Министерство образования Московской области
ГБПОУ МО «Воскресенский колледж»

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ КУРСОВОГО ПРОЕКТА

ПМ.02 «Осуществление интеграции программных модулей» МДК.02.01«Технология разработки программного обеспечения»

для специальности: 09.02.07 «Информационные системы и программирование»

Воскресенск, 2020 г.

Методические рекомендации по выполнению курсового проекта разработаны на основе Федерального государственного образовательного стандарта по специальности среднего профессионального образования 09.02.07 «Информационные системы и программирование», утверждённого приказом Министерства образования и науки РФ от 9 декабря 2016 г. №1547.

Организация-разработчик: ГБПОУ МО «Воскресенский колледж»

Разработчик:

Комиссаров Станислав Александрович, преподаватель специальных дисциплин ГБПОУ МО «Воскресенский колледж»

Сборник методических рекомендаций рассмотрен на заседании предметно-цикловой комиссии компьютерных дисциплин

«___»_____201__ г.

Председатель предметно-цикловой комиссии_____/Рязанцева О.В./

Утверждена зам директора по УР _____/Куприна Н.Л./

«___»_____201___ г.

Содержание

1. Цели и задачи курсового проектирования

Целью курсового проектирования по МДК.02.01 «Технология разработки программного обеспечения» является получение студентами практических навыков:

- Разрабатывать и оформлять требования к программным модулям по предложенной документации.
- Разрабатывать тестовые наборы (пакеты) для программного модуля.
- Разрабатывать тестовые сценарии программного средства.
- Инспектировать разработанные программные модули на предмет соответствия стандартам кодирования. Интегрировать модули в программное обеспечение.
- Отлаживать программные модули.
- Инспектировать разработанные программные модули на предмет соответствия стандартам кодирования.

Задачи курса. В результате выполнения курсового проекта студенты должны

уметь:

- Анализировать проектную и техническую документацию.
- Использовать специализированные графические средства построения и анализа архитектуры программных продуктов.
- Организовывать заданную интеграцию модулей в программные средства на базе имеющейся архитектуры и автоматизации бизнес-процессов.
- Определять источники и приемники данных.
- Проводить сравнительный анализ. Выполнять отладку, используя методы и инструменты условной компиляции (классы Debug и Trace).
- Оценивать размер минимального набора тестов.
- Разрабатывать тестовые пакеты и тестовые сценарии.
- Выявлять ошибки в системных компонентах на основе спецификаций.
- Использовать выбранную систему контроля версий.
- Использовать методы для получения кода с заданной функциональностью и степенью качества.

- Использовать различные транспортные протоколы и стандарты форматирования сообщений.
- Выполнять тестирование интеграции.
- Организовывать постобработку данных.
- Создавать классы-исключения на основе базовых классов.
- Выполнять ручное и автоматизированное тестирование программного модуля.
- Использовать приемы работы в системах контроля версий.
- Использовать инструментальные средства отладки программных продуктов.
- Использовать приемы работы в системах контроля версий.
- Выполнять отладку, используя методы и инструменты условной компиляции.

знать:

- Модели процесса разработки программного обеспечения.
- Основные принципы процесса разработки программного обеспечения.
- Основные подходы к интегрированию программных модулей.
- Виды и варианты интеграционных решений.
- Современные технологии и инструменты интеграции.
- Основные протоколы доступа к данным.
- Методы и способы идентификации сбоев и ошибок при интеграции приложений.
- Методы отладочных классов.
- Стандарты качества программной документации.
- Основы организации инспектирования и верификации.
- Встроенные и основные специализированные инструменты анализа качества программных продуктов.
- Графические средства проектирования архитектуры программных продуктов.
- Методы организации работы в команде разработчиков.
- Основы верификации программного обеспечения.
- Основные методы отладки.
- Методы и схемы обработки исключительных ситуаций.
- Основные методы и виды тестирования программных продуктов.
- Приемы работы с инструментальными средствами тестирования и отладки.
- Основные принципы процесса разработки программного обеспечения.

- Основные подходы к интегрированию программных модулей.

Проектирование ПО должно производиться в соответствии с Государственными стандартами Российской Федерации (ГОСТ Р), разработанными на основе прямого применения международных стандартов ISO.

ГОСТ Р ISO/МЭК 9294-93. Информационная технология. Руководство по управлению документированием программного обеспечения. Устанавливает рекомендации по эффективному управлению документированием ПО.

ГОСТ Р ISO/МЭК 9126-93. Информационная технология. Оценка программной продукции. Характеристики качества и руководство по их применению.

Стандарт определяет шесть комплексных характеристик:

функциональные возможности;

надежность;

практичность;

эффективность;

сопровождаемость;

мобильность.

ГОСТ Р ISO/МЭК 9127-94. Системы обработки информации. Документация пользователя и информация на упаковке для потребительских программных пакетов (ПП). Под документацией пользователя понимается документация, которая обеспечивает информацией по установке и эксплуатации ПП.

ГОСТ Р ISO/МЭК 8631-94. Информационная технология. Программные конструктивы и условные обозначения для их представления. Описывает представление процедурных алгоритмов.

