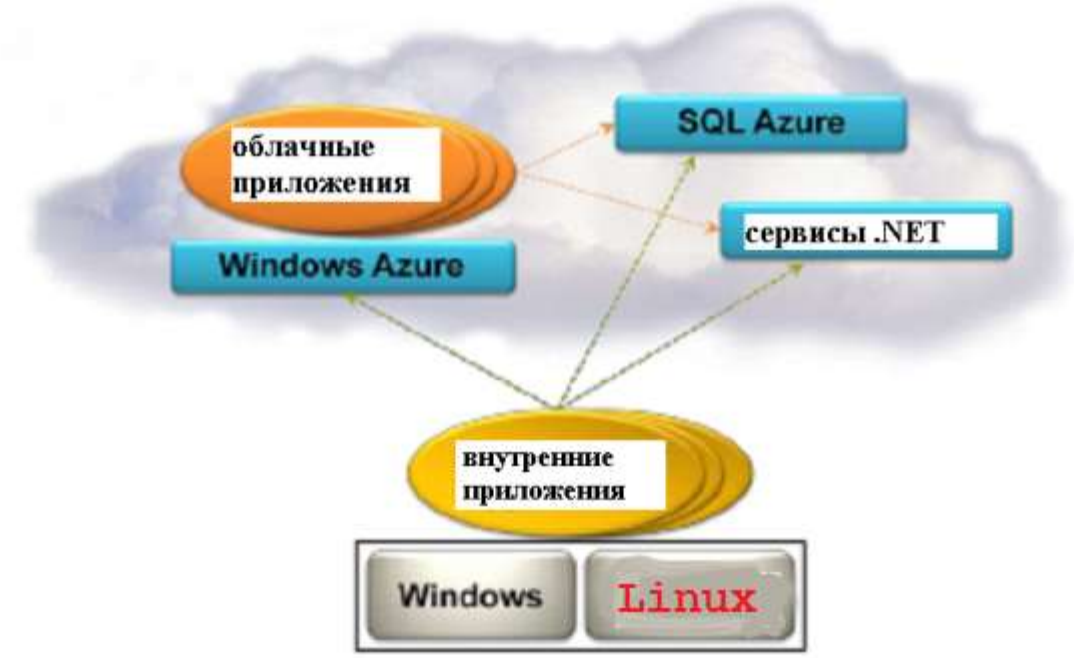
**ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**
**Северо-Кавказский филиал ордена Трудового Красного Знамени**
**федерального государственного бюджетного образовательного учреждения**
**высшего образования**
**«Московский технический университет связи и информатики»**

# Методические указания

**по выполнению курсовой работы**
**по дисциплине**

# Управление и администрирование информационных систем

направление подготовки **09.03.01. Информатика и вычислительная техника (профиль «Программное обеспечение и интеллектуальные системы»)**




**Ростов-на-Дону**
**2019**

**Методические указания**
по выполнению курсовой работы
по дисциплине

# Управление и администрирование информационных систем

Составитель: Е.О. Ткачук, доцент кафедры ИВТ
Рассмотрено и одобрено
на заседании кафедры ИВТ
Протокол от «26» августа 2019г. № 1

УДК 004.45

Ткачук Е.О.

Управление и администрирование информационных систем. Методические указания по выполнению курсовой работы. Моск. техн. ун-т связи и информатики, Сев.-Кавк. филиал. – Ростов н/Д, 2019, 44 с.

В пособии даются организационно-методические указания, варианты индивидуальных заданий теоретической и практической частей и порядок выполнения и оформления курсовой работы.

Предназначено для студентов обеих форм обучения, изучающих дисциплину «Управление и администрирование информационных систем», направление подготовки 09.03.01 «Информатика и вычислительная техника». Пособие также может быть полезно студентам других направлений подготовки, желающим самостоятельно освоить данную дисциплину.

**Содержание**

# Введение

Проблемы управления программными проектами впервые проявились в 60х начале 70х годов, когда провалились многие большие проекты по разработке программных продуктов. Были зафиксированы задержки в создании ИС, оно было ненадежным, затраты на разработку в несколько раз превосходили первоначальные оценки, созданные программные системы часто имели низкие показатели производительности. Причины провалов коренились в тех подходах, которые использовались в управлении проектами. Применяемая методика была основана на опыте управления техническими проектами и оказалась неэффективной при разработке программного обеспечения.

Имеется существенная разница между профессиональной разработкой ИС и любительским программированием. Необходимость управления программными проектами вытекает из того факта, что процесс создания профессионального ПО всегда является субъектом бюджетной политики организации, где оно разрабатывается, и имеет временные ограничения. Работа *руководителя программного проекта* по большому счету заключается в том, чтобы гарантировать выполнение этих бюджетных и временных ограничений с учетом бизнесс целей организации относительно разрабатываемой ИС.

*Руководители проектов* призваны спланировать все этапы разработки программного продукта. Они также должны контролировать ход выполнения работ и соблюдения всех требуемых стандартов. Постоянный контроль за ходом выполнения работ необходим для того, чтобы процесс разработки не выходил за временные и бюджетные ограничения. Хорошее управление не гарантирует успешного завершения проекта, но плохое управление обязательно приведет к его провалу. Это может выразиться в задержке сроков сдачи готовой ИС, в превышении сметной стоимости проекта и в несоответствии готового ПО спецификации требований.

Процесс разработки ИС существенно отличается от процессов реализации технических проектов, что порождает определенные сложности в управлении программными проектами:

1. *Программный продукт нематериален.* Программное обеспечение нематериально, его нельзя увидеть или потрогать. Руководитель программного проекта не видит процесс "роста" разрабатываемого ПО. Он может полагаться только на документацию, которая фиксирует процесс разработки программного продукта.

2. *Не существует стандартных процессов разработки ПО.* На сегодняшний день не существует четкой зависимости между процессом создания ПО и типом создаваемого программного продукта. Другие технические дисциплины имеют длительную историю, процессы разработки технических издлий многократно опробованы и проверены. Процессы создания большинства технических систем хорошо изучены. Изучением же процессов создания ПО специалисты занимаются только последнее время. Поэтому пока нельзя точно предсказать, на каком этапе процесса разработки ПО могут возникнуть проблемы, угрожающие всему программному проекту.

3. *Большие программные проекты  это часто "одноразовые" проекты.* Большие программные проекты, как правило, значительно отличаются от проектов, реализованных ранее. Поэтому, чтобы уменьшить неопределенность в планировании проекта, руководители проектов должны обладать очень большим практическим опытом. Но постоянные технологические изменения в компьютерной технике и коммуникационном оборудовании обесценивают предыдущий опыт. Знания и навыки, накопленные опытом, могут не востребоваться в новом проекте.

Перечисленные отличия могут привести к тому, что реализация проекта выйдет из временного графика или превысит бюджетные ассигнования. Программные системы зачастую оказываются новинками как в "идеологическом", так и в техническом плане. Поэтому, предвидя возможные проблемы в реализации программного проекта, следует всегда помнить, что многим из них свойственно выходить за рамки временных и бюджетных ограничений.

**Процесс управления разработкой информационной системы**

Невозможно описать и стандартизировать все работы, выполняемые в проекте по созданию ИС. Эти работы весьма существенно зависят от организации, где выполняется разработка ИС, и от типа создаваемого программного продукта. Но всегда можно выделить следующие:

Написание предложений по созданию ПО.

Планирование и составление графика работ по созданию ПО.

Оценивание стоимости проекта.

Подбор персонала.

Контроль за ходом выполнения работ.

Написание отчетов и представлений.

Первая стадия программного проекта может состоять из написания предложений по реализации этого проекта. Предложения должны содержать описание целей проектов и способов их достижения. Они также обычно включают в себя оценки финансовых и временных затрат на выполнение проекта. При необходимости здесь могут приводиться обоснования для передачи проекта на выполнение сторонней организации или команде разработчиков.

*Написание предложений* — очень ответственная работа, так как для многих организаций вопрос о том, будет ли проект выполняться самой организацией или разрабатываться по контракту сторонней компанией, является критическим. Не существует каких либо рекомендаций по написанию предложений, многое здесь зависит от опыт.

На этапе *планирования проекта* определяются процессы, этапы и полученные на каждом из них результаты, которые должны привести к выполнению проекта. Реализация этого плана приведет к достижению целей проекта. Определение стоимости проекта напрямую связано с его планированием, поскольку здесь оцениваются ресурсы, требующиеся для выполнения плана.

*Контроль за ходом выполнения работ (мониторинг проекта)* — это непрерывный процесс, продолжающийся в течение всего срока реализации проекта. Руководитель должен постоянно отслеживать ход реализации проекта и сравнивать фактические и плановые показатели выполнения работ с их стоимостью. Хотя многие организации имеют механизмы формального мониторинга работ, опыт-

ный руководитель может составить ясную картину о стадии развитии проекта просто путем неформального общения с разработчиками.

