Приложение 1.5.3 к ОПОП по специальности 09.02.07
Информационные системы и программирование

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет им. Г.И. Носова»

Многопрофильный колледж

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ЛАБОРАТОРНЫХ ЗАНЯТИЙ МЕЖДИСЦИПЛИНАРНОГО КУРСА МДК.05.03 Тестирование информационных систем

для обучающихся специальности

09.02.07 Информационные системы и программирование

Магнитогорск, 2025

ОДОБРЕНО

Предметно-цикловой комиссией «Информатики и вычислительной техники»
Председатель Т.Б. Ремез
Протокол № 5 от «22» января 2025г

Методической комиссией МпК

Протокол № 3 от «19» февраля 2025г

Разработчик (и):

преподаватель отделения № 2 «Информационных технологий и транспорта» Многопрофильного колледжа ФГБОУ ВО «МГТУ им. Г.И. Носова» С.М. Утралинова

Методические указания по выполнению практических и лабораторных работ разработаны на основе рабочей программы ПМ.05. Проектирование и разработка информационных систем, МДК.05.03 Тестирование информационных систем.

Содержание практических и лабораторных работ ориентировано на подготовку обучающихся к освоению профессионального модуля программы подготовки специалистов среднего звена по специальности 09.02.07 Информационные системы и программирование и овладению профессиональными компетенциями.

Содержание практических и лабораторных работ ориентировано на подготовку обучающихся к освоению вида деятельности ПМ.05. Проектирование и разработка информационных систем, МДК.05.03 Тестирование информационных систем программы подготовки специалистов среднего звена по специальности 09.02.07 Информационные системы и программирование. и овладению профессиональными компетенциями.

СОДЕРЖАНИЕ

1 ВВЕДЕНИЕ

Важную часть теоретической и профессиональной практической подготовки обучающихся составляют практические и лабораторные занятия.

Состав и содержание практических и лабораторных занятий направлены на реализацию Федерального государственного образовательного стандарта среднего профессионального образования.

Ведущей дидактической целью практических занятий является формирование профессиональных практических умений (умений выполнять определенные действия, операции, необходимые в последующем в профессиональной деятельности).

Ведущей дидактической целью лабораторных занятий является экспериментальное подтверждение и проверка существенных теоретических положений (законов, зависимостей).

В соответствии с рабочей программой программного модуля ПМ 05. Проектирование и разработка информационных систем, МДК.05.03 Тестирование информационных систем, предусмотрено проведение практических и лабораторных занятий.

В результате их выполнения, обучающийся должен:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Содержание практических и лабораторных занятий ориентировано на подготовку обучающихся к освоению профессионального модуля программы подготовки специалистов среднего звена по специальности и овладению ***профессиональными компетенциями:***

ПК 5.2. Разрабатывать проектную документацию на разработку информационной системы в соответствии с требованиями заказчика.

ПК5.5. Осуществлять тестирование информационной системы на этапе опытной эксплуатации с фиксацией выявленных ошибок кодирования в разрабатываемых модулях информационной системы

ПК5.6. Разрабатывать техническую документацию на эксплуатацию информационной системы.

А также формированию ***общих компетенций***:

- ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.
- ОК 02. Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности
- ОК 04. Эффективно взаимодействовать и работать в коллективе и команде.
- ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста.
- ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.

Выполнение обучающимися практических и лабораторных работ по учебной дисциплине «Проектирование и разработка информационных систем, МДК.05.03. Проектирование и разработка информационных систем, направлено на:

- обобщение, систематизацию, углубление, закрепление, развитие и детализацию

полученных теоретических знаний по конкретным темам учебной дисциплины;

- формирование умений применять полученные знания на практике, реализацию единства интеллектуальной и практической деятельности;
- развитие интеллектуальных умений у будущих специалистов: аналитических, проектировочных, конструктивных и др.;
- выработку при решении поставленных задач профессионально значимых качеств, таких как самостоятельность, ответственность, точность, творческая инициатива.

Практические и лабораторные занятия проводятся в рамках соответствующей темы, после освоения дидактических единиц, которые обеспечивают наличие знаний, необходимых для ее выполнения.

2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Тема 5.3.1. Отладка и тестирование информационных систем

Лабораторное занятие № 1 Разработка тестового сценария проекта

Цель работы – получения навыков создания тестовых сценариев на основе методики управления требованиями.

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации;

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства.

Задание:

1. Построить тестовые сценарии.

Порядок выполнения работы:

А. Определение переменных для каждого шага сценариев использования

1. Переменные для основного потока сценария использования:

Переписать основной поток сценария «Бронирование билетов» с указанием используемых переменных (с префиксом V):

B1. Турист вводит URL сайта.

B2. Система отображает домашнюю страницу сайта.

B3. Турист вводит информацию о полете:

V3.1. Дата вылета

V3.2. Время вылета

V3.3. Дата прибытия

V3.4. Время прибытия

V3.5. Число путешествующих взрослых

V3.6. Число путешествующих детей.

V3.7. Наличие путешествующих животных.

Турист выбирает «Поиск рейсов».

B4. Система отображает рейсы вылета, отсортированные по цене.

B5. Турист выбирает рейс.

V5.1. Рейс вылета

B6. Система отображает рейс прибытия.

