Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

Направление подготовки/специальность: 01.03.02 Прикладная математика и информатика

Наименование образовательной программы: Математическое и программное обеспечение вычислительных машин и компьютерных сетей

Уровень образования: высшее образование - бакалавриат

Форма обучения: Очная

Оценочные материалы
по дисциплине
Методы контроля программ

Москва
2023

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ РАЗРАБОТАЛ:

Преподаватель	Подписано электронной подписью ФГБОУ ВО «НИУ «МЭИ» Сведения о владельце ЦЭП МЭИ Владелец: Куриленко И.Е. Идентификатор: R73df8d6c-KurilenkoIY-5c331b90	И.Е. Куриленко
(должность)	(подпись)	(расшифровка подписи)

СОГЛАСОВАНО:

Руководитель образовательной программы	Подписано электронной подписью ФГБОУ ВО «НИУ «МЭИ» Сведения о владельце ЦЭП МЭИ Владелец: Маран М.М. Идентификатор: R7be141f2-MaranMM-804b01e2	М.М. Маран
(должность, ученая степень, ученое звание)	(подпись)	(расшифровка подписи)
Заведующий выпускающей кафедры	Подписано электронной подписью ФГБОУ ВО «НИУ «МЭИ» Сведения о владельце ЦЭП МЭИ Владелец: Варшавский П.Р. Идентификатор: R9a563c96-VarshavskyPR-efb4bbd	П.Р. Варшавский
(должность, ученая степень, ученое звание)	(подпись)	(расшифровка подписи)

ОБЩАЯ ЧАСТЬ

Оценочные материалы по дисциплине предназначены для оценки: достижения обучающимися запланированных результатов обучения по дисциплине, этапа формирования запланированных компетенций и уровня освоения дисциплины.

Оценочные материалы по дисциплине включают оценочные средства для проведения мероприятий текущего контроля успеваемости и промежуточной аттестации.

Формируемые у обучающегося компетенции:

1. ПК-1 Способен выполнять все этапы жизненного цикла программного обеспечения
   ИД-5 Определяет методы тестирования и умеет проводить все виды контроля программ
   ИД-6 Демонстрирует знания видов сопровождения и умеет применять их на практике

и включает:

для текущего контроля успеваемости:

Форма реализации: Компьютерное задание

1. Автоматизация нагрузочного тестирования (Лабораторная работа)
2. Автоматизация тестирования приложений с графическим интерфейсом (Лабораторная работа)
3. Автоматизация тестирования Web-приложений (Лабораторная работа)
4. Модульное тестирование (Лабораторная работа)
5. Мониторинг производительности и контроль расхода ресурсов (Лабораторная работа)

Форма реализации: Устная форма

1. Организация тестирования программного обеспечения (Интервью)

БРС дисциплины

7 семестр

Раздел дисциплины	Веса контрольных мероприятий, %						
	Индекс КМ:	КМ-1	КМ-2	КМ-3	КМ-4	КМ-5	КМ-6
	Срок КМ:	4	8	10	12	14	16
Основы тестирования программного обеспечения.							
Тестирование программного обеспечения. Методика тестирования.		+					
Методы тестирования. Организация тестирования.		+					
Системы отслеживания ошибок. Система управления задачами и заявками.		+					
Организация процесса тестирования вручную		+					
Автоматизация тестирования.							
Модульное тестирование			+				

Автоматизация тестирования приложений через графический интерфейс пользователя			+			
Методы автоматизации тестирования web-приложений				+		
Методы автоматизации тестирования					+	
Автоматизация тестирования баз данных						+
Вес КМ:	10	20	20	20	20	10

$Общая часть/Для промежуточной аттестации$

СОДЕРЖАНИЕ ОЦЕНОЧНЫХ СРЕДСТВ ТЕКУЩЕГО КОНТРОЛЯ

I. Оценочные средства для оценки запланированных результатов обучения по дисциплине, соотнесенных с индикаторами достижения компетенций

Индекс компетенции	Индикатор	Запланированные результаты обучения по дисциплине	Контрольная точка
ПК-1	ИД-5$_{ПК-1}$ Определяет методы тестирования и умеет проводить все виды контроля программ	Знать: методику тестирования программных средств разных типов Уметь: разрабатывать тестовые сценарии применять на практике различные средства автоматизации тестирования работать в системах отслеживания ошибок и в системах управления задачами и заявками	Организация тестирования программного обеспечения (Интервью) Модульное тестирование (Лабораторная работа) Автоматизация тестирования приложений с графическим интерфейсом (Лабораторная работа) Автоматизация нагрузочного тестирования (Лабораторная работа)
ПК-1	ИД-6$_{ПК-1}$ Демонстрирует знания видов сопровождения и умеет применять их на практике	Знать: современные технологии, инструментарии и языковые средства программирования, использующиеся для создания системного ПО Уметь: находить и использовать в работе средства	Модульное тестирование (Лабораторная работа) Автоматизация тестирования Web-приложений (Лабораторная работа) Мониторинг производительности и контроль расхода ресурсов (Лабораторная работа)

		автоматизации тестирования с открытым исходным кодом осуществлять систематическое тестирование сложных программных комплексов	

II. Содержание оценочных средств. Шкала и критерии оценивания

КМ-1. Организация тестирования программного обеспечения

Формы реализации: Устная форма
Тип контрольного мероприятия: Интервью
Вес контрольного мероприятия в БРС: 10
Процедура проведения контрольного мероприятия: Устное интервью в форме вопрос/ответ.