ГОСТ Р ISO/МЭК 12119:2000. Информационная технология. Пакеты программных средств. Требования к качеству и испытания. В этом стандарте установлены требования к качеству пакетов программ и инструкции по их испытаниям на соответствие заданным требованиям.

ГОСТ Р ISO/МЭК 12207-99. Информационная технология. Процессы жизненного цикла программных средств.

2. Указания к выполнению курсового проекта

2.1. Примерное содержание курсовой работы

Титульный лист.
Задание на курсовое проектирование.

Введение.

1. Теоретическая часть.
1.1. Исследование предметной области.
1.2. Моделирование предметной области с использованием диаграмм UML и SADT-технологий.
1.3. Анализ программного обеспечения аналогичного назначения.
1.4. Составление технического задания.
1.5. Требования к надежности разрабатываемого программного обеспечения
1.6. Обоснование выбора модели жизненного цикла ПО.
1.7. Схема и описание процессов жизненного цикла программного обеспечения.
1.8. Обоснование выбора языков программирования и информационных технологий.
1.9. Обоснование выбора вспомогательных инструментальных средств.
2. Практическая часть.
2.1. Моделирование структуры данных
2.2. Описание входных/выходных данных.
2.3. Создание первичных структур данных в выбранной среде разработки.
2.4. Реализация функциональной и интерфейсной частей приложения.
2.5. Проведение мероприятий по защите программного обеспечения.
2.6. Создание пакета инсталляции и справочной системы
3. Мероприятия по интеграции программного обеспечения
3.1. Составление плана тестирования и отладки программного обеспечения.
3.2. Составление тестовых сценариев (Test Case)
3.3. Тестирование и отладка программного обеспечения.
3.3.1. Функциональное тестирование
3.3.2. Нагрузочное тестирование
3.3.3. Конфигурационное тестирование
3.3.4. Тестирование удобства пользования
3.3.5. Тест сборки
3.3.6. Отладка программных модулей приложения
3.4. Составление отчета об ошибках (Bug Report) в ходе тестирования.
3.5. Составление протоколов тестирования и отладки.

Заключение.
Список используемой литературы.
Приложение А (Руководство пользователя)
Приложение Б (Руководство системного программиста)
Приложение В (Диск с программой)

К сведению студентов следует также отметить, что данное содержание является собирательным и отдельные его пункты могут варьироваться в зависимости от типа

разрабатываемого ПО. К примеру, интернет-магазин требует обязательного наличия SEO-продвижения, что однозначно необходимо отразить в содержании отдельным пунктом. Содержание для каждого вида работ отражено в п.3 настоящего документа.

2.2 Порядок выполнения курсового проекта

Введение. В этой части пояснительной записки обучающийся делает краткое вступление, знакомится с общими представлениями о предметной области, для которой он разрабатывает программное обеспечение, и озвучивает: цели курсового проекта (одну или несколько), задачи исследования, объект и предмет исследования, актуальность продукта и его практическую значимость.

Типичными целями проекта обычно являются: разработка моделей данных и алгоритмов; создание технической документации; реализация проекта и проведение мероприятий по интеграции созданного программного продукта.

Задачами исследования, как правило, являются исследование предметной области; анализ существующего ПО; анализ методов и средств автоматизации предметной области.

Типичными объектами исследования являются конкретные понятия и явления исследуемой предметной области, например, связи между подразделениями организации, перечень выполняемых сотрудником отдела работ, перечень типовой документации работника, список технологических операций при изготовлении деталей и т.д.

Предметом исследования являются узкоспециальные действия и методы, направленные на автоматизацию конкретных функций предметной области, к примеру, способы формирования приходных накладных в сфере розничной торговли, методы классификации технологических операций для баз данных, и т.д.

Под актуальностью продукта следует понимать его своевременность и востребованность в современных условиях. В этой части введения обучающийся должен кратко обосновать необходимость разработки своего проекта, доказать его целесообразность.

Также обучающийся, в продолжение темы об актуальности проекта, должен рассказать о практическом применении своего проекта, в т.ч. путем указания на предполагаемую целевую аудиторию.

Разработка ПО начинается с технического задания, пример которого приводится в приложении. Основанием для разработки может быть как документ, например, договор с заказчиком, приказ или распоряжение руководства организации, так и личная инициатива разработчика. Как правило, последний вариант также разделяется на отдельные ситуации, когда работа начинается в силу развития ситуации на рынке ПО (или внутри организации) без какого-либо документа, из-за необходимости, или используется свободными программистами и разработчиками как способ наполнения своего портфолио, и т.д.

Техническое задание должно исчерпывающе отражать все требования к разрабатываемому программному продукту, не допуская широких толкований. Более подробные требования к ТЗ описаны в соответствующих стандартах.