B7. Турист выбирает рейс прибытия.

V7.1. Рейс прибытия.

B8. Система отображает детали рейса.

B9. Турист подтверждает рейс.

B10. Пользователь предоставляет Идентификатор и Пароль для покупки билета.

V10.1. Идентификатор

V10.2. Пароль

B11. Турист предоставляет информацию пассажира.

V11.1. Фамилия.

V11.2. Имя.

V11.3. Отчество.

V11.4. Пол.

V11.5. Дата рождения.

V11.6. Номер паспорта.

V11.7. Серия паспорта.

B12. Система отображает свободные места.

B13. Турист выбирает места.

V13.1. Место.

B14. Система отображает доступное меню.

B15. Турист выбирает меню.

V15.1. Тип меню.

V15.2. Тип напитков.

B14. Турист предоставляет информацию по кредитной карте и расчетный адрес.

V14.1. Тип кредитной карты

V14.2. Номер кредитной карты

V14.3. Дата окончания срока действия

V14.4. Название карты

V14.5. Адрес

B15. Система предоставляет номер подтверждения.

2. Аналогичным образом ввести перечень переменных для каждого альтернативного потока.

Б. Определение различных вариантов для каждой переменной

1. Определить значения переменных для тестирования:

V3.1. Дата вылета

– Верная дата в будущем, установленная вручную

– Верная дата в будущем, установленная из календаря

– Дата в прошлом

– Сегодняшняя дата

– Февраль 30 или 31 число

– Пустое поле

V3.2. Время вылета:
– Верное время для будущей даты, установленное вручную
– Верное время для текущей даты, установленное вручную
– Неверный формат даты
– Пустое поле

V3.5. Число путешествующих взрослых
– 0
– 1
–2
– Максимально допустимое

V10.1. Идентификатор
– Верный идентификатор пользователя
– Идентификатор, содержащие недопустимые символы
– Несуществующий идентификатор пользователя
– Пустое поле

V10.2. Пароль
– Правильный пароль пользователя (с правильным идентификатором)
– Неправильный пароль (с правильным идентификатором)
– Верный пароль (с неверным идентификатором)
– Пароль, содержащий недопустимые символы
– Пустое поле

По аналогии определить возможные значения для остальных переменных (не менее трех значений на переменную).

В. Создание тестовых сценариев

1. Построить матрицу распределения тестовых сценариев:

Шаг	Переменная	T1	T2	T3	T4	T5	T6
B3	Время вылета						
B3	Дата прибытия						
B3	Время прибытия						
B3	Число путешествующих взрослых						

2. Для каждой строки ввести все необходимые для тестирования варианты:

Шаг	Переменная	T1	T2	T3	T4	T5	T6
B3	Время вылета	Верная дата в будущем, установленная вручную	Верная дата в будущем, установленная из календаря	Дата в прошлом	Сегодняшняя дата	Февраль 30 или 31 число	Пустое поле
				Верная дата в будущем, установленная из календаря		Верная дата в будущем, установленная из календаря	Верная дата в будущем, установленная вручную
B10	Пароль	Правильный пароль пользователя (с правильным идентификатором)	Неправильный пароль (с правильным идентификатором)	Верный пароль (с неверным идентификатором)	Пароль, содержащий недопустимые символы (с неверным идентификатором)	Пароль, содержащий недопустимые символы (с верным идентификатором)	Пустое поле
			Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)

3. Определить значения переменных:

3.1. Для первого тестового сценария (T1) определить конкретные значения переменных:

Шаг	Переменная	T1	Значение	Ожидаемый результат	Фактический результат	Удачный / неудачный	Комментарии
B3	Время вылета	Верная дата в будущем, установленная вручную	16.04.2015	Принято			
B10	Пароль	Правильный пароль пользователя (с правильным идентификатором)	hcvbbnsj	Принято			

3.2. Заполнить аналогичные таблицы по остальным тестовым сценариям.

Ход работы:

Процесс создания тестовых сценариев включает в себя четыре шага:

1. Определение переменных для каждого шага сценариев использования.
2. Определение существенно разных вариантов для каждой переменной.
3. Комбинирование вариантов для тестирования в тестовые сценарии.
4. Определение значений переменных. Первым делом нужно определить все входные переменные во всех шагах в представ-

ленном сценарии (алгоритме). Например, если на некотором шаге пользователь вводит свой идентификатор и пароль, это две переменных. Первая переменная – Идентификатор, вторая – Пароль. Переменной также может быть выбор, который пользователь должен сделать (такой как выбор рейса из списка).

Количество переменных может зависеть от введенных на предыдущем шаге значений.

На следующем шаге следует определить существенно различные варианты для каждой переменной. Варианты считаются «существенно разными», если они могут вызывать различное поведение системы.

Например, если выбрать Идентификатор, который предполагается быть длиной до десяти символов, следующие приведенные значения будут существенно разными:

Alex – слишком короткий и поэтому ожидается появление сообщения об ошибке.

Alexandra – верный Идентификатор.

Alexandrena – слишком длинный, ожидается, что система не позволит вводить слишком длинный Идентификатор.

Однако, Alexandria или JohnGordon различны не существенно, т.к. оба являются верными идентификаторами, вызывающими одно поведение системы.

