Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Курганский государственный университет»
(КГУ)

Кафедра «Программное обеспечение автоматизированных систем»

УТВЕРЖДАЮ:
Первый проректор
_____ / Т. Р. Змызгова/
«31» августа 2021 г.

Рабочая программа учебной дисциплины

УПРАВЛЕНИЕ КАЧЕСТВОМ И ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

образовательной программы высшего образования –
программы бакалавриата

09.03.04 Программная инженерия

Направленность:
Программное обеспечение автоматизированных систем

Форма обучения: очная, заочная

Курган 2021

Рабочая программа дисциплины «Управление качеством и тестирование программного обеспечения» составлена в соответствии с учебными планами по программе бакалавриата «Программная инженерия» (Программное обеспечение автоматизированных систем), утвержденными для очной формы обучения «30» августа 2021 года, для заочной формы обучения «30» августа 2021 года.

Рабочая программа дисциплины одобрена на заседании кафедры «Программное обеспечение автоматизированных систем» «30» августа 2021 года, протокол № 1.

Рабочую программу составил:

Доцент кафедры
«Программное обеспечение
автоматизированных систем»
к.т.н., доцент _____ А.М. Семахин

Согласовано:

Заведующий кафедрой
«Программное обеспечение
автоматизированных систем»
к.т.н., доцент _____ В. К. Волк

Начальник управления
образовательной деятельности _____ С. Н. Синицин

Специалист по учебно-методической
работе учебно-методического отдела _____ Г.В. Казанкова

1. ОБЪЕМ ДИСЦИПЛИНЫ

Всего: 5 зачетных единицы трудоемкости (180 академических часов)

Очная форма обучения

Вид учебной работы	На всю дисциплину	Семестр
		7
Аудиторные занятия (контактная работа с преподавателем), всего часов в том числе:	**64**	**64**
Лекции	16	16
Лабораторные работы	32	32
Практические занятия	16	16
Аудиторные занятия в интерактивной форме, часов	-	-
Самостоятельная работа, всего часов в том числе:	**116**	**116**
Контрольная работа	18	18
Подготовка к экзамену	27	27
Другие виды самостоятельной работы (самостоятельное изучение тем (разделов) дисциплины)	71	71
Вид промежуточной аттестации	**экзамен**	**экзамен**
Общая трудоемкость дисциплины и трудоемкость по семестрам, часов	**180**	**180**

Заочная форма обучения

Вид учебной работы	На всю дисциплину	Семестр
		6
Аудиторные занятия (контактная работа с преподавателем), всего часов в том числе:	**12**	**12**
Лекции	6	6
Лабораторные работы	6	6
Практические занятия	-	-
Аудиторные занятия в интерактивной форме, часов	-	-
Самостоятельная работа, всего часов в том числе:	**168**	**168**
Контрольная работа	18	18
Подготовка к экзамену	27	27
Другие виды самостоятельной работы (самостоятельное изучение тем (разделов) дисциплины)	123	123
Вид промежуточной аттестации	**экзамен**	**экзамен**
Общая трудоемкость дисциплины и трудоемкость по семестрам, часов	**180**	**180**

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Дисциплина «Управление качеством и тестирование программного обеспечения» относится к части, формируемой участниками образовательных отношений, блока 1 «Дисциплины (модули)».

Изучение дисциплины базируется на знаниях, умениях и навыках, приобретенных студентами, при изучении следующих дисциплин:

- Информатика.
- Основы программирования.
- Алгоритмы и структуры данных.
- Объектно-ориентированное программирование.
- Машинно-ориентированное программирование.

Результаты обучения по дисциплине необходимы для изучения дисциплин: «Технологии проектирования программных систем», «Распределённые вычислительные системы», «Сети ЭВМ и телекоммуникации».

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Цель освоения дисциплины: формирование знаний и практических навыков студентов в выявлении дефектов программного обеспечения.

Задачи дисциплины: формирование понятия качества программного обеспечения, изучение методов тестирования программного обеспечения, формирование навыков тестирования программного обеспечения.

Компетенции, формируемые в результате освоения дисциплины:

- способность осуществлять разработку, отладку, проверку работоспособности, оценку сложности программного обеспечения и рефакторинг программного кода (ПК-7);

- владение концепциями, атрибутами и методами обеспечения качества ПО, способность планировать и проводить тестирование и верификацию выпусков программного продукта (ПК-8).

В результате изучения дисциплины обучающийся должен

знать:

- методы разработки, отладки, проверки работоспособности, оценку сложности программного обеспечения и рефакторинг программного кода (ПК-7);

- методы обеспечения качества ПО, планирования и проведения тестирования и верификации выпусков программного продукта.

уметь:

- осуществлять разработку, отладку, проверку работоспособности, оценку сложности программного обеспечения и рефакторинг программного кода; (ПК-7);

- применять методы разработки, отладки, проверки работоспособности, оценку сложности программного обеспечения и рефакторинг программного кода (ПК-8);

владеть:

- методами разработки, отладки, проверки работоспособности, оценкой сложности программного обеспечения и рефакторингом программного кода (ПК-7);

- концепциями, атрибутами и методами обеспечения качества ПО, способностью планировать и проводить тестирование и верификацию выпусков программного продукта (ПК-8).