В ходе исследования предметной области обучающийся должен конкретизировать тот круг объектов и действий, который ему предстоит автоматизировать. К примеру, при создании базы данных технологических процессов изготовления деталей следует уделить повышенное внимание классификации этих процессов, номенклатуры станков и инструмента, используемых в ходе этих процессов, список структурных подразделений и цехов предприятий, ответственных за подготовку и реализацию этих процессов. Не следует отвлекаться на общую структуру производства и тем более на объекты и процессы, напрямую с технологическими процессами на связанные (сбыт, снабжение, обеспечение электроэнергией и иными видами топлива, отдел кадров, и т.п.). В этом случае модель данных будет существенно перегружена лишними деталями и в некоторых случаях физически не позволит получить четкие алгоритмы и предсказуемые практически значимые результаты.

Во избежание оставления «за скобками» в ходе исследования предметной области такая существенная её часть, как анализ ПО аналогичного назначения, вынесено в отдельный пункт проекта. Обучающемуся рекомендуется, в зависимости от типа своего программного обеспечения, исследовать: доступную базу данных автоматизированного рабочего места, или веб-сайт (например, интернет-магазин аналогичного профиля и с похожим функционалом), или приложение, которое имеется на рынке или же используется в конкретных организациях. Например, при создании АРМ кладовщика следует внимательно изучить такое ПО, как 1С: Торговля+Склад, понять, какие принципы положены в основу работы этой конфигурации, как классифицируются объекты и явления предметной области (в данном случае работы склада и документов и обязанностей кладовщика), привести несколько скриншотов из выбранного ПО, которые могут дать о нем хотя бы примерное представление – об интерфейсе и способах реализации конкретных функций.

В случаях, если ПО для предметной области существует много, необходимо исследовать три-четыре примера. Этого вполне достаточно в рамках данного проекта. В противном случае необходимо подробно описать хотя бы один пример. Также при этом допускается сравнение с другими типами ПО: например, можно рассмотреть базу данных веб-приложения, если предметная область совпадает, даже при том, что обучающийся разрабатывает обычную БД.

Примечание: подобные отклонения допустимы только в случае малого количества существующего ПО аналогичного назначения.

Также в ходе исследования предметной области обучающемуся необходимо не просто перечислить, какие модели жизненного цикла ПО существуют в принципе, его главная задача – дать исчерпывающий ответ на вопрос, какая из этих моделей является конкретно для его проекта предпочтительной, и почему. В данном пункте не следует углубляться в детали жизненного цикла, важнее именно привязка к своему проекту.

Далее студент занимается составлением схемы и описанием процессов жизненного цикла того программного обеспечения, которое он разрабатывает, на основе стандартной модели. Важно уделить серьёзное внимание каждому этапу жизненного цикла, избегая абстракции, перечислить те работы, которые нужны для разработки именно его ПО, для его предметной области, а не ограничиваться общими словами.

Еще один важный момент: обучающийся должен выбирать модель жизненного цикла, исходя из специфики своего задания, т.е. различать жизненный цикл прикладной программы и веб-сайта. При наличии очень многих общих моментов есть аспекты, где между этими двумя видами программных продуктов существенная разница. Это следует помнить при моделировании и описании жизненного цикла.

В п.1.6 нужно произвести теоретическое обоснование выбора языков программирования и информационных технологий. Это означает, что обучающийся изучает все возможные варианты специализированного инструментария программиста и выбирает один из них, подробно аргументируя, на чём основан его выбор, указать преимущества и недостатки. Описать выбранный инструментарий (к примеру, среду разработки и язык программирования) следует кратко, чтобы дать общее представление о нём. Детализированное описание среды и (или) языка давать не нужно, т.к. все необходимые для этого ссылки на книги, статьи и веб-страницы будут перечислены в конце пояснительной записки. В этом пункте достаточно в качестве примера привести три-четыре скриншота из среды разработки.

Пункт 1.7 содержит в себе те же самые требования. К вспомогательным средствам относятся, как известно, средства для создания справочной системы, для создания инсталлятора, средства обработки графики (если требуется создание логотипа и сплаш-скрина), и т.д. В данном пункте следует кратко перечислить доступные варианты вспомогательного ПО, указать, какие из них будут использованы в данном проекте, подробно описать критерии именно такого выбора, а также пояснить, зачем нужны выбранные вспомогательные средства. В этом пункте необходимо наличие нескольких скриншотов выбранных вспомогательных средств для получения представления, по крайней мере, об их внешнем виде у читателей, желающих ознакомиться с пояснительной запиской.

Последним пунктом теоретической части является описание требований к надежности разрабатываемого программного обеспечения. Обучающемуся следует помнить о том, что требования эти сильно отличаются в зависимости от типа разрабатываемого ПО (к примеру, у интернет-магазина и программы разная зависимость от настроек операционной системы), поэтому следует брать за основу не некоторые абстрактные требования, а требования именно к тому типу ПО, которое реализуется в проекте. Разумеется, обучающийся в первую очередь должен ориентироваться на соответствующие стандарты, принятыми в РФ и международном сообществе, и на эти стандарты ссылаться. Описание требований к надёжности должно быть также адаптировано к специфике конкретного проекта, не быть абстрактным.