Вариант может считаться существенно другим, если:

- Он вызывает другой ход процесса (обычно альтернативный поток).

Пример: Ввод неверного пароля вызовет Альтернативный Поток 2.

Он вызывает другое сообщение об ошибке.

Пример: Если адрес электронной почты слишком длинный, сообщение должно быть: «E-mail должен быть не более 50-ти символов».

Пример: Если адрес электронной почты не содержит символа «@», сообщение должно быть: «Неверный E-mail адрес».

– Он вызывает другой вид пользовательского интерфейса.

Пример: Если кредитная карта была выбрана в качестве способа оплаты, система должна отображать экран с полями для ввода номера кредитной карты, даты окончания срока действия и название организации, обслуживающей карту.

– Он вызывает различный набор значений списков.

Пример: Экран регистрации пользователя должен содержать список Страна и Штат/Провинция. Список Штат/Провинция будет содержать значения на основе выбранной страны: для США он должен содержать все штаты, для Канады – все провинции, для других стран он должен быть недоступен.

– Он является условием ограничения.

Пример: Пароль должен быть не менее 6-ти символов.

В этом случае необходимо протестировать следующее:

Пароль с пятью символами.

Пароль с шестью символами

– Что-то должно быть изменено вместо использования значения по умолчанию.

Пример: На экране оплаты кредитной картой название организации, обслуживающей кредитную карту, должно быть заполнено именем лица, размещающего заказ. Пользователь должен иметь возможность хранить это значение по умолчанию или ввести новое.

– Это создает два различных варианта:

Хранить установленное по умолчанию название организации, обслуживающей кредитную карту.

Изменить установленное по умолчанию название организации на другое.

– Формат ввода точно не определен и может быть интерпретирован разными способами.

Пример: поле ввода номера телефона должно принимать текст в свободной форме.Номера телефонов разными лицами пишутся по-разному:

- Использования скобок: (973) 123-4567
- Использование тире: 973-123-4567
- Использование пробелов: 973 123 4567
- Без пробелов: 9731234567

Все доступные варианты должны быть протестированы.

- Стандартные варианты могут быть разными для разных стран.

Формат даты срока окончания действия кредитной карты может быть разным для США и Европы.

На следующем шаге нужно скомбинировать их в последовательность шагов тестового сценария. Один из способов это сделать – создать Матрицу Распределения Тестовых Сценариев (Test Case Allocation Matrix). Строки этой матрицы содержат все переменные для всех шагов, требующие ввода данных он пользователя. Первая колонка содержит номер шага, вторая – название переменной, а остальные – тестовые сценарии. Их можно назвать Т1, Т2, и т.д. Нужно оценить, насколько много тестовых сценариев нужно для охвата данного сцена- рия (алгоритма). Жестким вариантом оценки будет максимальное количество существенно различных вариантов, определенное для переменной. Не проблема, если при оценке будет допущена ошибка, т.к. можно добавить или удалить колонку при заполнении матрицы. Обычно типичный сценарий охватывают от пяти до семи тестовых сценариев. Тем не менее, иногда в особых случаях требуется больше тестовых сценариев.

На четвертом шаге следует заменить неопределенные варианты, такие как «очень длинная фамилия» или «длинный номер телефона с ext. (дополнительным номером)» на действительные значения, например «Georgiamistopolis» и «011-48 (242) 425-3456 ext. 1234» соответственно. На этом шаге также разделяют все тестовые сценарии из Матрицы Распределения Тестовых Сценариев, создавая отдельную таблицу для каждого тестового сценария.

Метод извлечения функциональных тестовых сценариев (test cases) из сценариев использования (use cases) имеет несколько преимуществ:

– Тестовые сценарии получаются с использованием более автоматического подхода.

– Уход от дублирования тестовых сценариев.

– Достигается больший охват тестового пространства.

– Легкость мониторинга тестового процесса.

– Легкость распределения работы между тестерами.

– Легкость регрессионного тестирования.

– Раннее обнаружение пропущенных требований.

Созданные тестовые сценарии могут быть использованы для ручного тестирования, также как и для автоматического тестирования с использованием таких инструментов, как IMB Rational Robot или IBM Rational Functional Tester.

Форма представления результата:

Оформить отчет по лабораторной работе.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход решения задачи должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при решении задачи и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 2 Разработка тестовых пакетов

Цель: получить навыки разработки тестовых сценариев.

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства.

Задание:

1. Разработать набор тестовых сценариев (как позитивных, так и негативных) для следующей программы: Имеется консольное приложение (разработать самостоятельно). Ему на вход подается 2 строки. На выходе приложение выдает число вхождений второй строки в первую. Например:

Строка 1	Строка 2	Вывод
абвгабвг	аб	2
стстсап	стс	2

2. Системные основы разработки требований к сложным комплексам программ. Формализация эталонов требований и характеристик комплекса программ. Формирование требований компонентов и модулей путем декомпозиции функций комплексов программ.

Набор тестовых сценариев записать в виде таблицы.

3. Тестирование по принципу «белого ящика».