Краткое содержание задания:
Оценить понимание задачи и методики тестирования программного обеспечения. Оценить знание методов тестирования. Оценить понимание рекомендуемой организации процесса тестирования тестирования. Знание методики организация процесса тестирования вручную. Понимание технологии модульного тестирования.

Контрольные вопросы/задания:

Знать: методику тестирования программных средств разных типов	1.Перечислите виды тестирования ПО. 2.Перечислите методы тестирования ПО. 3.Что предполагает систематическое тестирование?

Описание шкалы оценивания:

Оценка: 5
Нижний порог выполнения задания в процентах: 70
Описание характеристики выполнения знания: Оценка "отлично" выставляется если задание выполнено в полном объеме или выполнено преимущественно верно

Оценка: 4
Нижний порог выполнения задания в процентах: 60
Описание характеристики выполнения знания: Оценка "хорошо" выставляется если большинство вопросов раскрыто. выбрано верное направление для решения задач

Оценка: 3
Нижний порог выполнения задания в процентах: 50
Описание характеристики выполнения знания: Оценка "удовлетворительно" выставляется если задание преимущественно выполнено

КМ-2. Модульное тестирование

Формы реализации: Компьютерное задание
Тип контрольного мероприятия: Лабораторная работа
Вес контрольного мероприятия в БРС: 20
Процедура проведения контрольного мероприятия: Проводится в форме выполнения и защиты лабораторной работы.

Краткое содержание задания:
Разработка модульных тестов для кода, разработанного на языках C++, C#, Java

1.1 - Разработать модульный тест с применением библиотеки CPPUNIT для программного кода, разработанного на языке C++

Цель работы

Научиться разрабатывать модульные тесты для кода, написанного на языке C++ с применением библиотеки CPPUNIT.

Порядок выполнения работы

1. Создать с помощью Microsoft Visual Studio 2013 консольный проект C++ (Console Application).
2. Создать несколько классов, которые будут имитировать тестируемую логику.
3. Разработать чек-лист (Список проверок).
4. Подключить к проекту CPPUNIT.
5. Создать в проекте новый контейнерный тестовый класс.
6. Описать в этом классе функции setUp() и tearDown().
7. Разработать не менее пяти тестирующих функций. При разработке этих функций следует активно применять макросы CPPUNIT_ASSERT.
8. Применить макросы CPPUNIT_ASSERT_THROW и CPPUNIT_ASSERT_NO_THROW для контроля генерации исключений.
9. Создать карту-описатель контейнерного теста с помощью макроса CPPUNIT_TEST_SUITE.
10. Внести в карту-описатель тестирующие функции с помощью макроса CPPUNIT_TEST.
11. Добавить в функцию main объект TestRunner.
12. Скомпилировать и запустить проект. Посмотреть на результат теста.
13. Настроить автоматический запуск теста при компиляции.
14. Внести в тестируемые классы изменения, приводящие к ошибкам.
15. Скомпилировать и запустить проект. Посмотреть, пойманы ли ошибки модульным тестом.

1.2 - Разработка модульного теста для кода на языке C++ с применением Microsoft C++ Unit Test Framework

Цель работы

Научиться разрабатывать модульные тесты для кода, написанного на языке C++ с применением C++ Unit Test Framework.

Порядок выполнения работы

1. Создать с помощью Microsoft Visual Studio 2013 консольный проект C++ (Console Application).
2. Создать несколько классов, которые будут имитировать тестируемую логику.
3. Разработать чек-лист (Список проверок).
4. Создать в проекте новый тестовый класс.
5. Разработать не менее пяти тестирующих функций. При разработке этих функций следует активно применять методы статического класса Assert.
6. Реализовать функции TEST_CLASS_INITIALIZE и TEST_CLASS_CLEANUP.
7. Скомпилировать проект. С помощью меню в Test Explorer запустить тесты. Посмотреть на результат теста.
8. Внести в тестируемые классы изменения, приводящие к ошибкам.
9. Скомпилировать проект. С помощью меню в Test Explorer запустить тесты. Посмотреть, пойманы ли ошибки модульным тестом.

2.1 - NUnit

Цель работы

Научиться разрабатывать модульные тесты для кода на языке C# с применением библиотеки NUnit. Разработать модульный тест с применением NUnit.