Практическая часть проекта начинается чаще всего с моделирования предметной области, что отражено в содержании. Данная часть обязательно должна включать в себя последовательное, поэтапное описание процессов и объектов, подлежащих

автоматизации, и содержать их графическое описание в виде диаграмм IDEF0, IDEF1 или IDEF1X, IDEF3, UML-диаграмм.

Идеальными моделями для полного описания предметной области с точки зрения её автоматизации являются UML-диаграммы, в частности, Use Case, а также некоторые SADT-диаграммы – IDEF0, IDEF1X, IDEF3, DFD. Использование других диаграмм нецелесообразно, т.к. перечисленных диаграмм вполне достаточно, чтобы применить их при проектировании собственно самого ПО.

Зачем вообще нужно моделирование предметной области? Во-первых, в этом случае разработчик будет наглядно видеть, какой конкретно функционал требуется от будущего программного продукта и для кого. Во-вторых, сразу же можно выполнить работы по рациональному дизайну ПП и включить в него только те функции, которые однозначно понадобятся при автоматизации конкретного рабочего места. Соответственно, сразу же можно создать рациональную структуру базы данных, которая будет максимально соответствовать техническому заданию и назначению программного продукта без дальнейших переделок.

В п.2.2 обучающийся моделирует структуру данных – конечным результатом должна быть схема данных с указанием типа связей между таблицами или иными структурными единицами модели.

Пункт 2.3 служит логическим продолжением предыдущего: здесь единицы моделей данных должны быть подробно описаны (с пояснениями), с указанием типов данных и переменных (например, текстовый, целочисленный, строка, вещественный, и т.д.). Входные и выходные данные для разных типов ПО должны отличаться. Очевидно, что типы выбранных данных должны по смыслу соответствовать тем задачам, которые с ними связаны. Например, нельзя ставить целочисленный тип в те операции, которые предполагают возникновение дробных чисел.

Для приложений, веб-приложений и баз данных входные данные это, как правило, некоторая информация, которая вводится пользователем с клавиатуры в поля данных (логины, пароли, сведения для формы документов и т.п.), выходные данные могут представлены как в виде чисел и текста, так и в виде готовых документов, и так далее. Если говорить конкретно о веб-сайтах, то в качестве входных данных следует рассматривать учетные данные пользователя, которые вводятся при регистрации, данные форм обратной связи, опросов, комментариев, а также элементы, помеченные как выбранные и положенные в корзину, если речь идет об интернет-магазине, где дополнительной информацией будут являться данные о количестве заказанной единицы товара, сумме одной единицы, сумме баллов или процентов скидки. Выходной информацией можно считать текстовые и числовые результаты, автоматические ответы на электронную почту пользователя, данные поиска и запросов. При этом не стоит забывать, что в случае использования веб-приложением базы данных всё, что в неё можно внести (неважно, со стороны администратора или пользователя), также является входными данными, а любые результаты обработки запросов – выходными, что необходимо отразить в данном пункте пояснительной записки.

Начиная с пункта 2.3 проводится собственно разработка программного продукта: создаются ER-диаграммы и блок-схемы, создаются структуры БД (при необходимости) в среде Microsoft SQL Server, затем создаются окна приложения в среде Microsoft Visual Studio и пишется код программы.

В 2.5. студент проводит мероприятия по защите приложения. В частности, это так называемая «защита от дурака», т.е. от выполнения случайных или намеренных действий, способных навредить программе или же исказить результаты. Сюда относятся такие меры, как невозможность ввода данных в поля, если тип вводимых данных не совпадает с установленным (например, в поле «Цена» нельзя вводить текст), невозможность ввода данных в те поля, где предусмотрен только вывод промежуточных или итоговых результатов, и так далее.

Еще одной популярной мерой защиты ПО является разграничение прав пользователей, и в зависимости от того, под каким пользователем был выполнен вход в программу, различаются их функциональные возможности и степень влияния на итоговые результаты.

В 2.6. нужно, используя любую программу соответствующего назначения, собрать все файлы проекта в так называемый пакет инсталляции. Шаги последующей установки должны включать в себя, соответственно, такие базовые опции, как место установки, создание ярлыков, возможность автоматического запуска и другое.

Далее студент переходит к мероприятиям по интеграции созданных им программных модулей. Чтобы в ходе работы не было хаоса, необходимо составить план тестирования, куда он включает все виды тестов и краткое описание методик – например, изберёт ли он для юзабилити-теста метод опроса пользователей или обратится к методикам, основанных на эргономике. Также необходимо создать тест-кейсы, которые выглядят примерно так:

Функциональный тест-кейс.

Шаг	**Тестовые шаги**	**Тестовые данные**	**Ожидаемый результат**	**Фактический результат**	**Статус (проход / неудача)**	**Скриншот**	**Заметки**
1	Перейдите на страницу входа		Пользователь должен иметь возможность войти				
2	Введите действительное имя пользователя	Пользователь =1a@gmail.com	Учетные данные могут быть введены				

3	Введите действительный пароль	Пароль: 1234	Учетные данные могут быть введены				
4	Нажмите на кнопку «Войти»		Пользователь вошел				

Разумеется, видов тестирования множество, и студент может выбрать любые из них, но минимальный тестовый набор следующий: функциональный, тест конфигурации, нагрузочный, юзабилити, совместимости, сборки.