Неформальный мониторинг часто помогает обнаружить потенциальные проблемы, которые в явном виде могут обнаружиться позднее. Например, ежедневное обсуждение хода выполнения работ может выявить отдельные недоработки в создаваемом программном продукте. Вместо ожидания отчетов, в которых будет отражен факт "пробуксовки" графика работ, можно обсудить со специалистами намечающиеся программистские проблемы и не допустить срыва графика работ.

В течение реализации проекта обычно происходит несколько формальных контрольных проверок хода выполнения работ по созданию ПО. Такие проверки должны дать общую картину хода реализации проекта в целом и показать, насколько уже разработанная часть ПО соответствует целям проекта.

Время выполнения больших программных проектов может занимать несколько лет. В течение этого времени цели и намерения организации, заказавшей программный проект, могут существенно измениться. Может оказаться, что разрабатываемый программный продукт стал уже ненужным либо исходные требования к создаваемому ПО просто устарели и их необходимо кардинально менять. В такой ситуации руководство организации разработчика может принять решение о прекращении разработки ПО или об изменении проекта в целом с тем, чтобы учесть изменившиеся цели и намерения организации заказчика.

Руководители проектов обычно обязаны сами *подбирать исполнителей* для своих проектов. В идеальном случае профессиональный уровень исполнителей должен соответствовать той работе, которую они будут выполнять в ходе реализации проекта. Однако во многих случаях руководители должны полагаться на команду разработчиков, которая далека от идеальной. Такая ситуация может быть вызвана следующими причинами:

1. Бюджет проекта не позволяет привлечь высококвалифицированный персонал. В таком случае за меньшую плату привлекаются менее квалифицированные специалисты.

2. Бывают ситуации, когда невозможно найти специалистов необходимой квалификации как в самой организации разработчике, так и вне ее. Например, в организации "лучшие люди" могут быть уже заняты в других проектах.

3. Организация хочет повысить профессиональный уровень своих работников. В этом случае она может привлечь к участию в проекте неопытных или недостаточно квалифицированных работников, чтобы они приобрели необходимый опыт и поучились у более опытных специалистов.

Таким образом, почти всегда подбор специалистов для выполнения проекта имеет определенные ограничения и не является свободным. Вместе с тем необходимо, чтобы хотя бы несколько членов группы разработчиков имели квалификацию и опыт, достаточные для работы над данным проектом. В противном случае невозможно избежать ошибок в разработке ПО.

Руководитель проекта обычно обязан посылать *отчеты* о ходе его выполнения как заказчику, так и подрядным организациям. Это должны быть краткие документы, основанные на информации, извлекаемой из подробных' отчетов о проекте. В этих отчетах должна быть та информация, которая позволяет четко оценить степень готовности создаваемого программного продукта.

В рамках дисциплины «управление жизненным циклом информационных систем» выделены следующие роли в группе по разработке ПО ИС:

Руководитель – общее руководство проектом, написание документации, общение с заказчиком ИС

Системный аналитик – разработка требований (составление технического задания, проекта программного обеспечения)

Тестер – составление плана тестирования и аттестации готового ПО (продукта), составление сценария тестирования, базовый пример, проведение мероприятий по плану тестирования

Разработчик – моделирование компонент программного обеспечения, кодирование

**Планирование проекта разработки программного обеспечения**

Эффективное управление жизненным циклом информационной системы напрямую зависит от правильного планирования работ, необходимых для его вы-

полнения. План помогает руководителю предвидеть проблемы, которые могут возникнуть на каких либо этапах создания ПО, и разработать превентивные меры для их предупреждения или решения. План, разработанный на начальном этапе проекта, рассматривается всеми его участниками как руководящий документ, выполнение которого должно привести к успешному завершению проекта. Этот первоначальный план должен максимально подробно описывать все этапы реализации проекта.

Процесс планирования начинается, исходя из описания системы, с определения проектных ограничений (временные ограничения, возможности наличного персонала, бюджетные ограничения и т.д.). Эти ограничения должны определяться параллельно с оцениванием проектных параметров, таких как структура и размер проекта, а также распределением функций среди исполнителей. Затем определяются этапы разработки и то, какие результаты документация, прототипы, подсистемы или версии программного продукта) должны быть получены по окончании этих этапов. Далее начинается циклическая часть планирования. Сначала разрабатывается график работ по выполнению проекта или дается разрешение на продолжение использования ранее созданного графика. После этого проводится контроль выполнения работ и отмечаются расхождения между реальным и плановым ходом работ.

Далее, по мере поступления новой информации о ходе выполнения проекта, возможен пересмотр первоначальных оценок параметров проекта. Это, в свою очередь, может привести к изменению графика работ. Если в результате этих изменений нарушаются сроки завершения проекта, должны быть пересмотрены (и согласованы с заказчиком ПО) проектные ограничения.

Конечно, большинство руководителей проектов не думают, что реализация их проектов пройдет гладко, без всяких проблем. Желательно описать возможные проблемы еще до того, как они проявят себя в ходе выполнения проекта. Поэтому лучше составлять "пессимистические" графики работ, чем "оптимистические". Но, конечно, невозможно построить план, учитывающий все, в том числе случайные, проблемы и задержки выполнения проекта, поэтому и возникает необходи-

мость периодического пересмотра проектных ограничений и этапов создания программного продукта.

План проекта должен четко показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. В некоторых организациях план проекта составляется как единый документ, содержащий все виды планов, описанных выше. В других случаях план проекта описывает только технологический процесс создания ПО. В таком плане обязательно присутствуют ссылки на планы других видов, но они разрабатываются отдельно от плана проекта.

Детализация планов проектов очень разнится в зависимости от типа разрабатываемого программного продукта и организацииразработчика. Но в любом случае большинство планов содержат следующие разделы.

1. *Введение*. Краткое описание целей проекта и проектных ограничений (бюджетных, временных и т.д.), которые важны для управления проектом.
2. *Организация выполнения проекта*. Описание способа подбора команды разработчиков и распределение обязанностей между членами команды.
3. *Анализ рисков*. Описание возможных проектных рисков, вероятности их проявления и стратегий, направленных на их уменьшение.
4. *Аппаратные и программные ресурсы, необходимые для реализации проекта*. Перечень аппаратных средств и программного обеспечения, необходимого для разработки программного продукта. Если аппаратные средства требуется закупать, приводится их стоимость совместно с графиком закупки и поставки.
5. *Разбиение работ на этапы*. Процесс реализации проекта разбивается на отдельные процессы, определяются этапы выполнения проекта, приводится описание результатов ("выходов") каждого этапа и контрольные отметки.
6. *График работ*. В этом графике отображаются зависимости между отдельными процессами (этапами) разработки ПО, оценки времени их выполнения и распределение членов команды разработчиков по отдельным этапам.

7. *Механизмы мониторинга и контроля за ходом выполнения проекта*. Описываются предоставляемые руководителем отчеты о ходе выполнения работ, сроки их предоставления, а также механизмы мониторинга всего проекта.

План должен регулярно пересматриваться в процессе реализации проекта. Одни части плана, например график работ, изменяются часто, другие более стабильны. Для внесения изменений в план требуется специальная организация документопотока, позволяющая отслеживать эти изменения.

*Общие сведения о требованиях к информационным системам*

Проблемы, которые приходится решать специалистам в процессе создания программного обеспечения, очень сложны. Природа этих проблем не всегда ясна, особенно если разрабатываемая программная система инновационная. В частности, трудно чётко описать те действия, которые должна выполнять система. Описание функциональных возможностей и ограничений, накладываемых на систему, называется требованиями к этой системе, а сам процесс формирования, анализа, документирования и проверки этих функциональных возможностей и ограничений – разработкой требований.

Требования подразделяются на пользовательские и системные. Пользовательские требования – это описание на естественном языке (плюс поясняющие диаграммы) функций, выполняемых системой, и ограничений, накладываемых на неё. Системные требования – это описание особенностей системы (архитектура системы, требования к параметрам оборудования и т.д.), необходимых для эффективной реализации требований пользователя.

***Первые шаги по разработке требований к информационным системам анализ осуществимости***

Разработка требований — это процесс, включающий мероприятия, необходимые для создания и утверждения документа, содержащего спецификацию системных требований. Для новых программных систем процесс разработки требований должен начинаться с анализа осуществимости. Началом такого анализа является общее описание системы и ее назначения, а результатом анализа — отчет, в котором должна быть четкая рекомендация, продолжать или нет процесс разра-

ботки требований проектируемой системы. Другими словами, анализ осуществимости должен осветить следующие вопросы.

1. Отвечает ли система общим и бизнес целям организации заказчика и организации разработчика?
2. Можно ли реализовать систему, используя существующие на данный момент технологии и не выходя за пределы заданной стоимости?
3. Можно ли объединить систему с другими системами, которые уже эксплуатируются?