Порядок выполнения работы

1. Создать с помощью MS Visual Studio консольный проект C# (Console Application).
2. Подключить к проекту библиотеку NUnit.Framework.dll.
3. Создать несколько классов, которые будут имитировать тестируемую логику.
4. Создать в проекте новый тестовый класс, пометить его с помощью атрибута [TestFixture].
5. Описать в этом классе функции setUp() и tearDown() и отметить их атрибутами [SetUp], [TearDown].
6. Разработать не менее пяти тестирующих функций, отметить их соответствующими атрибутами [Test]. При разработке этих функций следует активно применять функции класса Assert — AreEquals, IsTrue, IsNull.
7. Разработать тестовые методы со спецификацией ожидаемых исключений.
8. Запустить проверку разработанного тестового класса.
9. Скомпилировать и запустить проект. Посмотреть на результат теста.
10. Внести в тестируемые классы изменения, приводящие к ошибкам.
11. Скомпилировать и запустить проект. Посмотреть, пойманы ли ошибки модульным тестом.

2.2 - MSUnit

Цель работы

Научиться разрабатывать модульные тесты для кода на языке C# с применением библиотеки MSUnit. Разработать модульный тест с применением MSUnit.

Порядок выполнения работы

1. Создать с помощью MS Visual Studio проект консольного приложения C# (Console Application).
2. Создать несколько интерфейсов, для классов, которые будут имитировать тестируемую логику. Создать классы-реализации этих интерфейсов.
3. Сгенерировать тестовый проект с модульными тестами.
4. Реализовать не менее пяти тестирующих функций. При разработке этих функций следует активно применять функции класса Assert.
5. Разработать тестовый метод со спецификацией ожидаемых исключений.
6. Запустить модульные тесты. Проанализировать результаты запуска
7. Внести в тестируемые классы изменения, приводящие к ошибкам.
8. Скомпилировать и запустить проект. Посмотреть, пойманы ли ошибки модульным тестом.

3.1 - JUnit

Цель работы

Научиться разрабатывать модульные тесты для кода на языке Java с применением JUnit. Разработать модульный тест с применением библиотеки JUnit 4 для программного кода, разработанного на языке Java.

Подготовка к работе

1. Изучить механизмы аннотаций в Java

Порядок выполнения работы

1. Создать в среде разработки Eclipse (или IntelliJ IDEA) консольный проект Java (Java Application). Подключить к проекту библиотеку JUnit версии 4.
2. Создать несколько классов, которые будут имитировать тестируемую логику.
3. Создать в проекте новый тестовый класс.
4. Описать в этом классе функции setUp() и tearDown() и отметить их аннотациями @Before, @After.

5. Разработать не менее пяти тестирующих функций, отметить их соответствующими аннотациями. При разработке этих функций следует активно применять функции assertEquals, assertTrue, assertFalse.
6. Добавить в тестовые методы спецификацию ожидаемых исключений.
7. Разработать параметризованный тест.
8. Запустить проверку разработанного тестового класса.
9. Скомпилировать и запустить проект. Посмотреть на результат теста.
10. Внести в тестируемые классы изменения, приводящие к ошибкам.
11. Скомпилировать и запустить проект. Посмотреть, пойманы ли ошибки модульным тестом.
12. Добавить к одному из тестовых методов спецификацию ожидаемого времени работы
13. Сформировать отчет о проделанной работе

Контрольные вопросы/задания:

Знать: современные технологии, инструментарии и языковые средства программирования, использующиеся для создания системного ПО	1.Что такое параметризованный тест? 2.Зачем нужно обеспечивать фикстуру? 3.Можно ли с помощью модульных тестов измерять производительность?
Уметь: применять на практике различные средства автоматизации тестирования	1.Как в JUnit выполняется оформление тестов? 2.Зачем нужны атрибуты [SetUp], [TearDown] в NUnit.

Описание шкалы оценивания:

Оценка: 5
Нижний порог выполнения задания в процентах: 70
Описание характеристики выполнения знания: Оценка "отлично" выставляется если задание выполнено в полном объеме или выполнено преимущественно верно

Оценка: 4
Нижний порог выполнения задания в процентах: 60
Описание характеристики выполнения знания: Оценка "хорошо" выставляется если большинство вопросов раскрыто. выбрано верное направление для решения задач

Оценка: 3
Нижний порог выполнения задания в процентах: 50
Описание характеристики выполнения знания: Оценка "удовлетворительно" выставляется если задание преимущественно выполнено

КМ-3. Автоматизация тестирования приложений с графическим интерфейсом

Формы реализации: Компьютерное задание
Тип контрольного мероприятия: Лабораторная работа
Вес контрольного мероприятия в БРС: 20
Процедура проведения контрольного мероприятия: Проводится в форме выполнения и защиты лабораторной работы.

Краткое содержание задания:
Разработка тестов приложений с графическим интерфейсом с применением технологий Coded UI Test и IBM Rational Functional Tester

Тестирование с применением Coded UI Test.
Цель работы

Научиться автоматизировать тестирование приложений через графический интерфейс пользователя с применением технологии Coded UI Test.