Тест-кейс представляет собой бланк, где есть входные данные и ожидаемые результаты.

Далее начинается непосредственно тестирование, которое сопровождается соответствующими записями в тест-кейсах. По сути, каждый заполненный тест-кейс – это протокол тестирования без реквизитов. Для подтверждения реальности действий по тестированию необходимо вставить в тест-кейс скриншоты.

В ходе интеграции возможны различные ошибки: как в ходе компиляции, так и в процессе тестирования, и в реальной эксплуатации, от логических до синтаксических. Задача студента – выявлять все подобные случаи и в обязательном порядке проводить отладку ПО, а результаты записывать в специальную таблицу, которая называется отладочным листом, пример которого приведён ниже.

№ п/п	Название модуля (файл)	Номер строки с ошибкой	Текст кода с ошибкой	Сообщение об ошибке	Причина ошибки	Способ устранения	Используемый метод отладки (с указанием способа в главном меню системы программирования)	Результат отладки

Далее, в 3.4 студент составляет отчет об ошибках, возникших на всех этапах интеграции (без описания способов их устранения и указания причин, так называемый баг-репорт), в 3.5. протоколирует ход испытаний – не дублируя тест-кейсы, указывает, кем и в какое время проводилось тестирование программного модуля, а также условия тестирования – окружающее аппаратное и программное обеспечение. В

протоколирование отладки включаются те же самые сведения, плюс затраченное на исправление ошибок время. Протокол составляется в свободной форме.

В заключении студент подводит краткие итоги своей работы, перечисляет, что из запланированного в проекте удалось реализовать, а что – нет и что стало этому причиной.

Список литературы формируется на основании актуальных на момент выполнения курсового проекта учебников по данному МДК, а также с использованием актуальных интернет-ресурсов, как приведённых в списке литературы, так и найденных самостоятельно. Печатные издания актуальны не более 5 лет, затем они переводятся в разряд «Дополнительная литература». При указании интернет-источников необходимо указать дату и время захода на ресурс в связи с тем, что много источников используют динамические страницы типа aspx или php и содержимое их может неоднократно меняться и найти информацию можно будет в дальнейшем только через веб-архивы.

Нумерация источников сквозная. Разделы списка: «Основная литература», «Дополнительная литература», «Интернет-источники». Образец правильного оформления списка литературы можно увидеть в настоящем сборнике методических рекомендаций.

В приложении А студент составляет руководство пользователя по использованию функций разработанного программного продукта в виде пошаговых инструкций с текстовым описанием и скриншотами.

В приложении Б составляется руководство системного программиста, где приводятся ключевые элементы кода программы с указанием их функций, чтобы в случае необходимости программист, не имевший отношению к написанию данного кода, мог сориентироваться и внести необходимые изменения. Также составляется руководство по настройкам программного продукта и исправлению возможных ошибок.

Приложение В представляет собой приклеенный канцелярским клеем или с помощью двустороннего скотча диск в конверте, на котором записываются сам программный продукт в виде исходного кода и файл-инсталлятор (чаще всего setup.exe), а также файл с пояснительной запиской к курсовому проекту. Обратите внимание, что диск вкладывается в специальный конверт для диска, который приклеивается лицевой стороной к листу посередине, клапаном вверх для удобства извлечения диска. Самодельные конверты не допускаются.

2.3. Требования к описанию продукта, к пользовательской документации, программам и данным

Основным назначением описания продукта является определение свойств программного продукта и оказание помощи пользователю в работе с ним.

Требования к описанию ПО, установленные стандартом ГОСТ Р ISO/МЭК 12119:2000:

- Общим требования к содержанию
- Обозначения и указания
- Функциональные возможности
- Надежность
- Практичность
- Эффективность
- Сопровождаемость и мобильность.

Описание продукта должно удовлетворять **общим требования к содержанию**:

Быть достаточно понятным, полным и простым при изучении. Быть внутренне непротиворечивым. Каждый термин должен иметь один и тот же смысл по всему документу. Формулировки должны быть проверяемыми и корректными.

При описании продукта необходимо приводить следующие указания и обозначения:

1. При обозначении одного или нескольких продуктов в рамках одного пакета необходимо включать наименование продукта и обозначение его версии или даты выпуска.

2. Должны быть включены наименование и адрес поставщика.

3. Должны быть определены целевые рабочие задачи, которые могут быть выполнены данным продуктом.

4. Из описания продукта могут быть даны ссылки на нормативные документы, которым удовлетворяет данный продукт, в этом случае должны быть указаны соответствующие редакции данных документов.

5. Должна быть определена система (технические и программные средства и их конфигурация), необходимая для ввода продукта в эксплуатацию, включая наименования изготовителей и обозначения типов всех ее частей, например:

- Процессоры, включая сопроцессоры
- Объем основной (оперативной) памяти
- Типы и объемы (памяти) периферийных запоминающих устройств
- Расширяющие платы
- Оборудование ввода и вывода
- Сетевое оборудование
- Системные и прочие программные средства

6. Должны быть определены соответствующие интерфейсы или продукты, если в описании продукта имеются ссылки на интерфейсы с другими продуктами.