В Древней Греции (II в. до н.э.) был известен шифр, называемый "квадрат Полибия". Шифровальная таблица представляла собой квадрат с пятью столбцами и пятью строками, которые нумеровались цифрами от 1 до 5. В каждую клетку такого квадрата записывалась одна буква. В результате каждой букве соответствовала пара чисел, и шифрование сводилось к замене буквы парой чисел. Для латинского алфавита квадрат Полибия имеет вид:

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I, J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Пользуясь изложенным способом создать программу, которая:

а) зашифрует введенный текст и сохранит его в файл;

б) считает зашифрованный текст из файла и расшифрует данный текст.

4. Спроектировать тесты по принципу «белого ящика» для программы, разработанной в задании №3. Выбрать несколько алгоритмов для тестирования и обозначить буквами или цифрами ветви этих алгоритмов. Выписать пути алгоритма, которые должны быть проверены тестами для выбранного метода тестирования. Записать тесты, которые позволят пройти по путям алгоритма. Протестировать разработанную вами программу. Результаты оформить в виде таблиц:

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
…	…	…	…

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями
2. Выполните задание
3. Представьте выполненную работу в виде отчета

Ход работы

Теоретические вопросы

Оценка стоимости и причины ошибок в программном обеспечении. Виды и методы тестирования. Понятие теста. Требования к разработке тестовых сценариев. Правила разработки тестовых сценариев

Задание № 1. Написать программу решения квадратного уравнения ax2 + bx + c = 0.

Задание №2. Найти минимальный набор тестов для программы нахождения вещественных корней квадратного уравнения ax2 + bx + c = 0. Решение представлено в таблице.

Номер теста	a	b	c	Ожидаемый результат	Что проверяется
1	2	-5	2	$x_1=2$, $x_2=0,5$	Случай вещественных корней
2	3	2	5	Сообщение	Случай комплексных корней
3	3	-12	0	$x_1=4$, $x_2=0$	Нулевой корень
4	0	0	10	Сообщение	Неразрешимое уравнение
5	0	0	0	Сообщение	Неразрешимое уравнение
6	0	5	17	Сообщение	Неквадратное уравнение
7	9	0	0	$x_1=x_2=0$	Нулевые корни

Для этой программы предлагается минимальный набор функциональных тестов, исходя из 7 классов выходных данных. Заповеди по отладки программного средства, предложенные Г. Майерсом.

1. Считайте тестирование ключевой задачей разработки ПС, поручайте его самым квалифицированным и одаренным программистам, нежелательно тестировать свою собственную программу.

2. Хорош тот тест, для которого высока вероятность обнаружить ошибку, а не тот, который демонстрирует правильную работу программы.

3. Готовьте тесты как для правильных, так и для неправильных данных.

4. Документируйте пропуск тестов через компьютер, детально изучайте результаты каждого теста, избегайте тестов, пропуск которых нельзя повторить.

5. Каждый модуль подключайте к программе только один раз, никогда не изменяйте программу, чтобы облегчить ее тестирование.

6. Пропускайте заново все тесты, связанные с проверкой работы какой-либо программы ПС или ее взаимодействия с другими программами, если в нее были внесены изменения (например, в результате устранения ошибки).

Форма представления результата:
Оформить отчет по выполненным заданиям.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные

ошибки, при этом ход анализа должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при анализе и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 3 Использование инструментария анализа качества Цель:

получить навыки использования инструментария анализа качества.

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства

Задание:

1. Написать программу, генерирующую массив вещественных чисел в диапазоне от –10 до 10 и определяющую все минимальные положительные элементы.

2. Оценить эффективность разработанной программы:

	Исходная программа		Улучшенная программа	
	Недостатки	Количественная оценка	Улучшения	Количественная оценка
Время выполнения				
Оперативная память				
Внешняя память				

3. Оценить качество разработанной программы:

	Правильность	Универсальность	Проверяемость	Точность результатов
Недостатки				
Оценка				

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями

2. Выполните задание

3. Представьте выполненную работу в виде отчета

Ход работы

Теоретические вопросы

Общие требования к качеству функционирования сложных программных комплексов. Требования к характеристикам качества сложных программных комплексов. Требования к эффективности использования ресурсов ЭВМ программным комплексом в реальном времени. Проверка корректности функциональных требований к сложным комплексам программ.

Форма представления результата:

Оформить отчет по выполненным заданиям.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход анализа должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при анализе и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 4 Анализ и обеспечение обработкиисключительных ситуаций

Цель: получение навыков анализа и обеспечения обработки исключительных ситуаций.

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства.

Задание:

1. Написать программу, в которой обрабатываются следующие исключительные ситуации: "отрицательное значение возраста" и "год рождения больше текущего".

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями

2. Выполните задание

3. Представьте выполненную работу в виде отчета

Ход работы:

Исключение — это событие при выполнении программы, которое приводит к её ненормальному или неправильному поведению.

Существует два вида исключений:

1.Аппаратные (структурные, SE-Structured Exception), которые генерируются процессором. К ним относятся, например,

- деление на 0;
- выход за границы массива;
- обращение к невыделенной памяти;
- переполнение разрядной сетки.

2.Программные, генерируемые операционной системой и прикладными программами – возникают тогда, когда программа их явно инициирует. Когда встречается аномальная ситуация, та часть программы, которая ее обнаружила, может сгенерировать, или возбудить, исключение.