Подготовка к работе

1. Изучить основные понятия технологии Coded UI Test

Порядок выполнения работы

1. Проанализировать графический интерфейс приложения, которое будет тестироваться.
2. Разработать список автоматизированных GUI тест-кейсов и подготовить тестовые данные.
3. Создать проект Coded UI Test в Microsoft Visual Studio.
4. Записать действия и получить генерируемый средой код теста.
5. Настроить проверки.
6. Запустить тест.
7. Подготовить отчет.

Тестирование с применением IBM Rational Functional Tester.

Цель работы

Научиться автоматизировать тестирование приложений через графический интерфейс пользователя с применением IBM Rational Functional Tester.

Краткое введение

IBM Rational Functional Tester предназначен для автоматизации:

- функционального и регрессивного тестирования приложений на платформах Windows и Linux;
- тестирования пользовательского интерфейса;
- тестирования, управ-ляемого данными.

Тесты могут создаваться путем написания кода, записи действий тестировщика, или комбинации этих действий. Тестовый скрипт является полноценной программой на ЯВУ (Java).

В составе скрипта есть декларативная (описательная) часть и императивная исполняемая последовательность, содержащая контрольные точки (точки верификации).

Подготовка к работе

1. Изучить материалы по продукту IBM Rational Functional Tester.

Порядок выполнения работы

1. Проанализировать графический интерфейс приложения, которое будет тестироваться.
2. Разработать список автоматизированных GUI тест-кейсов и подготовить тестовые данные.
3. Создать новый проект в IBM Rational Functional Tester.
4. Записать действия и получить генерируемый средой код теста.
5. Настроить точки верификации.
6. Запустить тест.
7. Подготовить отчет.

Контрольные вопросы/задания:

Уметь: разрабатывать тестовые сценарии	1.Опишите процесс создания теста в IBM Rational Functional Tester. 2.Что такое средство записи тестов? Какие изменения следует внести после работы такого средства?

Описание шкалы оценивания:

Оценка: 5
Нижний порог выполнения задания в процентах: 70
Описание характеристики выполнения знания: Оценка "отлично" выставляется если задание выполнено в полном объеме или выполнено преимущественно верно

Оценка: 4
Нижний порог выполнения задания в процентах: 60
Описание характеристики выполнения знания: Оценка "хорошо" выставляется если большинство вопросов раскрыто. выбрано верное направление для решения задач

Оценка: 3
Нижний порог выполнения задания в процентах: 50
Описание характеристики выполнения знания: Оценка "удовлетворительно" выставляется если задание преимущественно выполнено

КМ-4. Автоматизация тестирования Web-приложений

Формы реализации: Компьютерное задание
Тип контрольного мероприятия: Лабораторная работа
Вес контрольного мероприятия в БРС: 20
Процедура проведения контрольного мероприятия: Проводится в форме выполнения и защиты лабораторной работы.

Краткое содержание задания:
Разработка тестов web-приложений с применением Selenium

Применение Selenium
Цель работы
Научиться автоматизировать тестирование web-приложений с помощью средств автоматизации Selenium.

Краткое введение
Проект Selenium предоставляет комплект средств для автоматизированного управления браузерами, что позволяет автоматизировать тестирование web-приложений.

Подготовка к работе
1. Изучить основы разработки тестовых сценариев для автоматизации тестирования Web-приложений с применением Selenium
2. Изучить документацию по Selenium IDE
3. Установить Selenium IDE

Порядок выполнения работы
1. Сформулировать описание тест-кейса и критерии успеха
2. Запустить IDE
3. Создать новый сценарий
4. В режиме записи записать требуемую последовательность действий
5. Добавление проверки через контекстное меню
6. Проверить скрипт, при необходимости отредактировать команды
7. Сохранить тестовый сценарий
8. Запустить сценарий, убедиться в том, что сработали проверки

Контрольные вопросы/задания:

Уметь: находить и использовать в работе средства автоматизации тестирования с открытым исходным кодом	1.Какая команда Selenium используется для имитации ввода с клавиатуры? 2.Как Selenium определяет, что тестируемая страница полностью загрузилась?

Описание шкалы оценивания:

Оценка: 5
Нижний порог выполнения задания в процентах: 70
Описание характеристики выполнения знания: Оценка "отлично" выставляется если задание выполнено в полном объеме или выполнено преимущественно верно

Оценка: 4
Нижний порог выполнения задания в процентах: 60
Описание характеристики выполнения знания: Оценка "хорошо" выставляется если большинство вопросов раскрыто. выбрано верное направление для решения задач

Оценка: 3
Нижний порог выполнения задания в процентах: 50
Описание характеристики выполнения знания: Оценка "удовлетворительно" выставляется если задание преимущественно выполнено

КМ-5. Автоматизация нагрузочного тестирования

Формы реализации: Компьютерное задание
Тип контрольного мероприятия: Лабораторная работа
Вес контрольного мероприятия в БРС: 20
Процедура проведения контрольного мероприятия: Проводится в форме выполнения и защиты лабораторной работы.