7. Должен быть определен каждый физический компонент поставляемого продукта, в частности все печатные документы и все носители данных

8. Должен быть установлен вид поставляемых программ, например исходные программы, объектные (рабочие) модули или загрузочные модули.

9. Должно быть указано, будет ли инсталляция продукта проводиться пользователем или нет

При описании функциональных возможностей необходимо отразить:

1.Обзор функций

В описании продукта должен быть приведен обзор функций продукта, вызываемых пользователем, необходимых для них данных и предоставляемых средств. Для каждой функции (особенно для ее опции или варианта) должно быть четко установлено, является ли она частью:

- Продукта.
- Расширения продукта, полностью приведенного в описании продукта.
- Расширения продукта, на которое дана ссылка в описании продукта.
- Не гарантируемого (необязательного) приложения.

2.Граничные значения

Если использование продукта ограничено конкретными граничными значениями, они должны быть указаны в описании продукта, например:

- Минимальные или максимальные значения.
- Длины ключей
- Максимальное число записей в файле
- Максимальное число критериев поиска
- Минимальный объем выборки.

3.Защита

При необходимости в описании продукта должна быть включена информация по средствам предотвращения несанкционированного доступа к программе и данным.

Описывая надежность продукта, необходимо провести информацию по процедурам сохранения данных. Например:

- Проверка достоверности исходных данных
- Описание технологии сбора, передачи, обработки и выдачи информации.
- Защита против серьезных последствий ошибки пользователя

- Восстановление при ошибках.

Описывая практичность, необходимо описать:

1.Интерфейс пользователя. Должен быть назван тип интерфейса: строка команд; меню; окна; функциональные клавиши; функция подсказки и др.

2.Требуемые знания. Должны быть определены конкретные знания, которые необходимо усвоить пользователю для применения соответствующего продукта, например:

- Знание соответствующей технической области
- Знание операционной системы
- Знания, получаемые в результате специального обучения

3.Адаптация к потребностям пользователя. Если продукт может настраиваться (адаптироваться) пользователем, то должны быть указаны инструментальные средства для проведения такой настройки и условия их применения, например:

- изменение параметров;
- изменение алгоритмов вычислений;
- назначение функциональных клавиш.

4.Защита от нарушения авторских прав. В описании продукта должны быть указаны виды и средства такой защиты. Например:

- техническая защита от копирования;
- запрограммированные даты окончания использования продукта;
- интерактивные напоминания об оплате за копии.

5.Эффективность применения и удовлетворение потребностей пользователя. В описание продукта может быть внесена информация по эффективности применения продукта.

Описывая эффективность, необходимо отразить информацию о характере поведения продукта во времени, например, указать время ответа и время оценки производительности для заданных функций при установленных условиях (например, для заданных конфигураций системы и профилей загрузки).

В описание продукта могут быть внесены формулировки требований (правил) по сопровождению и мобильности продукта.

Документация пользователя

Документация пользователя: полный комплект документов, поставляемых в печатном или другом виде, который обеспечивает применение продукта, а также является его неотъемлемой частью.

Документация пользователя должна отвечать следующим характеристикам:

1. ***Полнота***. Документация пользователя должна содержать информацию, необходимую для использования продукта. В ней должны быть полностью описаны все функции, установленные в описании продукта, и все вызываемые пользователем функции из программы. Кроме того, граничные значения, заданные в описании продукта, должны быть продублированы в документации пользователя. Если установка (инсталляция) продукта может быть проведена пользователем, то в документацию пользователя должно быть включено руководство по установке продукта, содержащее всю необходимую информацию. Если сопровождение продукта может проводиться пользователем, то в документацию пользователя должно быть включено руководство по сопровождению программы, содержащее всю информацию, которая необходима для обеспечения данного вида сопровождения.

2. ***Правильность***. Вся информация в документации пользователя должна быть правильной. Кроме того, представление данной информации не должно содержать неоднозначных толкований и ошибок.

3. ***Непротиворечивость***. Документы, входящие в комплект документации пользователя, не должны противоречить сами себе, друг другу и описанию продукта. Каждый термин должен иметь один и тот же смысл во всех документах.

4. ***Понятность***. Документация пользователя должна быть понятной для сообщества пользователей, выполняющих указанную рабочую задачу, например, посредством использования в ней соответствующим образом подобранных терминов, графических вставок, уточняющих пояснений и путем ссылок на полезные источники информации.

5. ***Простота обозрения***. Документация пользователя должна быть достаточно проста для изучения пользователем.

Программы и данные

Программы и данные должны соответствовать всем обязательным формулировкам, приведенным в описании продукта и документации пользователя.

Надежность.

Это требование должно одинаково удовлетворяться в случаях, когда:

- возможность реализуется при конкретных ограничениях;
- имеют место попытки реализации возможности вне заданных ограничений;
- неправильные исходные данные вводятся пользователем или от других программ, перечисленных в описании продукта;
- нарушаются инструкции, заданные в документации пользователя.