Механизм структурной обработки исключений позволяет однотипно обрабатывать как программные, так и аппаратные исключения.

Обработка программных исключений

Фундаментальная идея обработки исключительных ситуаций состоит в том, что функция, обнаружившая проблему, но не знающая как её решить, генерирует исключение в надежде, что вызвавшая её (непосредственно или косвенно) функция сможет решить возникшую проблему. Функция, которая может решать проблемы данного типа, указывает, что она перехватывает такие исключения.

Обработка исключительных ситуаций самой программой заключается в том, что при возникновении исключительной ситуации управление передаётся некоторому заранее определённому обработчику — блоку кода, процедуре, функции, которые выполняют необходимые действия.

Форма представления результата:

Оформить отчет по лабораторной работе.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход анализа должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при анализе и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 5 Функциональное тестирование

Цель: описать набор тестовых сценариев для верификации ПО. Оптимизировать тестовый набор. Научиться составлять спецификацию для разработки тестов

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации;

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства

Задание:

1. Протестировать функциональность формы приема заявок, требования к которой предоставлены в следующей таблице:

Элемент	Тип элемента	Требования
Тип обращения	combobox	Набор данных: 1. Консультация 2. Проведение тестирования 3. Размещение рекламы 4. Ошибка на сайте * - на процесс выполнения операции приема заявок не влияет.
Контактное лицо	editbox	1. Обязательное для заполнения 2. Максимально 25 символов 3. Использование цифр и спец символов не допускается
Контактный телефон	editbox	1. Обязательное для заполнения 2. Допустимые символы "+" и цифры 3. "+" можно использовать только в начале номера 4. Допустимые форматы: • начинается с плюса - 11-15 цифр • +31612361264 • +375291438884 • без плюса - 5-10 цифр, например: • 0613261264 • 2925167
Сообщение	text area	1. Обязательное для заполнения 2. Максимальная длина 1024 символа
Отправить	button	Состояние: 1. По умолчанию - не активна (Disabled) 2. После заполнения обязательных полей становится активна (Enabled) Действия после нажатия 1. Если введенные данные корректны - отправка сообщения 2. Если введенные данные НЕ корректны - валидационное сообщение

Порядок выполнения работы:

1. Анализ требований

Читаем, анализируем требования и выделяем для себя следующие нюансы:

- какие из полей обязательные для заполнения?
- имеют ли поля ограничения по длине или по размерности (границы)?
- какие из полей имеют специальные форматы?

2. Определение набора тестовых данных

Отталкиваясь от требований к полям, используя техники тест дизайна начинаем определение набора тестовых данных:

- в зависимости от того обязательное поле или нет, определим какие поля необходимо проверить на пустое значение, так как оно может вызывать ошибку (В результирующей таблице **оранжевый** цвет)
- т.к. исчерпывающее тестирование не представляется возможным *из-за огромного числа всевозможных комбинаций значений*, в первую очередь необходимо определить **минимальный набор данных**. Это можно сделать используя такие **техники**, как **EP** и **BVA**. (В результирующей таблице **голубой** цвет)
- На форме присутствует поле, имеющее составной тип (цифры используются совместно с символами), обладает специальным форматом данных и поэтому выделение тестовых данных для него - это достаточно трудоемкая задача. *В пределах данной работы ограничимся только простой проверкой форматов и основных требований описанных в форме приема заявок*.
- По завершению генерации данных используя стандартные техники, можно добавить некоторое количество значений на основании личного опыта (**техника EG**) - это будет использование спец. символов, очень длинных строк, разных форматов данных, регистров в строках (Upper, Lowwer, Mixed cases), отрицательные и нулевые значения, кейворды Null - NaN - Infinity и т.д. Сюда можно включить все, что вы полагаете может вывести приложение из строя (в результирующей таблице **фиолетовый** цвет)

Ход работы:

Функциональное тестирование рассматривает заранее указанное поведение и основывается на анализе спецификаций функциональности компонента или системы в целом.

Функциональные тесты основываются на функциях, выполняемых системой, и могут проводиться на всех уровнях тестирования (компонентном, интеграционном, системном, приемочном). Как правило, эти функции описываются в требованиях, функциональных спецификациях или в виде случаев использования системы (**use cases**).

Тестирование функциональности может проводиться в двух аспектах:

- требования
- бизнес-процессы

Тестирование в перспективе «требования» использует спецификацию функциональных требований к системе как основу для дизайна тестовых случаев (Test Cases). В этом случае необходимо сделать список того, что будет тестироваться, а что нет, приоритезировать требования на основе рисков (если это не сделано в документе с требованиями), а на основе этого приоритезировать тестовые сценарии (test cases). Это позволит сфокусироваться и не упустить при тестировании наиболее важный функционал.
Тестирование в перспективе «бизнес-процессы» использует знание этих самых бизнес-процессов, которые описывают сценарии ежедневного использования системы. В этой перспективе тестовые сценарии (**test scripts**), как правило, основываются на случаях использования системы (use cases).

Преимущества функционального тестирования:

- имитирует фактическое использование системы;

Недостатки функционального тестирования:

- возможность упущения логических ошибок в программном обеспечении;
- вероятность избыточного тестирования.

Достаточно распространенной является автоматизация функционального тестирования.