Краткое содержание задания:
Разработка нагрузочного теста с применением IBM Rational Performance Tester

Нагрузочное тестирование
Цель работы
Научиться автоматизировать нагрузочное тестирование с применением IBM Rational Performance Tester.

Подготовка к работе
1. Изучить материалы по продукту IBM Rational Performance Tester.
2. Посмотреть презентацию 'Нагрузочное тестирование с помощью Rational Performance Tester'

Порядок выполнения работы
1. Создать демонстрационное приложение ASP .NET, реализовать функцию входа в систему
2. Запустить IBM Rational Performance Tester.
3. Нажать кнопку записи теста, выбрать тип проекта HTTP Recording
4. В открывшемся браузере открыть наше демонстрационное приложение ASP .NET и пройти сценарий входа в систему (ввести логин и пароль)
5. Изучить сгенерированный тест
6. Заменить значения логина и пароля пользователя на переменные, инициализируемые из пула данных
7. Добавить точки верификации

8. Запустить тест с одним виртуальным пользователем
9. Проанализировать отчет
10. Создать план нагрузки
11. Запустить тестирование с несколькими виртуальными пользователями
12. Подготовить отчет о проделанной работе

Контрольные вопросы/задания:

Уметь: работать в системах отслеживания ошибок и в системах управления задачами и заявками	1.Что такое план нагрузки? 2.Как правильно организовать тест, если необходимо проверить один сценарий для различных входных данных?

Описание шкалы оценивания:

Оценка: 5
Нижний порог выполнения задания в процентах: 70
Описание характеристики выполнения знания: Оценка "отлично" выставляется если задание выполнено в полном объеме или выполнено преимущественно верно

Оценка: 4
Нижний порог выполнения задания в процентах: 60
Описание характеристики выполнения знания: Оценка "хорошо" выставляется если большинство вопросов раскрыто. выбрано верное направление для решения задач

Оценка: 3
Нижний порог выполнения задания в процентах: 50
Описание характеристики выполнения знания: Оценка "удовлетворительно" выставляется если задание преимущественно выполнено

КМ-6. Мониторинг производительности и контроль расхода ресурсов

Формы реализации: Компьютерное задание
Тип контрольного мероприятия: Лабораторная работа
Вес контрольного мероприятия в БРС: 10
Процедура проведения контрольного мероприятия: Проводится в форме выполнения и защиты лабораторной работы.

Краткое содержание задания:
Тестирование расхода ресурсов и анализ производительности приложения.

Тестирование расхода ресурсов и анализ производительности приложения.
Цель работы
Научиться получать точную информацию об узких местах в производительности приложений, созданных на основе платформы .NET Framework. Получить навыки профилирования приложения в нескольких режимах, включая tracing (на основе подсчета числа вызовов), sampling (на основе подсчета времени исполнения) и построчный режим (детальный анализ производительности).

Порядок выполнения работы
1. Подготовить тестируемое приложение
    - Разработать с помощью Visual Studio консольное приложение
    - Реализовать в приложении два алгоритма сортировки (быструю сортировку и сортировку пузырьком)
    - Сортируемые данные должны загружаться из файла

- Создать несколько наборов несортированных тестовых данных (в т.ч. данные размером 1Кб, 2Кб)

2. Запустить приложение в режиме профилирования sampling. Определить наименее производительные функции.
3. Запустить приложение в режиме профилирования tracing. Получить результат. Определить узкие места в реализации программы.
4. Запустить приложение в режиме профилирования line by line. Получить информацию какие строки кода исполняются чаще всего.
5. Используя полученную информацию улучшить реализацию и проверить это с помощью повтора шагов 2-4

Контрольные вопросы/задания:

Уметь: осуществлять систематическое тестирование сложных программных комплексов	1.Опишите процедуру анализа расхода ресурсов оконным приложением. 2.Как выполняется локализация места распределения ресурса в процессе поиска ошибок в работе с памятью?

Описание шкалы оценивания:

Оценка: 5
Нижний порог выполнения задания в процентах: 70
Описание характеристики выполнения знания: Оценка "отлично" выставляется если задание выполнено в полном объеме или выполнено преимущественно верно

Оценка: 4
Нижний порог выполнения задания в процентах: 60
Описание характеристики выполнения знания: Оценка "хорошо" выставляется если большинство вопросов раскрыто. выбрано верное направление для решения задач

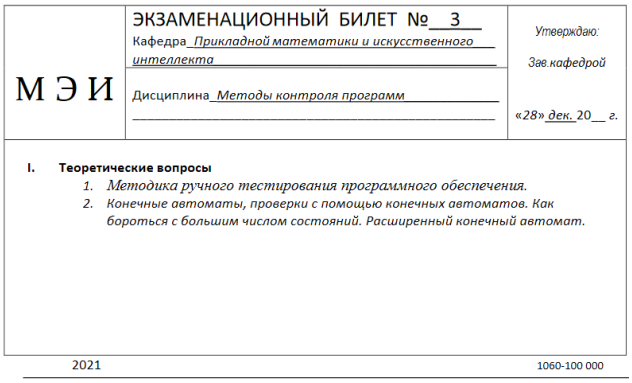
Оценка: 3
Нижний порог выполнения задания в процентах: 50
Описание характеристики выполнения знания: Оценка "удовлетворительно" выставляется если задание преимущественно выполнено