Исключаются только те возможности прерывания технических средств и операционной системы, которые не могут быть распознаны.

Практичность.

1. Понятность.

Запросы, сообщения и результаты выполнения программ должны быть понятными, например:

- путем соответствующего выбора терминов;

- путем графических представлений;

- путем представления исходной информации;

- путем пояснений из функции подсказки.

В сообщениях об ошибках должна содержаться подробная информация, разъясняющая причину или способ исправления соответствующих ошибок из-за неправильного применения продукта (например, путем ссылки на элемент документации пользователя).

2. Простота обозрения.

Программы должны предоставлять пользователю информацию в таком виде, чтобы данная информация им легко воспринималась и читалась. Сообщения от программ следует проектировать таким образом, чтобы пользователь мог легко различать их типы, пример:

- подтверждение приема;

- запросы от программ;

- предупреждения;

- сообщения об ошибках.

Форматы входных экранов, отчеты, а также другие исходные и выходные данные следует проектировать так, чтобы их можно было легко просматривать.

3. Простота использования.

Исполнение функций, приводящих к серьезным последствиям при эксплуатации системы, должно быть обратимым или программы должны выдавать четкие предупреждения о последствиях выполнения данных функций и запрашивать разрешающее подтверждение перед выполнением соответствующей команды.

Задания на курсовое проектирование

Задания разделяются на три группы. Студенты по собственному желанию могут выбрать разработку базы данных, веб-сайта или программного продукта. Являясь постановщиком задачи, каждый из них может дополнить данные вариантов заданий необходимой нормативно-справочной и оперативной информацией.

1. Тема проекта: «Проектирование и создание АРМ работника складского комплекса»
2. Комплексная автоматизация предприятия малого бизнеса на примере автосервиса
3. Проектирование и создание АРМ сотрудника отдела кадров предприятия общественного питания
4. Проектирование и создание АРМ администратора и сотрудников ветеринарной клиники
5. Проектирование и создание БД для учета арендованного транспорта и спецтехники
6. Проектирование и создание БД «Библиотека кинофильмов»
7. Проектирование и создание АРМ администратора пункта проката велосипедов
8. Проектирование и создание АРМ администратора автомойки
9. Проектирование и создание АРМ администратора гостевого дома
10. Проектирование и создание АРМ сотрудника ФОК
11. Проектирование и создание веб-приложения «Электронный учебник по дисциплине «Элементы высшей математики»
12. Проектирование и создание веб-приложения «Электронный учебник по дисциплине «Элементы математической логики»
13. Проектирование и создание АРМ администратора фитнес-клуба
14. Проектирование и создание АРМ сотрудника фирмы, осуществляющей ремонт компьютеров
15. Проектирование и создание АРМ диспетчера метрополитена
16. Проектирование и создание АРМ сотрудника фуд-корта
17. Проектирование и создание БД «Справочник технологических процессов разработки программного обеспечения»
18. Проектирование и создание АРМ коменданта общежития ГБПОУ МО «Воскресенский колледж»
19. Разработка веб-приложения для тестирования обучающихся в учебных заведениях среднего профессионального образования
20. Разработка веб-приложения для формирования электронной очереди на прием в медицинское учреждение
21. Проектирование и создание веб-сайта «Автоматизированный каталог мультимедиа-ресурсов»
22. Разработка веб-сайта «Интернет-магазин элитных сортов чая и кофе»
23. Разработка веб-сайта «Интернет-магазин одежды и товаров»
24. Проектирование и создание АРМ сотрудника библиотеки ГБПОУ МО «Воскресенский колледж»
25. Проектирование и создание мобильного приложения для настройки смартфонов под управлением Android

26. Проектирование и создание программы для организации и осуществления общественного питания в ГБПОУ МО «Воскресенский колледж»
27. Проектирование и создание приложения для комплексной автоматизации пункта обмена валюты
28. Проектирование и создание АРМ инженера кабинета информатики ГБПОУ МО «Воскресенский колледж»
29. Проектирование и создание программы «Мобильный мессенджер»

Требования к оформлению работы:

Гарнитура Times New Roman, размер шрифта – 12, интервал между строками 1,5 пт, отступ абзаца - 1,25, размер левого поля – 3 см (под подшивку), правого – 2 см, верхнего и нижнего – по 1 см.

Рекомендуемый объём курсового проекта – не менее 50 страниц.