Критическим является вопрос, будет ли система соответствовать целям организации. Если система не соответствует этим целям, она не представляет никакой ценности для организации. В то же время многие организации разрабатывают системы, не соответствующие их целям, либо не совсем ясно понимая эти цели, либо под влиянием политических или общественных факторов.

Проблемы, которые приходится решать специалистам в процессе создания, информационных систем очень сложны. Природа этих проблем не всегда ясна, особенно если разрабатываемая программная система инновационная. В частности, трудно чётко описать те действия, которые должна выполнять система. Описание функциональных возможностей и ограничений, накладываемых на систему, называется требованиями к этой системе, а сам процесс формирования, анализа, документирования и проверки этих функциональных возможностей и ограничений – разработкой требований.

Требования подразделяются на пользовательские и системные. Пользовательские требования – это описание на естественном языке (плюс поясняющие диаграммы) функций, выполняемых системой, и ограничений, накладываемых на неё. Системные требования – это описание особенностей системы (архитектура системы, требования к параметрам оборудования и т.д.), необходимых для эффективной реализации требований пользователя.

Разработка требований

Разработка требований — это процесс, включающий мероприятия, необходимые для создания и утверждения документа, содержащего спецификацию си-

стемных требований. Различают четыре основных этапа процесса разработки требований:

1. анализ технической осуществимости создания системы,
2. формирование и анализ требований,
3. специфицирование требований и создание соответствующей документации,
4. аттестация этих требований.

Основным документом, регламентирующим взаимоотношения ИС-службы и бизнес-подразделений предприятия, является соглашение об уровне сервиса (Service Level Agreement – SLA). В данном документе дается качественное и количественное описание ИТ-сервисов, как с точки зрения службы ИС, так и с точки зрения бизнес-подразделений.

Соглашение об уровне сервиса определяет взаимные ответственности поставщика ИТ-сервиса и пользователей этого сервиса.

Типовая модель SLA должна включать следующие разделы:

- определение предоставляемого сервиса, стороны, вовлеченные в соглашение, и сроки действия соглашения;
- доступность ИТ-сервиса;
- число и размещение пользователей и/или оборудования, использующих данный ИТ-сервис;
- описание процедуры отчетов о проблемах;
- описание процедуры запросов на изменение.

Спецификации целевых уровней качества сервиса, включая:

- средняя доступность, выраженная как среднее число сбоев на период предоставления сервиса;
- минимальная доступность для каждого пользователя;
- среднее время отклика сервиса;
- максимальное время отклика для каждого пользователя;
- средняя пропускная способность;
- описания расчета приведенных выше метрик и частоты отчетов;
- описание платежей, связанных с сервисом;

- ответственности клиентов при использовании сервиса (подготовка, поддержка соответствующих конфигураций оборудования, программного обеспечения или изменения только в соответствии с процедурой изменения);
- процедура разрешения споров, связанных с предоставлением сервиса.

Существенной частью SLA является каталог сервисов. Каталог ИТ-сервисов представляет собой документ, в котором сформулированы все ИТ-сервисы, предоставляемые пользователям, при необходимости указывается цена услуги, общий порядок обращения за услугой. Каталог включает информацию описательную и операционную.

Как правило, в описывающей части содержится следующая информация:

- имя сервиса;
- ссылки на связанные сервисы;
- описание сервисов, функций, границ предоставления сервисов, профилей пользователей;
- поддерживаемые платформы или инфраструктуры;
- характеристики доступности, производительности;
- процедуры поддержки;
- метрики;
- процедуры мониторинга.

В операционной части приводят:

- имя владелец сервиса;
- профиль клиента;
- зависимости от других сервисов;
- модель Operations Level Agreement (OLA);
- детальная информация о технической инфраструктуре, необходимой для обеспечения сервиса;
- единицы инфраструктуры, рассматриваемые как активы;
- план поддержания целостности, улучшения качества сервисов, развития возможностей;
- результаты аудита;

- информация о ценах.

SLA позволяет установить формализованные критерии оценки результатов деятельности ИС-службы, установить единообразные и обязательные для всех участников процесса процедуры оценки результатов деятельности ИС-службы.

Сервисный подход к управления ИС-службой требует определенной зрелости как для самой ИС-службы, так и для бизнес-заказчиков. При этом следует учитывать ряд факторов:

- требуется определенный уровень развития управления процессами и сервисами ИТ-службы предприятия, который предполагает, что процессы и ИТ-сервисы являются измеримы;
- бизнес должен быть готов воспринимать некоторые "стандартные услуги" ИТ-службы как набор управляемых сервисов, выдвигать адекватные требования к уровню качества их предоставления, участвовать в повышении их качества;
- обеспечение прозрачности ценообразования ИТ-сервисов, при которой ИТ-служба должна обосновывать формирование цены ИТ-сервиса и возможные пути её снижения;
- наличие исключительных ситуаций, которые трудно предусмотреть заранее, процедуры выхода из них;
- процессы, люди, взгляды подвержены изменениям. SLA, как и бизнес, должен адекватно изменяться при изменении внутренних и внешних факторов.

Следует отметить, что модель ITSM может применяться для предприятий с ИТ-службами различного размера: от 1 – 5 сотрудников до нескольких десятков сотрудников.

Для малых предприятий ролевой подход, принятый в ITSM, допускает совмещение одним и тем же сотрудником сколь угодно большого количества ролей в пределах его возможностей и компетенции. В предельном случае модель ITSM может использовать ИС-служба, состоящая из одного человека. Инструментальные программные средства, которые используются для управления ИТ-инфраструктурой, могут варьироваться в широких пределах: от офисных пакетов,

в простейшем случае, до специализированных инструментальных средств при большом размере ИС-службы.

Впервые термин Совокупная стоимость владения (Total Cost of Ownership – TCO) был введен Полем Страссманом, под **TCO** он понимал денежные затраты на обслуживание, модернизацию, ремонт, приобретение новых программных продуктов (ПП) для технического средства (например, компьютера), или поддержание в рабочем состоянии ПП (например, базы данных) за все предполагаемое или фактическое время его существования.

Наиболее простым определением **TCO** ИС является следующее: это затраты, связанные с *приобретением, внедрением и использованием* ИС. При этом необходимо рассматривать *первоначальные* и *последующие* затраты, в совокупности определяя их как *единые затраты* на информационную систему в процессе ее создания и эксплуатации.

Любое предприятие с помощью автоматизации стремится повысить эффективность ведения своего бизнеса. Одно из главных условий достижения данной цели — «разумные» (т. е. не больше, но и не меньше) ИТ-затраты, которые точно так же, как и любые другие, требуют планирования, учета и контроля. Исходя из этого, для отечественных предприятий и ИТ-менеджеров неотъемлемыми и требующими детального рассмотрения являются вопросы, связанные с проблемой снижения совокупной стоимости владения информационной системой.

TCO первоначально разрабатывалась как средство расчета стоимости владения компьютером на Wintel-платформе и благодаря усилиям Gartner Group и Interpose эта методика стала основным инструментом для подсчета TCO и в других областях компьютерных технологий. Например, сейчас имеются методики расчета TCO документооборота, различных аппаратных платформ, сетей, ПО. Каждая из методик имеет свою специфику расчета, поэтому приведем лишь общую технологию расчета TCO.

Прежде всего мы попытаемся определить исходное число операторских мест и портов IVR (от англ. Interactive Voice Response – "интерактивная голосовая справка"), необходимое для того, чтобы начать работу. Для правильного расчета

нужно понять, какие задачи должен решать контакт-центр и в какой степени можно автоматизировать общение с абонентами, заменив операторов IVR.

Количество портов IVR не должно превышать число мест агентов более чем в два раза, так как в противном случае может образоваться огромная неуправляемая очередь, в которой абоненты будут долго ожидать ответа оператора.

В общем случае количество портов IVR не должно превышать число мест агентов более чем в два раза, так как в противном случае может образоваться огромная неуправляемая очередь, в которой абоненты будут долго ожидать ответа оператора. Исключением является случай, когда IVR используется в качестве автоматической информационно-справочной службы, в которой абонент может получить, например, информацию о состоянии своего лицевого счета, пополнить свой счет, послушать перечень и стоимость предоставляемых услуг и многое другое.

Предлагаемое пособие поможет качественно выполнить курсовую работу, целью которой является:

– углубить знания теории и практики управления и администрирования информационных систем;

– закрепить практические навыки написания программных кодов на языке высокого уровня Паскаль либо (по выбору студента) в среде визуального программирования Delphi, Lazarus или C++, на которых написаны большая часть программных кодов современных ОС.

# 1 Организационно-методические указания по выполнению курсовой работы

1. Курсовая работа оформляется с помощью текстового процессора MS Word на листах формата А4, шрифт 12, одинарный интервал, поля: сверху, снизу, справа – 15 мм, слева – 25 мм.

2. В записке должны быть приведены все использованные таблицы, схемы алгоритмов и иллюстрации.