СОДЕРЖАНИЕ ОЦЕНОЧНЫХ СРЕДСТВ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

7 семестр

Форма промежуточной аттестации: Экзамен

Пример билета



М Э И

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ №__3__

Кафедра *Прикладной математики и искусственного интеллекта*

Дисциплина *Методы контроля программ*

Утверждаю:

Зав.кафедрой

«28» дек. 20__ *г.*

I. **Теоретические вопросы**

1. *Методика ручного тестирования программного обеспечения.*
2. *Конечные автоматы, проверки с помощью конечных автоматов. Как бороться с большим числом состояний. Расширенный конечный автомат.*

2021 1060-100 000

Процедура проведения

Экзамен устный, по билетам. Студенту дается 60 минут на подготовку, во время ответа задаются дополнительные вопросы на темы, рассмотренные в рамках курса.

I. Перечень компетенций/индикаторов и контрольных вопросов проверки результатов освоения дисциплины

1. Компетенция/Индикатор: ИД-5$_{ПК-1}$ Определяет методы тестирования и умеет проводить все виды контроля программ

Вопросы, задания

1.
Организация ручного тестирования программного обеспечения

1. Методика ручного тестирования программного обеспечения.
2. Критерий успеха. Чек-лист. Сценарий тестирования (тест кейс). Тест скрипт. Набор тестов. Список идей тестов. Модель нагрузки. Дефекты.
3. Средства поддержки процесса ручного тестирования. Продукт HP Sprinter.

2.
Автоматическое тестирование программного обеспечения

1. Недостатки «ручного» тестирования.
2. Автоматическое тестирование программного обеспечения.
3. Что можно автоматизировать.
4. Преимущества автоматического тестирования.
5. Подходы к автоматизации тестирования.
6. Автоматизация тестирования и современные жизненные циклы программного обеспечения.
7. Проблемы автоматизации тестирования.

3.
Модульное тестирование

1. Технология модульного тестирования.
2. Показатели оценки качества покрытия кода тестами.
3. Преимущества технологии модульного тестирования.
4. Недостатки модульного тестирования.
5. Фреймворки модульного тестирования.
6. CppUnit
7. NUnit
8. JUnit

4.
Модульное тестирование баз данных

1. Цели модульного тестирования базы данных.
2. Особенности тестирования базы данных.
3. Схемы организации модульного тестирования базы данных.
4. DBUnit
5. Модульное тестирование баз данных с помощью MS Visual Studio

5.
Автоматизация тестирования приложений через графический интерфейс пользователя

1. Автоматизация тестирования приложений через графический интерфейс пользователя.
2. Четыре поколения средств автоматизации приложений через графический интерфейс пользователя.
3. Утилиты записи и воспроизведения.
4. Плееры тестовых скриптов (сценариев).
5. Тестирование, управляемое данными.
6. Тестирование на основе DSL (domain specific languages).

6.
Coded UI Test

1. Разработка теста, режимы создания тестов.
2. Структура теста.
3. Настройка проверок.
4. Кодирование проверок и сценариев вручную.
5. Настройка и редактирование источников данных.

7.
Автоматизация тестирования приложений через графический интерфейс пользователя – IBM Rational Functional Tester

1. Обзор продукта IBM Rational Functional Tester.
2. Схема работы.
3. Структура скрипта.
4. Технология Script Assure.
5. Точки верификации.
6. Пулы данных.

8.
Тестирование производительности – IBM Rational Performance Tester

1. Обзор продукта IBM Rational Performance Tester.

2. Схема работы.
3. Запись теста.
4. Пулы данных.
5. Проблема корреляции данных.
6. Распределение и назначение рабочей нагрузки.
7. План тестирования.
8. Мониторинг производительности.

9.
Статический анализ кода

1. Обзор методики статического анализа кода.
2. Виды статических анализаторов.
3. Метод метрик. Размерно-ориентированные метрики. Объектно-ориентированные метрики. Комбинированные метрики. Преимущества и недостатки.
4. Статические анализаторы на основе правил. FxCop.
5. Средства статической валидации, интегрированные в редактор кода. JetBrains ReSharper. Visual Assist.

10.
Динамический анализ кода

1. Динамический анализ. Виды динамического анализа. Цели динамического анализа.
2. Измерение производительности — профайлеры.
3. Средства борьбы с утечками памяти и ресурсов. Методы анализа — отладочное API, анализ дампов, автоматическое инструментирование.