План разработки тест кейсов предлагается следующий:

1. Анализ требований.
2. Определение набора тестовых данных на основании **EP**, **BVA**, **EG**.
3. Разработка шаблона теста на основании **CE.**
4. Написание тест кейсов на основании первоначальных требований, тестовых данных и шагов теста.

Форма представления результата:

Оформить отчет по лабораторной работе в виде набора тестовых сценариев.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход анализа должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при анализе и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 6 Тестирование безопасности

Цель: получение навыков тестирования безопасности информационной системы.

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства

Задание:

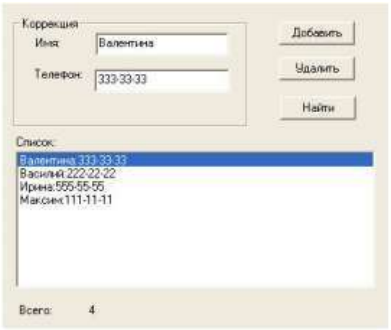
1. Изучите и опишите одно из средств выявления уязвимостей: Таблица 1. Обзор средств выявления уязвимостей, работающих на уровне кода

Наименование средства	Назначение	Поддерживаемые языки программирования	Примечание
Иностранные средства выявления уязвимостей			
Its4	Статически просматривает исходный код для обнаружения потенциальных уязвимостей защиты	C/c++	Отмечает вызовы потенциально опасных функций, таких, как strcpy/memcpy, и выполняет поверхностный семантический анализ, пытаясь оценить, насколько опасен такой код, а также дает советы по его улучшению
Rats(rough auditing tool for security)	Просматривает исходный текст, находя потенциально опасные	C/c++, php, perl, python	Использует сочетание проверок надежности защиты от семантических проверок в its4 до

Наименование средства	Назначение	Поддерживаемые языки программирования	Примечание
	обращения к функциям		глубокого семантического анализа в поисках дефектов, способных привести к переполнению буфера, полученных из mops
Flawfinder	Просматривает исходный текст, находя потенциально опасные обращения к функциям	C/c++	Выполняет поиск функций, которые чаще всего используются некорректно, присваивает им коэффициенты риска (опираясь на такую информацию, как передаваемые параметры) и составляет список потенциально уязвимых мест, упорядочивая их по степени риска
Flexelint (pc-lint)	Производит семантический анализ исходного кода, анализ потоков данных и управления	C/c++	В конце работы выдаются сообщения нескольких основных типов: – возможен нулевой указатель – проблемы с выделением памяти (например, нет free() после malloc()) – проблемный поток управления (например, недостижимый код), – возможно переполнение буфера, арифметическое переполнение, – предупреждения о плохом и потенциально опасном стиле кода
Parasoft c++ test	Формирование тестов анализа уязвимостей на уровне метода, класса, файла и проекта	C++	Генерирует тестовый код, вызывая для его подготовки компилятор visual c++
Coverity	Используется для выявления и исправления дефектов безопасности и качества в приложениях критического назначения	C/c++, java	Способен с минимальной положительной погрешностью обрабатывать десятки миллионов строк кода, обеспечивая 100-процентное покрытие трассы

Klocwork k7	Предназначен для автоматизированного статического анализа кода, выявления и предотвращения дефектов программного обеспечения и проблем безопасности	C/c++, java	Выявляет коренные причины недостатков качества и безопасности программного обеспечения
Codesurfer	Может применяться для поиска ошибок в исходном коде, для улучшения понимания исходного кода	C/c++	Позволяет проводить анализ указателей, использовать и определять переменные, зависимости данных, строить графы вызовов
Fxcop	Способен обнаружить более 200 недочетов (или ошибок) в следующих областях: – архитектура библиотеки; – правила именования; – производительность; – безопасность	C/c++	Откомпилированный код проверяется с помощью механизмов рефлексии, парсинга msil и анализа графа вызовов
Qaudit	Быстрый анализ исходных файлов на наличие переполнения буфера, ошибок форматной строки, запросов исполняемых вызовов, переменных среды, и функций, имеющих проблемы защиты	C/c++	Написать на интерпретируемом языке perl, прост в использовании
Российские средства выявления уязвимостей			
Ак-вс	Автоматизированный анализ исходных текстов, с целью выявления потенциально опасных сигнатур	C/c++, java, pascal, c#, php, assembler	Позволяет проводить статический анализ исходных текстов, динамический анализ, имеет базы сигнатур для каждого из поддерживаемых языков программирования
Аист-с	Автоматизированный анализ исходных текстов	C/c++	Позволяет проводить статический анализ исходных текстов
Ксаит	Автоматизированный анализ исходных текстов	C/c++	Позволяет проводить статический анализ исходных текстов
Uca	Предназначено для выявления потенциально опасных сигнатур	C/c++, pascal, perl, plm	Имеет базы сигнатур для каждого из поддерживаемых языков программирования
Viva64	Помогает отслеживать в исходном коде потенциально опасные фрагменты, связанные с переходом от 32-битных систем к 64-битным	C/c++	Помогает писать корректный и оптимизированный код для 64-битных систем

2. Разработать приложение, интерфейс которого представлен на рисунке.



3. Добавить в программу форму авторизации по имени и паролю.