Литература

Стандарты

ГОСТ Р ISO/МЭК 9294-93. Информационная технология. Руководство по управлению документированием программного обеспечения.
ГОСТ Р ISO/МЭК 9126-93. Информационная технология. Оценка программной продукции. характеристика качества и руководство по их применению.
ГОСТ Р ISO/МЭК 9127-94. Системы обработки информации. Документация пользователя и информация на упаковке для потребительских программных пакетов.
ГОСТ Р ISO/МЭК 8631-94. Информационная технология. Программные конструктивы и условные обозначения для их представления.
ГОСТ Р ISO/МЭК 12119:2000. Информационная технология. Пакеты программных средств. Требования к качеству и испытания.
ГОСТ 2.108 - 68 - Спецификация (ЕСКД)
ГОСТ 2.105 - 95 - Общие требование к текстовым документам. (ЕСКД)
ГОСТ 2.103 - 68 - Стадии разработки (ЕСКД)
ГОСТ 2.106 - 68 - Ведомость спецификаций (ЕСКД)
ГОСТ - 7.1- 84 - Библиографическое описание документа. Общие требования и правила составления. (ЕСКД)
ГОСТ 2.004 - 88 - Общие требования к выполнению конструкторских и технологических документов на печатающих и графических устройствах вывода ЭВМ. (ЕСКД)
ГОСТ 19.101-77 - Виды программ и программных документов
ГОСТ 19.103-77 - Обозначение программ и программных документов.
ГОСТ 19.102 - 77 - Стадии разработки
ГОСТ 19.104 - 78 - Основные надписи
ГОСТ 19.105 - 78 - Общие требования к программным документам
ГОСТ 19.201 - 78 - Техническое задание.
ГОСТ 19.202 - 78 - Спецификация
ГОСТ 19.781- 90 - Термины и определения
ГОСТ 19.701-90 - Схемы алгоритмов, программ данных и схем. Условные обозначения и правила выполнения

Основная литература:

1. Федорова Г.Н. Осуществление интеграции программных модулей: учебник для студ.учреждений сред. проф. образования. 2-е изд., стер. – М.: Издательский центр «Академия», 2018. – 288 с.
2. Рудаков А.В. Технология разработки программных продуктов: учебник для студ.учреждений сред. проф. образования. 9-е изд., стер. – М.: Издательский центр «Академия», 2014. – 208 с.
3. Рудаков А.В., Федорова Г.Н. Технология разработки программных продуктов. Практикум: учеб. пособие для студ.учреждений сред. проф. образования. 4-е изд., стер. – М.: Издательский центр «Академия», 2014. – 192 с.
4. Федорова Г.Н. Участие в интеграции программных модулей: учеб. пособие для студ. учреждений сред. проф. Образования/Г.Н.Федорова. – М.:Издательский центр «Академия», 2016. – 304 с.

Дополнительная литература:

5. Гагарина Л.Г., Виснадул Б.Д., Игошин А.В. Основы технологии разработки программных продуктов: Учеб.пособие. – М.: ФОРУМ: ИНФРА-М, 2006. – 192 с.
6. Плаксин М.А. Тестирование и отладка программ – для профессионалов будущих и настоящих. – М,: БИНОМ. Лаборатория знаний, 2007. – 167 с., ил.
7. Смирнова И.Е. Начала Web-дизайна. – СПб.: БХВ-Петербург, 2—3. – 256 с.: ил.
8. Основы Web-технологий/П.Б. Храмцов, С.А. Брик, А.М.Русак, А.И.Сурин. – М.: ИНТУИТ.РУ «Интернет-университет Информационных технологий, 2003. – 512 с.
9. Создание веб-страниц. Самоучитель/Т.Стацфер. – СПб.: Питер, 2003. – 448 с.: ил.
10. Флэнаган Д.. JavaScript. Подробное руководство. – Пер. с англ. – СПб: Символ-Плюс, 2004. – 960 с.
11. Дунаев В. Самоучитель JavaScript, 2-е изд. = Спб.: Питер, 2005. – 395 с.: ил.
12. Р.Ноблес, К.-Л. Греди. Эффективный веб-сайт. Учебное пособие. - М.: Издательство ТРИУМФ, 2004. – 56- с.: ил.
13. Джеймс Р. Грофф, Пол Н. Вайнберг. SQL: полное руководство: пер. с англ. – К.: Издательская группа BHV, 2000. – 608 с.
14. Карпов Б. Delphi: специальный справочник. – СПб: Питер, 2002. – 688 с.: ил.

15. Хомоненко А.Д. и др. Delphi 7/под общ. ред. А. Д. Хомоненко. – СПб.: БХВ-Петербург, 2003. – 1216 с.: ил.
16. Михеева В.Д., Харитонова И.А. Microsoft Access 2002. – СПб: БХВ-Петербург, 2002. – 1040 с.: ил.

Интернет-источники:

17. Protesting.ru
18. https://docs.microsoft.com/ru-ru/visualstudio/debugger/quickstart-debug-with-cplusplus?view=vs-2019
19. http://cppstudio.com

Министерство образования Московской области
Государственное бюджетное профессиональное
образовательное учреждение Московской области
«Воскресенский колледж»

Специальность 09.02.07 «Информационные системы и программирование»

КУРСОВОЙ ПРОЕКТ

ПМ.02 «Осуществление интеграции программных модулей»

МДК.02.01«Технология разработки программного обеспечения»

Тема: «Проектирование и создание АРМ сотрудника библиотеки ГБПОУ МО «Воскресенский колледж»»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
КП.ТРПО.01.09.00.ПЗ.

Принял(а)
_____

Выполнил(а)
студент(ка) группы ДП-4
_____

20__ г.