11.
Автоматизация тестирования Web-приложений

1. Подходы к автоматизации тестирования Web-приложений.
2. Selenium

Материалы для проверки остаточных знаний

1.Что такое требование в контексте тестирования программного обеспечения?

Ответы:

Описание того, какие функции и с соблюдением каких условий должно выполнять приложение в процессе решения полезной для пользователя задачи

Обращение к разработчику программного обеспечения с настоятельным призывом что-то сделать.

Правила, налагаемые кем-либо.

Устаревший термин, используемый ранее для описания фич программного обеспечения.

Верный ответ: Описание того, какие функции и с соблюдением каких условий должно выполнять приложение в процессе решения полезной для пользователя задачи

2.Что такое отказ?

Ответы:

Отказ – событие, заключающееся в нарушении работоспособного состояния объекта.

Это сбой, из-за которого не работает программа.

Это причина дефекта.

Это дефект.

Верный ответ: Отказ – событие, заключающееся в нарушении работоспособного состояния объекта.

3.Что не входит в показатели надежности программного обеспечения?

Ответы:

Показатели безотказности
Экономические показатели
Показатели долговечности
Показатели ремонтопригодности
Показатели сохраняемости

Верный ответ: Экономические показатели

4.Что не является видом тестирования производительности?

Ответы:

Нагрузочное тестирование.
Стресс-тестирование.
Тестирование стабильности.
Аварийное испытание.

Верный ответ: Аварийное испытание.

5.Существуют ли измеримые показатели для оценки удобства пользованием приложением?

Ответы:

Не существует. Удобство - это очень субъективное понятие.
Существуют.
В некоторых конкретных случаях, когда проектная команда пришла к соглашению, что какой-то параметр может считаться таким показателем, например время изучения основ работы с приложением. Но в общем случае нет.

Верный ответ: Существуют.

6.Укажите список видов тестирования с точки зрения использования в процессе знаний о внутреннем устройстве и логике работы программы?

Ответы:

1	Тестирование «чёрного» ящика, Тестирование «белого» ящика
2	Тестирование «чёрного» ящика, Тестирование «белого» ящика, Тестирование «серого» ящика
3	Тестирование «белого» ящика Тестирование «серого» ящика
4	Все варианты неверные

Верный ответ: 2

7.Чем отличаются статическое и динамическое тестирование?

Ответы:

К тестированию такие категории не применимы.
Статическое тестирование это такой тест программы, когда ее включают и оставляют без воздействий тестировщика, т.е. тест в "статике". А динамическое предполагает активное участие в тесте тестировщика.
При динамическом тестировании программа обязательно должна запуститься, при статическом тестировании нет.
Динамическое тестирование предполагает запуск программы для анализа ее поведения. Статическое тестирование выполняется на основе исходного кода программы без ее запуска.

Верный ответ: Динамическое тестирование предполагает запуск программы для анализа ее поведения. Статическое тестирование выполняется на основе исходного кода программы без ее запуска.

8.Чем является тест-кейс?

Ответы:

Это папка (кейс) с тестами. Сейчас уже не используется. Тесты хранятся в электронном виде.

Упрощенное описание теста, набор определенных шагов, по которым проверяется функциональность системы.

Это скрипт тестирования

Это программа для робота тестировщика

Верный ответ: Упрощенное описание теста, набор определенных шагов, по которым проверяется функциональность системы.

9.Что такое тестирование интерфейса?

Ответы:

Тестирование интерфейса проверяет взаимодействие отдельных модулей.

Это тестирование проводится на веб-сайте для проверки загрузки, функциональности, интерфейса, совместимости и других вопросов, относящихся к качеству работы веб интерфейса.

Чаще всего исследуется эффект разного дизайна, используется метрика для веб-сайтов. Две версии сайта запускаются на одной или нескольких веб-страницах, чтобы определить разницу в кликах и сделать вывод о качестве их интерфейса.

Верный ответ: Тестирование интерфейса проверяет взаимодействие отдельных модулей.

2. Компетенция/Индикатор: ИД-6$_{ПК-1}$ Демонстрирует знания видов сопровождения и умеет применять их на практике

Вопросы, задания

1.

Современное состояние дел в области разработки программного обеспечения

1. Текущая ситуация на рынке разработки программного обеспечения
2. Факторы, тормозящие создание и внедрение новых IT-решений
3. Классические проблемы IT-инфраструктуры предприятий, которые развивали ее по историческим соображениям
4. Тенденции развития методов разработки программного обеспечения

2.

Системы регистрации ошибок

3.

Введение в тестирование программного обеспечения

1. Тестирование программного обеспечения.
2. Виды тестирования.
3. Тестирование и модель жизненного цикла.
4. Как получить тесты.
5. Тестовый сценарий.
6. Построение сценариев.
7. Чек-лист.
8. Тест-кейс.
9. Типовой цикл тестирования.
10. Функциональное тестирование.
11. Тестирование производительности.
12. Тестирование удобства пользования.
13. Тестирование графического интерфейса.