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. Учебно-тематический план.

Очная форма обучения

Рубеж	Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем		
			Лекции	Лабораторные работы	Практические работы
Рубеж 1	1	Основные понятия и определения тестирования. Критерии выбора тестов	2	2	2
	2	Структурное тестирование программ	2	4	2
	3	Оценка оттестированности программы. Метод тестирования базового пути	2	2	2
		Рубежный контроль №1	-	2	-
Рубеж 2	4	Функциональное тестирование программ. Эквивалентное разбиение. Анализ граничных значений	2	4	2
	5	Метод функциональных диаграмм. Предположение об ошибке. Системное и регрессионное тестирование	2	4	2
	6	Интеграционное тестирование. Метод нисходящего тестирования	2	4	2
	7	Метод восходящего тестирования	2	4	2
	8	Автоматизированное и ручное тестирование	2	4	2
		Рубежный контроль №2	-	2	-
Всего:			**16**	**32**	**16**

Заочная форма обучения

Рубеж	Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем	
			Лекции	Лабораторные работы
Рубеж 1	1	Основные понятия и определения тестирования. Критерии выбора тестов	0,5	0,5
	2	Структурное тестирование программ	0,5	0,5
	3	Оценка оттестированности программы. Метод тестирования базового пути	1	0,5
Рубеж 2	4	Функциональное тестирование программ. Эквивалентное разбиение. Анализ граничных значений	1	0,5
	5	Метод функциональных диаграмм. Предположение об ошибке. Системное и регрессионное тестирование	1	1
	6	Интеграционное тестирование. Метод нисходящего тестирования	1	1
	7	Метод восходящего тестирования	0,5	1
	8	Автоматизированное и ручное тестирование	0,5	1
Всего:			**6**	**6**

4.2. Содержание лекционных занятий

Тема 1. Основные понятия и определения тестирования. Критерии выбора тестов

Концепция тестирования. Подходы к обоснованию истинности формул, программ и их связь с тестированием. Понятие тестирования и отладки. Организация тестирования. Методы поиска ошибок и процедура тестирования. Фазы тестирования. Проблемы тестирования и задача выбора конечного набора тестов. Классификация видов тестирования.

Требования к идеальному критерию тестирования. Классы критериев. Структурные критерии. Функциональные критерии. Стохастические критерии. Мутационный критерий.

Тема 2. Структурное тестирование программ

Особенности тестирования «белого ящика». Методы тестирования по принципу «белого ящика». Метод покрытия операторов. Метод покрытия решений (покрытия переходов). Метод покрытия условий. Метод покрытия решений/условий. Метод комбинаторного покрытия условий. Достоинства и недостатки тестирования «белого ящика».

Тема 3. Оценка оттестированности программы. Метод тестирования базового пути

Графовые модели проекта. Плоская модель. Иерархическая модель. Сложность тестирования. Степень оттестированности программы. Управляющий граф программы. Особенности управляющего графа программы. Метрики оттестированности.

Назначение метода тестирования базового пути. Цикломатическая сложность. Способы расчёта цикломатической сложности. Независимый путь. Базовое множество. Свойства базового множества. Алгоритм метода базового пути.

Тема 4. Функциональное тестирование программы. Эквивалентное разбиение. Анализ граничных значений

Функциональное назначение, цель и задачи тестирования «чёрного ящика». Эквивалентное разбиение. Этапы метода эквивалентного разбиения. Типы классов эквивалентности. Правила определения классов эквивалентности. Шаги построения тестов.

Анализ граничных значений. Правила метода анализа граничных значений. Отличия метода анализа граничных значений от метода эквивалентного разбиения.

Достоинства и недостатки методов. Пример тестирования программы с применением дерева разбиения области исходных данных.

Тема 5. Метод функциональных диаграмм. Предположение об ошибке. Системное и регрессионное тестирование

Особенности диаграммы причинно-следственных связей. Этапы построения тестов методом функциональных диаграмм. Базовые символы записи функциональных диаграмм. Функции. Логические отношения. Достоинства и недостатки. Пример.

Особенности метода тестирования предположение об ошибке. Этапы метода. Достоинства и недостатки. Пример.

Системное тестирование. Назначение, цели и задачи. Категории тестов системного тестирования. Регрессионное тестирование. Особенности. Достоинства и недостатки.

Тема 6. Интеграционное тестирование. Метод нисходящего тестирования

Функциональное назначение, цели и задачи. Методы сборки модулей: монолитный и инкрементальный. Особенности интеграционного тестирования для процедурного программирования. Особенности интеграционного тестирования для объектно-ориентированного программирования.