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями

2. Выполните задание

3. Представьте выполненную работу в виде отчета

Ход работы:

Тестирование восстановления. Тестирование безопасности. Технологии тестирования безопасности. Тестирование безопасности – оценка уязвимости программного обеспечения к различным атакам. Компьютерные системы очень часто являются мишенью незаконного проникновения. Под проникновением понимается широкий диапазон действий: попытки хакеров проникнуть в систему из спортивного интереса, месть рассерженных служащих, взлом мошенниками для незаконной наживы. Тестирование безопасности проверяет фактическую реакцию защитных механизмов, встроенных в систему, на проникновение. В ходе тестирования безопасности испытатель играет роль взломщика. Ему разрешено все: – попытки узнать пароль с помощью внешних средств; – атака системы с помощью специальных утилит, анализирующих защиты; – подавление, ошеломление системы (в надежде, что она откажется обслуживать других клиентов); – целенаправленное введение ошибок в надежде проникнуть в систему в ходе восстановления; – просмотр несекретных данных в надежде найти ключ для входа в систему. При неограниченном времени и ресурсах хорошее тестирование безопасности взломает любую систему. Задача проектировщика системы – сделать цену проникновения более высокой, чем цена получаемой в результате информации.

Форма представления результата:

Должны быть приведены исходные коды программы, результаты тестирования безопасности ПО.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет

исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход анализа должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при анализе и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 7 Нагрузочное тестирование, стрессовоетестирование

Цель: С помощью систем нагрузочного тестирования определить производительность web-серверов Apache и Nginx, добиться отказа в обслуживании

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства

Задание:

1. Нагрузочное тестирование веб-сервера с Apache

Для тестирования используются 2 машины – одна с установленным и работающим Apache, вторая будет отсылать запросы и делать выводы о производительности web-сервера.

Тестирование на PHP-запросы:

2. Определить максимальное число параллельных запросов, при котором сервер нас не будет блокировать.
3. Провести тест при использовании максимального числа запросов.

Тестирование на HTML-запросы:

4. Определить максимальное число параллельных запросов
5. Провести тест при использовании максимального числа запросов.
6. Провести сравнение результатов и сформировать выводы.

Итоговая таблица сравнения

		Максимальное число запросов	Запросы/сек	Время, затрачиваем
Apache	PHP			
	HTML			
LB + Apache	PHP			
	HTML			
Nginx	PHP			
	HTML			
LB + Nginx	PHP			
	HTML			

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями

2. Выполните задание

3. Представьте выполненную работу в виде отчета

Ход работы:

Нагрузочное тестирование - это автоматизированный процесс, имитирующий одновременную работу определенного количества пользователей на каком-либо общем ресурсе.

Приложение - тестируемое прикладное программное обеспечение.

Цели нагрузочного тестирования

1. Оценка работоспособности и производительности приложения на этапе разработки и при передаче в эксплуатацию
2. Оценка работоспособности и производительности приложения на этапе выпуска новых релизов, патч-сетов
3. Оптимизация производительности приложения, включая оптимизацию программного кода и настройку серверов
4. Подбор соответствующей для данного приложения аппаратной и программной платформы, а также нужной конфигурации сервера.

Использование Apache benchmark tool

Apache benchmark — одна из самых простых утилит, которая применяется для нагрузочного тестирования сайта. Идет в комплекте с веб сервером Apache, в первоначальной настройке не нуждается. Задача, которая ставится перед Apache benchmark

— показать, какое количество запросов сможет выдержать веб сервер и как быстро он их обработает.

Пример нагрузки на сервер в 5000 последовательных запросов:

<ab -n 5000 http://192.168.1.116/index.html>

Пример нагрузки на сервер в 5000 запросов, но 500 из них будут направлены на сервер одновременно (параллельные запросы):

<ab -n 5000 -c 500 http://192.168.1.116/index.html>

Для выполнения лабораторной работы меняйте значения после -n и -c, чтобы узнать с каким количеством запросов может справиться сервер. На этих примерах выполнены HTML- запросы, для тестирования на PHP-запросы измените цель на index.php.

Использование httperf

Еще одно консольное приложение, используемое также для создания нужного количества параллельных запросов - httperf.

Его отличие от ab в том, что httperf посылает запросы согласно своим настройкам, невзирая на то, отвечает сервер на них или уже нет. Таким образом можно определить не только какую максимальную нагрузку может выдержать сервер, но и как будет себя вести сервер в момент, когда нагрузка достигла своего пика

Пример запуска 100 запросов от 10 посетителей параллельно:

httperf --port 80 --server <domain> --uri=/ --num-conns=100 --rate=10

Использование Siege

Установка:

sudo apt-get install siege

Количество запросов не лимитируется, но можно задавать время в течение которого выполнять тестирование.

Пример: 5 пользователей, которые безостановочно загружают главную страницу в течении одной минуты.

```
siege -c 5 -b -t 1m ip-адрес
```

Балансировщик нагрузки Nginx

Балансировка нагрузки (англ. load balancing) — метод распределения заданий между несколькими сетевыми устройствами (например, серверами) с целью оптимизации использования ресурсов, сокращения времени обслуживания запросов, горизонтального масштабирования кластера (динамическое добавление/удаление устройств), а также обеспечения отказоустойчивости (резервирования).