3. Пояснительная записка должна иметь вид законченного документа и включать следующие обязательные пункты:

3.1. Титульный лист.

3.2. Лист утвержденного индивидуального задания на курсовую работу.

3.3. Содержание.

3.4.Теоретическая часть - обзорный – реферат на тему индивидуального задания согласно табл.1. по дисциплине «Управление и администрирование информационных систем». Вариант задания – номер студента по журналу группы. Студент должен самостоятельно выбрать литературу, использовать несколько источников, выделить основное и законспектировать. Не допускается дословное копирование текста из учебников и интернет – источников. Уровень оригинальности, полученный при проверке на сайте https://www.antiplagiat.ru/ должен быть не менее 60%. Объем – 10-15 листов формата А4, включая рисунки, графики и таблицы. Список использованной литературы должен включать в себя не менее 10 источников не старше 5 лет, и быть оформлен в соответствии с ГОСТ 7.1 – 2003, .

3.5. Практическая часть - разработка компьютерного теста контроля знаний в формате GIFT по одному из разделов изучаемой дисциплины в соответствии с индивидуальным заданием согласно табл.1, а также рабочего программного модуля тестирующей программы. Тестирующая программа должна использовать в качестве исходных данных файл тестовых вопросов в формате GIFT, загружаемый специальной директивой с использованием пароля доступа. позволяющую в режиме диалога проверить знания обучаемого.

3.6. Объем пояснительной записки к практической части – 15-20 печатных листов формата А4. Она должна включать следующие пункты:

3.6.1. постановка задачи и сравнительная характеристика существующих методов построения тестирующих программ;

3.6.2. обоснованный (исходя из возможностей выбранного средства программирования) выбор метода построения диалога;

3.6.3. исходные данные (тематика тестов, перечень проверяемых тем, вопросов и ответов);

3.6.4. разработка общего сценария тест-программы;

3.6.5. схемы алгоритмов функционирования программы;

3.6.6. текст программного модуля;

3.6.7. руководство программиста и руководство пользователя;

3.6.8. рекомендации по дальнейшему совершенствованию интерфейса и тест-программы в целом;

3.7. заключение (с необходимыми выводами);

3.8. список используемой литературы.

4. При защите курсовой работы студент демонстрируется функционирование разработанной программы путем проведения контрольного тестирования.

## 1.1 Особенности выполнения практической части курсовой работы

При разработке тестирующей программы студент самостоятельно выбирает уровень сложности разрабатываемой программы. Требования к программе, выполнение которых при условии успешной защиты курсовой работы позволит студенту получить оценку «Удовлетворительно», «Хорошо» или «Отлично» приводятся ниже.

***Минимальные требования для получения оценки «Удовлетворительно»***

Используя возможности выбранного средства программирования разработать тестирующую программу, позволяющую в режиме диалога проверить знания обучаемого по одному из разделов изучаемых на кафедре дисциплин (в соответствии с вариантом индивидуального задания) с выставлением оценки. Формат файла вопросов – GIFT, кодировка UTF-8. Количество вопросов – не менее 40, из

которых (по случайному закону) для тестирования выбираются 10 вопросов. В состав вопросов должны входить вопросы следующих типов:

- Множественный выбор – 15шт.;
- Краткий ответ – 10 шт.;
- Верно-неверно – 5 шт.;
- На соответствие – 5 шт.;
- Числовой ответ – 5 шт..

В вопросах типа «Множественный выбор» правильный ответ выбирается из 4 – 5 вариантов ответов, при этом неправильные ответы не должны явно или радикально отличаться от правильных. После выбора ответа – автоматический переход к следующему вопросу без указания, правильно или нет ответил тестируемый. В конце на мониторе высвечивается оценка с комментарием, сколько процентов составили правильные ответы и на какие вопросы (номера) даны неверные ответы. Критерии выставления итоговой оценки:

– Отлично – более чем на 85% вопросов получены правильные ответы;

– Хорошо – более чем на 70% вопросов получены правильные ответы;

– Удовлетворительно – более чем на 55% вопросов получены правильные ответы;

– Неудовлетворительно – более чем на 45% вопросов получены неправильные ответы.

***Дополнительные требования для получения оценки «Хорошо»***

Использовать текстовые и графические возможности выбранного средства программирования. Количество вопросов – не менее 50, из которых (по случайному закону) для тестирования выбираются 15 вопросов. В конце, помимо вышеуказанного, высвечиваются правильные ответы на те вопросы, по которым тестируемый дал неверный ответ, высвечивается оценка с комментарием, сколько процентов составили правильные ответы и на какие вопросы (номера) даны неверные ответы.

***Дополнительные требования для получения оценки «Отлично»***

Использовать текстовые, графические и аудио возможности выбранного средства программирования. Количество вопросов – не менее 60, из которых (по

случайному закону) для тестирования выбираются 20 вопросов. Ответ оценивается по дифференцированной 10-ти бальной шкале в зависимости от сложности вопроса, при неправильном ответе начисляется ноль баллов.

В конце высвечиваются:

- максимально возможная сумма баллов;
- набранная тестируемым сумма баллов и ее % от максимума;
- правильные ответы на те вопросы, по которым тестируемый дал неверный ответ;
- рекомендации по самостоятельному изучению материала с указанием литературных источников вплоть до страниц;
- итоговая оценка за тест;

***Описание формата GIFT***

Формат GIFT - это наиболее подходящий формат для экспорта текстовых вопросов в текстовый файл. Он разработан для облегчения создания вопросов. GIFT поддерживает вопросы множественного выбора, верно/неверно, краткий ответ, вопросы на соответствие, численные вопросы и вопросы с пропущенными словами. Вопросы различных типов могут быть совмещены в одном файле, формат также поддерживает названия вопросов, комментарии к вариантам ответов, отзыв и процентное оценивание.

**Правила создания файла в формате GIFT**

- Любой файл GIFT должен быть закодирован в формате UTF-8
- В одном файле могут содержаться любые типы вопросов (множественный выбор сопоставление и т.д)
- Между собой вопросы разделяются как минимум одной пустой строкой. Пустая строка определяет что начался новый вопрос.

Специальные символы ~ = # { } : контролируются фильтром и не могут быть использованы в тесте вопроса. Они участвуют в разделении частей вопроса, и называются "Символы управления." Но иногда вам приходится использовать эти символы в тексте вопроса, например, в математических формулах. Путь для решения таких проблем - "пропуск" символов управления. Он заключается в том,

что перед символом управления необходимо поставить обратный слеш "\". При обработке вопроса обратный слеш удаляется и не отображается.

Таблица специальных символов формата GIFT

| **Symbols** | **Use** |
|---|---|
| // text | Комментарий в конца строки (пометка на память для себя, нигде не появится, опционально) |
| ::title:: | Заголовок вопроса (опционально) |
| text | Текст вопроса (станет названием вопроса, если название не указано) |
| […format…] | Формат куска текста. Варианты [HTML-код], [обычный] и [уценка].. |
| { | Фигурная скобка указывает что начался блок с ответами |
| } | Конец блока ответов |
| {T} or {F} | True или false ответа; используется в типе вопроса верно/неверно |
| { … =right … } | Правильный ответ на множественный выбор внутри блока вариантов ответа (=правильный ответ) |
| { … ~wrong … } | Правильный ответ на множественный выбор внутри блока вариантов ответа (~неправильный ответ) |
| { … =item -> match … } | Ответ на вопросы соответствия |
| #feedback text | Ответ обратной связи на соответствие, с заполнением пустых полей или числовые ответы |
| ####general feedback | Общие отзывы |
| {# | Начало цифрового ответа |
| answer:tolerance | Числовой ответ принимается в пределах ± допустимых пределах |
| low..high | Нижнее и верхнее значения диапазона принимаемых числовых ответов |
| =%n%answer:tolerance | N процент кредита на один из нескольких числовых диапазонов в пределах допуска от ответа |

| | |
|---|---|
| \character | Обратный слеш отменяет действие символов ~, =, #, {, }, and : |
| \n | Помещает строку в текст вопроса — символ заменяет пустую стороку между вопросами |

**Описание типов вопросов**

***Множественный выбор***

**Описание:** Студент выбирает ответ на вопрос из нескольких предложенных ему вариантов, причём вопросы могут предполагать один или сразу несколько правильных ответов.

Для вопросов типа "Множественный выбор" неправильные варианты ответов начинаются со знака тильды(~), правильные - знаком равенства (=).

Для удобства варианты ответов могут быть написаны каждый с новой строки.

ПРИМЕР

---

::Вопрос 1::Кто является главным героем сказки «Айболит»?