14. Тестирование безопасности.
15. Тестирование локализации
16. Тестирование совместимости
17. Структурное тестирование
18. Восстановительные тесты.
19. Конфигурационное тестирование.
20. Сравнительное тестирование.
21. Аттестационное тестирование.

4.
Надежность программного обеспечения

1. Понятие "надежность программного обеспечения". Отличие от надежности технических систем.
2. Качество и работоспособность. Отличие качества от работоспособности.
3. Отказы и сбои. Типы отказов.
4. Цена ошибки.
5. Методы обеспечения надежности.

Материалы для проверки остаточных знаний

1.Что включает в себя проектная документация?

Ответы:

Материалы маркетингового характера для продвижения продукта на рынке.
Проектная документация включает в себя как документацию по продукту, так и некоторые дополнительные виды документов: пользовательскую и сопроводительную документацию, руководство по установке и использованию, лицензионные соглашения, маркетинговую документацию, проектные артефакты (описание требований, сценариев и вариантов использования, архитектуры и т.д.).
Требования к программному обеспечению и UML диаграммы.
Исходный код программы, включающий документирующие комментарии.

Верный ответ: Проектная документация включает в себя как документацию по продукту, так и некоторые дополнительные виды документов: пользовательскую и сопроводительную документацию, руководство по установке и использованию, лицензионные соглашения, маркетинговую документацию, проектные артефакты (описание требований, сценариев и вариантов использования, архитектуры и т.д.).

2.Чем бизнес-требования отличаются от пользовательских требований.

Ответы:

Пользовательские требования описывают поведение системы, сценарии работы и могут быть использованы для оценки объёма работ, стоимости проекта, времени разработки и т.д. Бизнес-требования как правило дают лишь общее видение задачи без детализации поведения системы и иных технических характеристик, но вполне могут быть использованы для оценки рисков и приоритезации решаемых бизнес-задач.
Пользовательские требования описывают общее видение задачи без детализации поведения системы и иных технических характеристик. При разработке тестов надо ориентироваться на бизнес-требования, дающие более комплексное представление о тестируемом продукте.
Пользовательские требования могут игнорироваться при разработке и тестировании, т.к. поступают от рядовых пользователей. Бизнес-требования подлежат обязательной проверке тестами, так как поступают от представителей бизнеса и не могут игнорироваться.
Пользовательские требования формулируются пользователями, а бизнес-требования формулируются представителями бизнеса и заказчиком.

Верный ответ: Пользовательские требования описывают поведение системы, сценарии работы и могут быть использованы для оценки объёма работ, стоимости

проекта, времени разработки и т.д. Бизнес-требования как правило дают лишь общее видение задачи без детализации поведения системы и иных технических характеристик, но вполне могут быть использованы для оценки рисков и приоритезации решаемых бизнес-задач.

3.Укажите свойства качественных требований.

Ответы:

Двусмысленность
Интерпретируемость
Завершенность
Атомарность
Выполнимость
Актуальность
Стабильность
Ясность
Наукоемкость
Нечеткость (для возможность проще проходить ПСИ)
Однозначность

Верный ответ: Завершенность Атомарность Выполнимость Актуальность Стабильность Ясность Однозначность

4.Можно ли как то оценить прогресс тестирования?

Ответы:

Конечно нет. В программном обеспечении ошибка может быть где угодно и все найти невозможно.
Да, можно при наличии плана тестирования продукта.
Тестирование творческий процесс, оценить его прогресс из-за этого невозможно.
Можно в случае отдельных видов тестирования - например, модульного тестирования.

Верный ответ: Да, можно при наличии плана тестирования продукта.

5.Что такое формальное тестирование?

Ответы:

Верификация программного обеспечения, согласно тест-плану, тестовым процедурам и соответствующей документации, с учетом пожеланий клиента.
Это формальный подход к тестированию программного обеспечения, когда тестировщик не пытается сделать свою работу максимально качественно, а делает работу для галочки (считай для вида, формально протестировано, а реально нет).

Верный ответ: Верификация программного обеспечения, согласно тест-плану, тестовым процедурам и соответствующей документации, с учетом пожеланий клиента.

II. Описание шкалы оценивания

Оценка: 5
Нижний порог выполнения задания в процентах: 70
Описание характеристики выполнения знания: Работа выполнена в рамках "продвинутого" уровня. Ответы даны верно, четко сформулированные особенности практических решений

Оценка: 4
Нижний порог выполнения задания в процентах: 60
Описание характеристики выполнения знания: Работа выполнена в рамках "базового" уровня. Большинство ответов даны верно. В части материала есть незначительные недостатки

Оценка: 3
Нижний порог выполнения задания в процентах: 50

Описание характеристики выполнения знания: Работа выполнена в рамках "порогового" уровня. Основная часть задания выполнена верно. на вопросы углубленного уровня

III. Правила выставления итоговой оценки по курсу

Оценка определяется в соответствии с Положением о балльно-рейтинговой системе для студентов НИУ «МЭИ» на основании семестровой и аттестационной составляющих.