Установка:

```
sudo apt-get install nginx
```

Настройка:

```
sudo nano /etc/nginx/sites-available/default

upstream web_backend {

   server 192.168.1.113;

   server 192.168.1.114;

}

server {

   listen 80;

   location / {

      proxy_set_header XForwardedFor $proxy_add_x_forwarded_for;

      proxy_pass http://web_backend;

   }

}
```

После настройки перезапускаем Nginx

```
sudo service nginx reload
```

Методы балансировки нагрузки (описываются в начале секции upstream):

- ip_hash - согласно этому методу запросы от одного и того же клиента будут всегда отправляться на один и тот же backend сервер на основе информации об ip адресе клиента. Не совместим с параметром weight.
- least_conn - запросы будут отправляться на сервер с наименьшим количеством активных соединений.
- round-robin – режим по умолчанию. То есть если вы не задали ни один из вышеупомянутых способов балансировки - запросы будут доставляться по очереди на все сервера в равной степени.

Форма представления результата:

Оформить отчет по лабораторной работе.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход решения задачи должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при решении задачи и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 8 Тестирование интеграции

Цель: получение навыков тестирования интеграции

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства

Задание:

1. Разработать приложение, состоящее из трех модулей:

1) главный модуль, считывающий из текстового файла координаты точек на плоскости;

2) модуль, содержащий функции расчета расстояния между двумя точками;

3) модуль, содержащий функцию, определяющую треугольник с максимальной площадью.

2. Описать этапы нисходящего проектирования разработанного приложения.
3. Описать этапы восходящего проектирования разработанного приложений.

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями, используя дополнительные источники литературы

2. Выполните задание

3. Представьте выполненную работу в виде отчета

Ход работы:

Теоретические вопросы

Особенности тестирования интеграции.
Методы интеграционного тестирования.
Нисходящее тестирование интеграции.
Восходящее тестирование интеграции.
Сравнение нисходящего и восходящего тестирования интеграции

Форма представления результата:

Оформить отчет по лабораторной работе.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется:: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход решения задачи должен быть верным.

Оценка «неудовлетворительно» выставляется:: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при решении задачи и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 9 Конфигурационное тестирование

Цель: получение навыков проведения конфигурационного тестирования

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями
2. Выполните задание
3. Представьте выполненную работу в виде отчета

Ход работы:

Теоретическая часть

Особенности конфигурационного тестирования. Конфигурационное тестирование (Configuration testing). Проверяется работоспособность при различных конфигурациях, предполагает тестирование работы системы на различных платформах: различных вариантах аппаратной конфигурации, версиях операционной системы и окружения.

Задание:

1. Дана структура с именем ZNAK, состоящая из полей:

– фамилия, имя;

– знак Зодиака;

– дата рождения (массив из трех чисел).

Написать программу, которая выполняет следующие действия:

– ввод с клавиатуры данных в массив, состоящий из 8 элементов типа ZNAK, и занесение их в файл данных;

– чтение данных из файла и вывод их на экран;

– вывод на экран информации о людях, родившихся в месяц, значение которого введено с клавиатуры (если таких нет – вывести об этом сообщение);

– список должен быть упорядочен по знакам Зодиака.

2. Описать и обосновать итоги тестирования работы разработанного приложения на различных платформах: различных вариантах аппаратной конфигурации, версиях операционной системы и окружения.

Форма представления результата:

Оформить отчет по лабораторной работе.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход решения задачи должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при решении задачи и примеров допущено более 5 вычислительных ошибок.

Лабораторное занятие № 10 Тестирование установки

Цель: получение навыков тестирования установки.

Выполнив работу, Вы будете:

уметь:

- У1. осуществлять постановку задач по обработке информации;
- У4. использовать алгоритмы обработки информации для различных приложений;
- У12. разрабатывать проектную документацию на эксплуатацию информационной системы;
- У13. использовать стандарты при оформлении программной документации.

Материальное обеспечение:

Мультимедийные средства хранения, передачи и представления информации. Учебно-методическая документация, дидактические средства

Задание:

1. Разработать web-приложение, интерфейс которого определен преподавателем.
2. Провести комплексное тестирование разработанного приложения.

Порядок выполнения работы:

1. Ознакомьтесь с краткими теоретическими сведениями

2. Выполните задание

3. Представьте выполненную работу в виде отчета

Ход работы:

Тестирование установки (инсталляционное тестирование) позволяет удостовериться в том, что ПО корректно устанавливается и настраивается, накат новых версий происходит без ошибок, а также есть возможность деинсталлировать и удалить данное ПО.

Форма представления результата:

Оформить отчет по лабораторной работе.

Критерии оценки:

Оценка «отлично» выставляется: вся работа выполнена безошибочно и нет исправлений;

Оценка «хорошо» выставляется: допущены 1-2 вычислительные ошибки.

Оценка «удовлетворительно» выставляется: допущены ошибки в ходе решения задачи при правильном выполнении всех остальных заданий или допущены 3-4 вычислительные ошибки, при этом ход решения задачи должен быть верным.

Оценка «неудовлетворительно» выставляется: допущены ошибки в ходе решения задачи и хотя бы одна вычислительная ошибка или при решении задачи и примеров допущено более 5 вычислительных ошибок.