{

~Синяя Борода

~Джефферсон

=Айболит

~Конёк-Горбунок

}

---

Можно использовать формат пропущенного слова, для этого в текст вопроса автоматически должна вставляться  линия пропущенного слова (_____), а варианты ответов показываются ниже

ПРИМЕР

::Вопрос 1::Ленин {~похоронен =родился ~живёт} в Симбирске

---

Для множественного выбора могут заданы веса после тильды (~). Значение веса с обоих сторон заключается в знак % (например, %50%). Эта опция может быть скомбинирована с комментариями на варианты ответов.

ПРИМЕР

---

::Вопрос 1::Какие цвета используются в светофоре?

{

~синий

~%50%красный

~%50%желтый

~фиолетовый

~белый

}

---

***Краткий ответ***

Ответом на вопрос является слово или короткая фраза, допускается несколько правильных ответов с различными оценками. Ответы оцениваются путем сравнения с разными образцами ответов, в которых могут использоваться подстановочные знаки.

Ответы в вопросе "Краткий ответ" начинаются знаком равенства(=), показывающим правильный ответ. Ответы не должны содержать тильду.

Если в тексте вопроса использовать комбинацию из _ (нижней линии), то поле ввода ответа появится сразу в тексте. Длина поля зависит от количества _

ПРИМЕР

---

::Вопрос 1::Кто является героем сказки «Айблит»?{

=Айболит

=Бармалей

}

---

::Вопрос 1::Ленин _____ в Симбирске{

=родился

}

---

***Верно/неверно***

При ответе на вопрос, студент выбирает между двумя вариантами «Верно» и «Неверно».

Ответ должен быть написан как {TRUE} или {FALSE}, или сокращённо {T} или {F}.

ПРИМЕР

::Вопрос 1::Ленин родился в Симбирске{TRUE}

***На соответствие***

Каждому элементу ответов первой группы нужно сопоставить элемент ответов второй группы.

Вопросы на соответствие не поддерживают процентное оценивание.

Совпадающие пары начинаются знаком (=) и разделяются знаком "->".

Должны быть как минимум три совпадающие пары.

::Название вопроса:: Укажите …{

=подвопрос1 -> подответ1

=подвопрос2 -> подответ2

=подвопрос3 -> подответ3

ПРИМЕР

---

::Вопрос 1:: Укажите столицы государств: {

=Канада -> Оттава

=Италия -> Рим

=Япония -> Токио

=Индия -> Нью Дели

}

***Числовой ответ***

Так же, что и краткий ответ, только на выполнение вычислительных операций, числовой ответ может иметь заданный интервал предельно допустимой погрешности отклонения от правильного значения.

Секция ответа в числовом вопросе должна начинаться с решетки (#). Числовой ответ может включать погрешность, которая пишется после правильного ответа и отделяется двоеточием. Например, если правильный ответ находится в диапазоне от 1.5 до 2.5, тогда вопрос должен быть написан так: {#2:0.5}. Эта запись показывает что 2 с допуском 0.5 - правильный ответ (т.е. диапазон от 1.5 до 2.5). Если погрешность не определена, то по умолчанию она устанавливается в ноль.

ПРИМЕР

---

::Вопрос 1:: В каком году родился Пушкин А.С.?
{#1799}

::Вопрос 2::Значение числа Пи (4 цифры после запятой)?
{#3.1415:0.0005}

# 2 Задания на курсовую работу

Таблица 1. Индивидуальные задания для курсовой работы

| **Вариант №** | **Содержание задания №1 курсовой работы** |
|---|---|
| 1 | CASE-технологий разработки информационной системы |
| 2 | Фазы развития информационных систем.. |
| 3 | Концептуальная фаза жизненного цикла информационной системы |
| 4 | Подготовка технического предложения. |
| 5 | Проектирование информационной системы. |
| 6 | Разработка технического задания информационной системы. |
| 7 | Ввод системы в эксплуатацию. |
| 8 | Основные процессы жизненного цикла: разработка, эксплуатация, сопровождение. |
| 9 | Вспомогательные процессы жизненного цикла. |
| 10 | Организационные процессы. |
| 11 | Модели жизненного цикла информационной системы. |
| 12 | Структура жизненного цикла информационной системы. |
| 13 | Каскадная модель жизненного цикла системы: основные этапы разработки |
| 14 | Сервис ИТ в деятельности службы ИС. |
| 15 | Функциональные области управления службой ИС. |
| 16 | Организационная структура службы ИС. Плоская структура службы ИС |
| 17 | Организационная структура службы ИС. Развернутая структура службы ИС. |
| 18 | Организационная структура службы ИС. Дивизиональная структура службы ИС. |
| 19 | Функции службы ИС и параметры сервиса ИТ. |
| 20 | Процессы службы ИС и преодоление ограничений функционального подхода. |
| 21 | ITIL/ITSM − концептуальная основа процессов службы ИС. Проект ITIL |
| 22 | Модели управления сетевыми ресурсами |
| 23 | Служба Active Directory Service |
| 24 | Системы автоматизации учета заявок и поддержки пользователей информационных систем |
| 25 | Администрирование пользователей. Политики безопасностей, их реализация в операционных системах |
| 26 | Методы анализа бизнес-плана проекта по разработке информационной системы в среде Project Expert |
| 27 | Классификация и сравнительный анализ программных средств оценки и учета затрат на разработку программного обеспечения |
| 28 | Администрирование сервера БД. Стратегии резервного копирования |

| | |
|---|---|
| 29 | Обеспечение целостности данных. Резервное копирование и восстановление данных. Стратегии резервного копирования |
| 30 | Оценка совокупной стоимостью владения и ее применение для управления информационной службой |
| **Вариант №** | **Содержание задания №2 курсовой работы (тематика разрабатываемого теста)** |
| 1 | Операционная система Windows. Интерфейс пользователя. Управление Windows. Рабочий стол, окна Windows, файловая система, специальные папки Рабочего стола. |
| 2 | Операционная система Windows. Панель задач. Диалоговые окна. Контекстное меню. Справочная система. Корзина. Подготовка к выключению компьютера. Действия при «зависании» компьютера. |
| 3 | Операционная система Windows. Проводник Windows. Основные операции с объектами. Запуск программ. Создание ярлыка. Изменение значка ярлыка. |
| 4 | Операционная система Windows. Форматирование дисков. Настройка экрана. Панель управления Настройка панели задач. Поиск файлов. |
| 5 | Текстовый процессор MSWord. Работа с окнами. Курсор ввода. Панели инструментов. Координатные линейки. Строка состояния. Режимы отображения документа. Полосы прокрутки. Выход из Microsoft Word. |
| 6 | Текстовый процессор MSWord. Создание нового документа. Открытие документа. Со-хранение документа. Закрытие документа. Ввод текста. Выделение фрагмента текста. |
| 7 | Текстовый процессор MSWord. Редактирование текста. Отмена операций над текстом. Копирование текста. Перемещение текста. Вставка символа. Поиск и замена текста. Контекстное меню. |
| 8 | Текстовый процессор MSWord. Форматирование текста. Изменение параметров шри-фта. Изменение интервала и положения символов. Форматирование абзацев. Установление позиций табуляции. |
| 9 | Текстовый процессор MSWord. Упорядочение списков. Стили форматирования. Оформление страниц документа. Установление параметров страницы. Вставка разрывов страниц. Нумерация страниц. |
| 10 | Текстовый процессор MSWord. Установление колонтитулов. Создание многоколонного документа. Печать документов. |
| 11 | Текстовый процессор MSWord. Структура таблицы Word произвольной конфигурации. Способы создания таблиц. Форматирование таблицы. Использование в таблице формул. Изменение структуры таблицы. |
| 12 | Создание составных (интегрированных) документов. Понятие составного документа. Создание составного документа через буфер обмена по технологии OLE. Внедрение и связывание объектов. |
| 13 | Текстовый процессор MSWord. Вставка графических объектов (рисунков, файлов). Использование автофигур, объектов WordArt. Группирование объектов и настройка изображений. |
| 14 | Табличный процессор MS Excel. Основные понятия интерфейс табличного процессора. Данные, хранимые в ячейках электронной таблицы. Относительные и абсолютные ссылки. |
| 15 | Табличный процессор MS Excel. Функциональные возможности табличных процес- |

| | |
|---|---|
| | соров. Типовые операции с рабочими книгами и листами. Типовые операции с блоками ячеек. |
| 16 | Табличный процессор MS Excel. Сортировка и фильтрация данных. Понятие о списке (базе данных Excel). Сортировка данных в списке. Автофильтрация и расширенный фильтр. |
| 17 | Табличный процессор MS Excel. Структурирование таблиц и создание сводных таблиц. Автоструктурирование таблиц. Создание сводных таблиц с помощь. Мастера сводных таблиц. |
| 18 | Табличный процессор MS Excel. Технология работы в электронной таблице. Проектирование электронной таблицы. Объединение электронных таблиц. Макросы как средство автоматизации работы. |
| 19 | Табличный процессор MS Excel. создание и заполнение таблицы постоянными данными и формулами Характеристика режимов и команд. Графические возможности MS Excel. |
| 20 | Табличный процессор MS Excel. Построение, редактирование и форматирование диаграмм. Построение тренда. Операции с листами рабочих книг. |
| 21 | Табличный процессор MS Excel. Организация расчётов в электронной таблице. Работа с формулами. Функции Мастер функций. Массивы формул. |
| 22 | Табличный процессор MS Excel. Операции с элементами таблицы. Вставка и удаление элементов таблицы. Копирование и перемещение данных. Поиск данных. Замена данных. |
| 23 | Табличный процессор MS Excel. Формат чисел. Маски форматов. Выравнивание содержимого ячеек. Установление шрифта. Изменение размеров строк и столбцов. Оформление таблиц. |
| 24 | Табличный процессор MS Excel. Группирование элементов таблицы. Работа с окнами. Разделение окон. Создание нового окна. Фиксация подокон. Присвоение имени ячейке, диапазону или формуле. Создание примечаний. |
| 25 | Табличный процессор MS Excel. Работа с базами данных. Сортировка данных. Формы данных. Фильтрация (выборка) данных. Установление диапазона критериев. Авто-фильтр. Расширенный фильтр. |
| 26 | Язык программирования Pascal. Алфавит и базовые типы данных. Структура программы. Стандартные функции. Запись аналитических зависимостей. Константы и переменные. Ввод-вывод данных. |
| 27 | Основные понятия алгоритмического языка. Оператор перехода. Условный оператор. Оператор варианта. Элементы структурного программирования. |
| 28 | Язык программирования Pascal. Цикл с предусловием, цикл с постусловием, цикл с параметром. Операторы завершения цикла. Массивы, строки. Процедуры и функции. |
| 29 | Язык программирования Pascal. Перечисляемый тип данных. Интервальный тип данных. Модули. Множества, Записи. Файлы: текстовые файлы, компонентные файлы, бестиповые файлы. |
| 30 | Язык программирования Pascal. Указатели. Динамические переменные. Динамические структуры данных. Очереди. Линейные списки. |

## 3 Пример оформления пояснительной записки

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**
**СЕВЕРО-КАВКАЗСКИЙ ФИЛИАЛ ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»**

# Курсовая работа

**по дисциплине**

# Управление и администрирование информационных систем

**Вариант № XX**

**Выполнил студент**
**группы ВМ-41**
**Иванченко И.И..,**
**Шифр XXXXX**
**Проверил: канд.техн.наук**
**С.н.с. ТКАЧУК Е.О.**

**Работа защищена**
**с оценкой «_____»**
**«___» _____ 2016 г.**

**Е.О. Ткачук**

**Ростов-на-Дону**
**2016**

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**
**СЕВЕРО-КАВКАЗСКИЙ ФИЛИАЛ ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»**

**«У Т В Е Р Ж Д А Ю»**
Заведующий кафедрой ИВТ

Проф. _____С.В. Соколов

**З А Д А Н И Е**
**на курсовую работу по дисциплине «Управление и администрирование информационных систем»**

студент(у, ке) Иванченко И.И.
группы № , шифр
Руководитель к.т.н., доцент Ткачук Е.О.

Вариант № XX

**Техническое задание (минимальные требования)**

1. Теоретическая часть. Выполнить реферат на тему «Передача голоса по пакетным сетям, предпосылки появления пакетной телефонии (VoIP), особенности пакетного преобразования голоса».
2. Практическая часть. Тема теста «Общие принципы и средства построения компьютерных сетей». Подготовить не менее 40 вопросов в формате GIFT, В состав вопросов должны входить вопросы следующих типов: Множественный выбор – 15шт.; Краткий ответ – 10 шт.; Верно-неверно – 5 шт.; На соответствие – 5 шт.; Числовой ответ – 5 шт..
3. В вопросах типа «Множественный выбор» правильный ответ выбирается из 4 – 5 вариантов ответов, при этом неправильные ответы не должны явно или радикально отличаться от правильных. Для тестирования по случайному закону выбирается 10 вопросов, за правильный ответ начисляются баллы (от 1 до 5 в зависимости от сложности вопроса), в конце выставляется оценка:
   - Отлично – если набрано более 85% от максимально возможной суммы баллов;
   - Хорошо – если набрано от 70% до 85% от максимально возможной суммы баллов;
   - Удовлетворительно – если набрано от 55% до 70% от максимальной суммы баллов;
   - Неудовлетворительно – если набрано менее 55% от максимальной суммы баллов.
4. Содержание практической части работы:

4.1. Постановка задачи и сравнительная характеристика существующих методов построения тестирующих программ;

4.2. Обоснованный выбор метода построения тестов;

4.3. Разработка общего сценария тест-программы;

4.4. Разработка схемы алгоритма функционирования программы;

4.5. Текст программного модуля на языке PASCAL;

4.6. Руководства для пользователя и программиста;

4.7. Заключение (с необходимыми выводами) и список использованной литературы.

5. При защите курсовой работы исполнитель демонстрирует функционирование разработанной программы путем проведения контрольного тестирования.

6. Объем пояснительной записки 25-35 страниц печатного текста формат А4, размер шрифта 12.

Задание выдано «_ __» _____ 201__ года.

Исполнитель курсовой работы _____ Иванченко И.И.

Руководитель _____ Ткачук Е.О.

**Содержание**

## 3.1 Теоретическая часть
## Управление физической памятью. Иерархия памяти. Распределение памяти. Уплотнение памяти. Стратегии размещения информации в памяти.

**Далее на 10 – 15 листах излагается теоретический материал по заданной теме с необходимыми таблицами, рисунками и графиками.**

## 3.2 Практическая часть

### 3.2.1 Постановка задачи

Использовать текстовые, графические и аудио возможности языка Паскаль либо системы визуального программирования Delphi или ее СПО-аналога Lazarus. Количество вопросов – не менее 40, из которых (по случайному закону) для тестирования выбираются 20 вопросов. Формат файла вопросов – GIFT, кодировка UTF-8. Количество вопросов – не менее 40, из которых (по случайному закону) для тестирования выбираются 10 вопросов. В состав вопросов должны входить вопросы следующих типов:

- Множественный выбор – 15шт.;
- Краткий ответ – 10 шт.;
- Верно-неверно – 5 шт.;
- На соответствие – 5 шт.;
- Числовой ответ – 5 шт..

В вопросах типа «Множественный выбор» правильный ответ выбирается из 4 – 5 вариантов ответов, при этом неправильные ответы не должны явно или радикально отличаться от правильных. После выбора ответа – автоматический переход к следующему вопросу без указания, правильно или нет ответил тестируемый. В конце на мониторе высвечивается оценка с комментарием, сколько процентов составили правильные ответы и на какие вопросы (номера) даны неверные ответы. Ответ оценивается по дифференцированной 10-ти бальной шкале в зависимости от сложности вопроса, при неправильном ответе начисляется ноль баллов. Возможно наличие на один вопрос нескольких правильных ответов, имеющих разный «вес» в баллах в зависимости от их рациональности. В конце высвечиваются:

– максимально возможная сумма баллов;
– набранная тестируемым сумма баллов и ее % от максимума;
– правильные ответы на те вопросы, по которым тестируемый дал неверный ответ;
– рекомендации по самостоятельному изучению материала с указанием литературных источников вплоть до страниц;
– итоговая оценка за тест

Критерии выставления итоговой оценки:

– Отлично – набрано более 85% баллов от максимально возможной суммы;
– Хорошо – набрано более 70% баллов от максимально возможной суммы;
– Удовлетворительно – набрано более 55% баллов от максимально возможной суммы;
– Неудовлетворительно – набрано менее 55% баллов от максимальной суммы.

### 3.2.2 Исходные данные

Назначение программы – проверка знаний обучающегося по одному из разделов изучаемых на кафедре дисциплин. В соответствии с вариантом индивидуального задания, моя тест-программа предназначена для проверки знаний обучающегося в области текстового процессора Word

#### 3.2.2.1 Перечень проверяемых тем

<<<Здесь приводится перечень проверяемых тем>>>

#### 3.2.2.2 Перечень вопросов

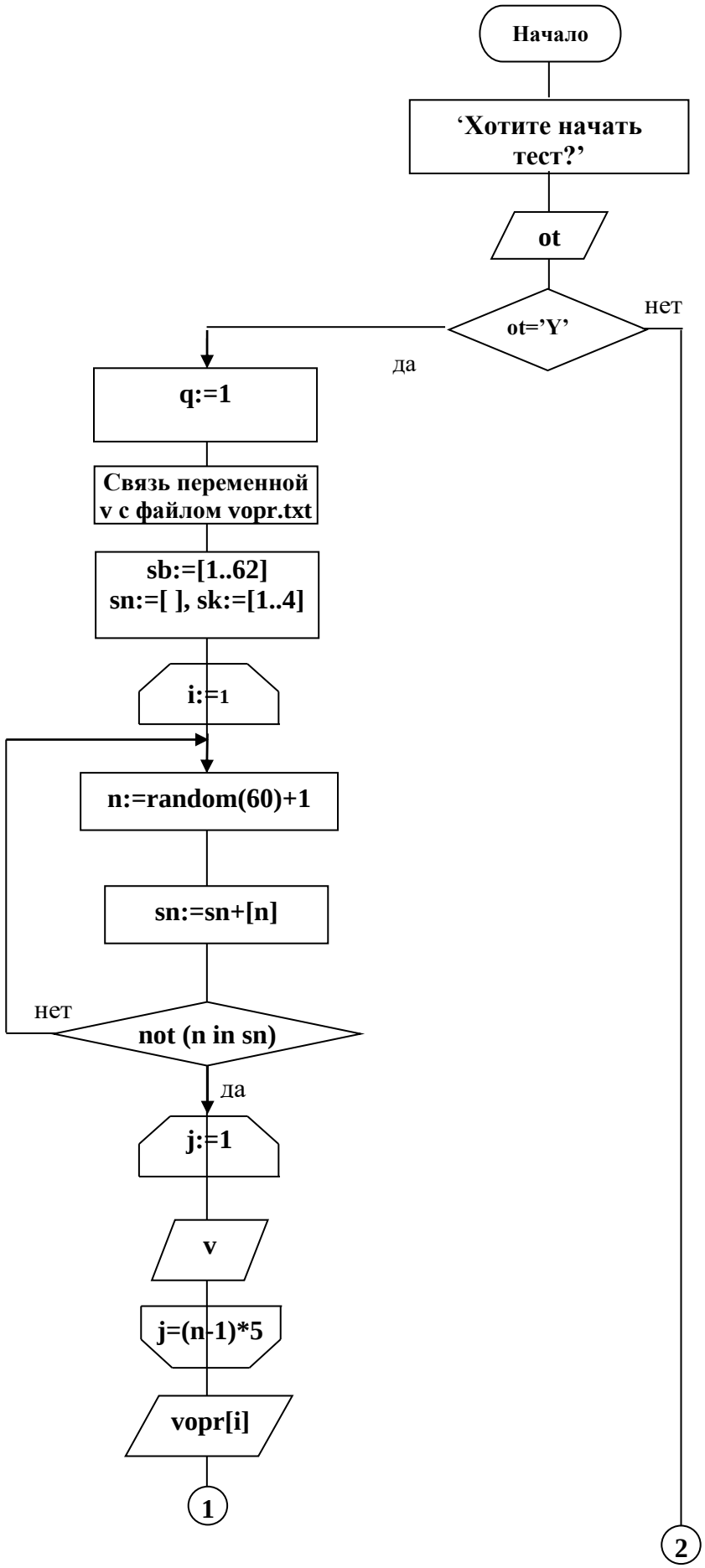
<<<Здесь приводится перечень вопросов в формате GIFT>>>

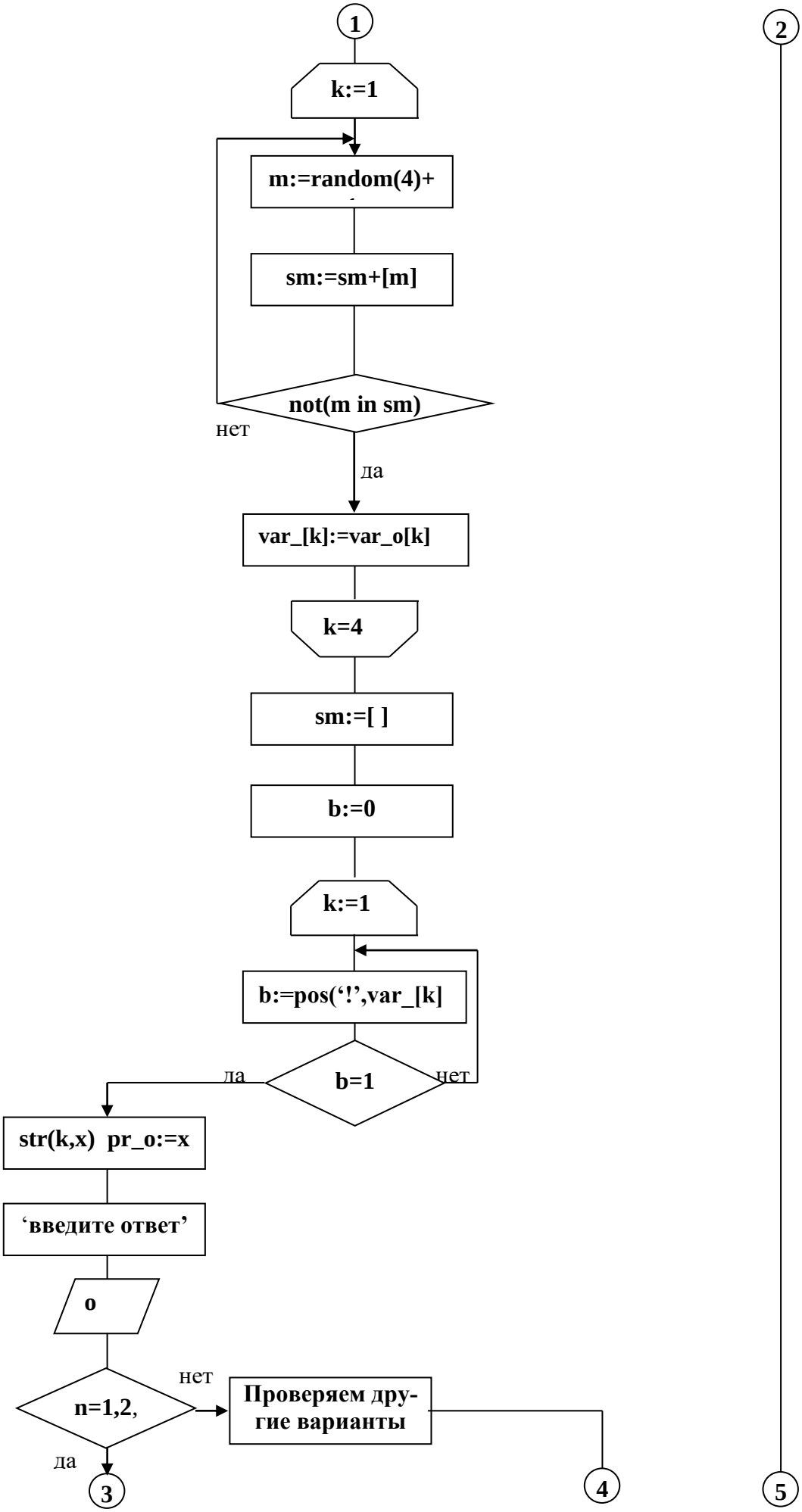
### 3.2.4 Разработка общего сценария тест-программы

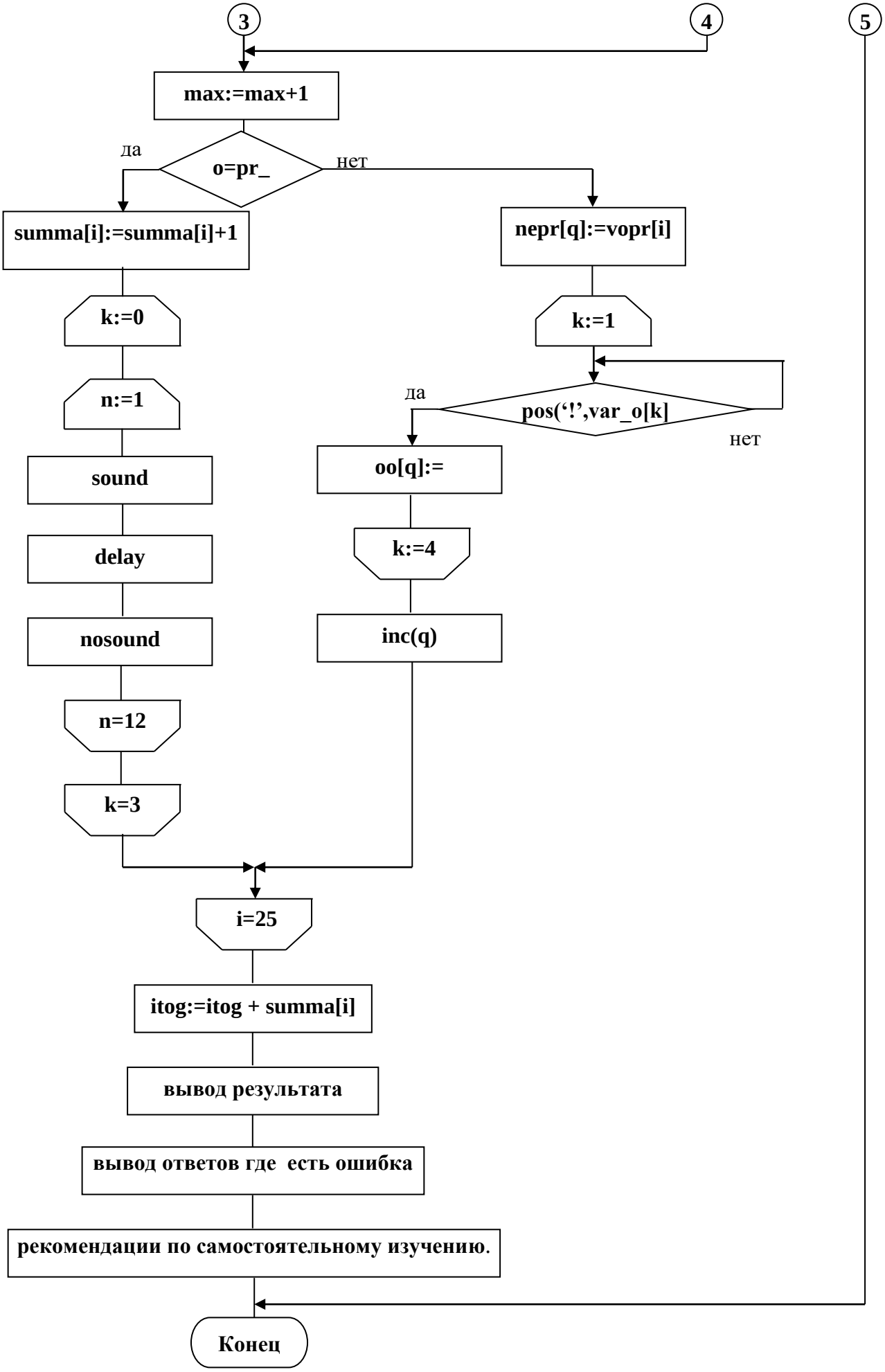
Написание подобной программы мне представляется следующим образом.

1. Вначале создается текстовый файл в кодировке UTF-8 в формате GIFT, содержащий в себе не менее 60 вопросов.
2. Далее случайным образом будем считывать по одному вопросу и заносить его в массив с данными типа string (vopr[i]);
3. Считываем в отдельный массив варианты ответа на вопрос и выдаем их на экран в произвольном порядке (var_o[j]);
4. Ищем правильный ответ: если вариант ответа помечен !, то переменной (pr_o) типа string присваиваем номер правильного ответа (переведя его из числового значения в строку);
5. Сравниваем введенный пользователем ответ (o) с правильным, если они совпадают, начисляем нужный балл, если нет – ноль;
6. Если ответ, данный пользователем, неверный, заносим в новый массив (nepr[q]) вопрос и правильный ответ (oo[q]);
7. Выдаем результаты тестирования:
   - максимально возможную сумму баллов;
   - набранную тестируемым сумму баллов и её % от максимума;
   - правильные ответы на те вопросы, по которым тестируемый дал неверный ответ;
   - рекомендации по самостоятельному изучению материала с указанием литературных источников вплоть до страниц;
   - итоговую оценку за тест.

### 3.2.5 Схема алгоритма функционирования программы




1
2
k:=1
m:=random(4)+
sm:=sm+[m]
not(m in sm)
нет
да
var_[k]:=var_o[k]
k=4
sm:=[ ]
b:=0
k:=1
b:=pos('!',var_[k]
b=1
да
нет
str(k,x) pr_o:=x
'введите ответ'
o
n=1,2,
нет
да
Проверяем дру-
гие варианты
3
4
5


3
4
5
max:=max+1
да
o=pr_
нет
summa[i]:=summa[i]+1
nepr[q]:=vopr[i]
k:=0
k:=1
n:=1
да
pos('!',var_o[k]
нет
sound
oo[q]:=
delay
k:=4
nosound
inc(q)
n=12
k=3
i=25
itog:=itog + summa[i]
вывод результата
вывод ответов где  есть ошибка
рекомендации по самостоятельному изучению.
Конец

### 3.2.6 Текст программного модуля на языке Pascal

**<<<Здесь приводится код программы на языке PASCAL или ином, обеспечивающем выполнение задания>>>**

### 3.2.7 Руководство программиста и рекомендации по дальнейшему совершенствованию интерфейса и тест-программы в целом

При дальнейшей разработке и усовершенствовании моей тестирующей программы хотелось бы осуществить обобщение всех созданных тестов по текстовому процессору Word, а в дальнейшем и по других сферам знаний, и предоставить возможность пользователю выбирать тест, который ему хотелось бы пройти. Кроме того, можно создать часть программы, в которой бы мог регистрироваться пользователь, где фиксировались бы все тесты, которые он прошел, и полученные результаты; так можно было бы отслеживать успехи обучаемого. В связи с этим потребовалось бы изменить интерфейс программы: создать меню пользователя.

Так же хотелось бы улучшить графику тестирующей программы, возможно добавить анимацию.

Необходимым изменением мне кажется введение времени ответа на вопрос, тем самым можно было бы свести к минимуму возможность подсказок или дополнительной помощи.

### 3.2.8 Руководство пользователя

Данный тест позволяет в режиме диалога проверить знания пользователя по одному из разделов изучаемых на кафедре дисциплин, а именно знания текстового процессора MS Word в области знаний об упорядоченных списках, стилях форматирования, оформления страниц документа, установления параметров страницы, вставки разрыва страниц, их нумерации и установления колонтитулов.

Пользователю будут представлены 25 вопросов, каждый из которых имеет 4 варианта ответа. Список вопросов теста меняется при каждом новом тестировании, очередность следования вариантов ответа тоже. Правильным может быть только один ответ. Ответ оценивается по дифференцированной 10-бальной шкале в зависимости от сложности вопроса.

Звуковые сигналы помогут узнать, правильно ли ответил пользователь, уже на стадии прохождения теста и примерно предположить уровень своих знаний.

В конце пользователь получит всю информацию о результатах прохождения теста, а именно:

- максимально возможную сумму баллов;
- набранную тестируемым сумму баллов и её % от максимума;
- правильные ответы на те вопросы, по которым тестируемый дал неверный ответ;
- рекомендации по самостоятельному изучению материала с указанием литературных источников вплоть до страниц;
- итоговую оценку за тест.

Итоговая оценка выставляется по следующим критериям:

- Отлично – набрано более 85% баллов от максимально возможной суммы;
- Хорошо – набрано более 70% баллов от максимально возможной суммы;
- Удовлетворительно – набрано более 55% баллов от максимально возможной суммы;
- Неудовлетворительно – набрано менее 55% баллов от максимально суммы.

## Заключение

Задачей данной курсовой работы было создание тест-программы, которая бы помогала осуществить проверку знаний пользователя в режиме диалога, а также содержать некоторый обучающий элемент. В данной курсовой работе выполнены все требования для получения оценки «отлично», а именно: использовать графические и аудио возможности языка Pascal. Количество вопросов – не менее 60, из которых (по случайному закону) для тестирования выбираются 25 вопросов. Вопросы и ответы организуются и хранятся в файлах с возможностью простой коррекции и редактирования учебного материала. Правильный ответ выбирается из 4 вариантов ответов, при этом предлагаемая очередность следования вариантов ответов на каждый вопрос меняется при очередном тестировании по случайному закону. Ответ оценивается по дифференцированной 10-ти бальной шкале в зависимости от сложности вопроса, при неправильном ответе начисляется ноль баллов. В конце высвечиваются:

- максимально возможную сумму баллов;
- набранную тестируемым сумму баллов и её % от максимума;
- правильные ответы на те вопросы, по которым тестируемый дал неверный ответ;
- рекомендации по самостоятельному изучению материала с указанием литературных источников вплоть до страниц;
- итоговую оценку за тест.

Курсовая работа была протестирована на предмет правильной работы, ошибок не выявлено.

## Список использованной литературы


1. Карпов В.Е., Коньков К.А. Основы операционных систем: учебное пособие. Изд. 2-е, доп. и испр .— М.: Интернет-Университет информационных технологий (ИНТУИТ.РУ), 2015 .— 531 с.
2. Фаронов В.В. Турбо Паскаль 7.0.Начальный курс. Учебное пособие. Издание 7-е, переработанное. – М.: Издательство ООО ОМД «Групп», 2015. – 576 с., ил;
3. Зуев Е.А. Программирование на языке Turbo Pascal 7.0. – М.: Веста, Радио и связь, 2017. – 384с., ил;
4. Меженный О.А. Word 2012. Самоучитель. – М.: Издательский дом «Вильямс», 2016. – 288с., ил.
5. Симонович, Евсеев, Алексеев. Специальная информатика. Учебное пособие. Универсальный курс. – М.: Издательство АСТ – пред.: Инфорком-пресс, 2016г.
6. Левин. Самоучитель работы на компьютере. – С.-Пб.: Издательский дом «Питер», 2016г.