Особенности нисходящего тестирования. Правила нисходящего тестирования. Главный управляющий модуль. Подчинённые модули. Этапы нисходящего тестирования. Заглушки. Категории заглушек: A-отображает проходящий параметр, B-отображает трассируемое сообщение, C-возвращает величину из таблицы, D-выполняет табличный поиск по ключу (входному параметру) и воз-

вращает связанный с ним выходной параметр. Достоинства и недостатки нисходящего тестирования. Пример.

Тема 7. Метод восходящего тестирования

Особенности восходящего тестирования. Терминальные модули. Кластер. Драйвер. Типы драйверов: A-вызывает подчинённый модуль, B-посылает элемент данных из внутренней таблицы, C-отображает параметр из подчинённого модуля, C-является комбинацией драйверов B и C. Этапы метода восходящего тестирования. Достоинства и недостатки. Сравнение с методом нисходящего тестирования. Пример.

Тема 8. Автоматизированное и ручное тестирование программ

Автоматический прогон тестов. Структура тестового прогона. Инструментальная система автоматизации тестирования. Эффективность методов тестирования.

Модульное тестирование. Основные принципы модульного тестирования. Требования к системам тестирования. Тестовые фреймворки CppUnit, Boost, t2c. Задачи и особенности. Достоинства и недостатки. Пример.

Тестирование программных приложений с использованием Visual Studio 2019 Community. Создание модульных тестов. Запуск модульных тестов. Просмотр динамических результатов модульных тестов в Visual Studio. Создание модульных тестов с помощью IntelliTest. Платформа тестирования для модульного теста Google Test, Boost Test, CTest. Модульное тестирование приложения универсальной платформы Windows.

Ручное тестирование программ. Инспекции исходного текста. Сквозные просмотры. Проверка за столом. Оценка посредством просмотра. Сравнение ручного и автоматизированного тестирования.

4.3. Лабораторные занятия

Номер раздела, темы	**Наименование раздела, темы**	**Наименование лабораторной работы**	**Норматив времени, час.**	
			Очная форма обучения	Заочная форма обучения
1	Основные понятия и определения тестирования. Критерии выбора тестов	Тестирование компиляторов языка программирования Си на соответствие стандарту ANSI ISO/IEC 9899:1999	2	0,5
2	Структурное тестирование программ	Тестирование программ методами «белого ящика»	4	0,5

3	Оценка оттестированности программы. Метод тестирования базового пути	Разработка управляющего потокового графа	2	0,5
		Рубежный контроль №1	2	-
4	Функциональное тестирование программ. Эквивалентное разбиение. Анализ граничных значений	Тестирование программ методами «чёрного ящика». Дерево решений	4	0,5
5	Метод функциональных диаграмм. Предположение об ошибке. Системное и регрессионное тестирование	Тестирование программ методами «чёрного ящика». Диаграмма причинно-следственных связей	4	1
6	Интеграционное тестирование. Метод нисходящего тестирования	Нисходящее тестирование	4	1
7	Метод восходящего тестирования	Восходящее тестирование	4	1
8	Автоматизированное и ручное тестирование	Автоматизированное тестирование программ	4	1
		Рубежный контроль №2	2	-
		Всего:	**32**	**6**

4.4. Практические занятия

Номер раздела, темы	**Наименование раздела, темы**	**Наименование практической работы**	**Норматив времени, час.**
			Очная форма обучения
1	Основные понятия и определения тестирования. Критерии выбора тестов	Тестирование компиляторов языка программирования Си на соответствие стандарту ANSI ISO/IEC 9899:1999	2
2	Структурное тестирование программ	Тестирование программ методами «белого ящика»	2
3	Оценка оттестированности программы. Метод тестирования базового пути	Разработка управляющего потокового графа	2
4	Функциональное тестирование программ. Эквивалентное разбиение. Анализ граничных значений	Тестирование программ методами «чёрного ящика». Дерево решений	2

5	Метод функциональных диаграмм. Предположение об ошибке. Системное и регрессионное тестирование	Тестирование программ методами «чёрного ящика». Диаграмма причинно-следственных связей	2
6	Интеграционное тестирование. Метод нисходящего тестирования	Нисходящее тестирование	2
7	Метод восходящего тестирования	Восходящее тестирование	2
8	Автоматизированное и ручное тестирование	Автоматизированное тестирование программ	2
		Всего:	**16**

4.5. Контрольная работа

Контрольная работа посвящена разработке визуального приложения, формализующего алгоритм решения задачи, представленной в варианте задания согласно методическим рекомендациям, указанным в разделе 8.

5. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Лекционный курс основывается на методе обучения, использующем технологию, при которой студенты конспектируют теоретический материал, участвуют в опросах и дискуссиях. В этом случае задействованы зрительная, слуховая, моторная и ассоциативная виды памяти.

При прослушивании лекций рекомендуется в конспекте отмечать все важные моменты, на которых заостряет внимание преподаватель, в частности те, которые направлены на качественное выполнение соответствующей лабораторной работы.

Преподавателем запланировано использование при чтении лекций технологии учебной дискуссии. Поэтому рекомендуется фиксировать для себя интересные моменты с целью их активного обсуждения на дискуссии в конце лекции.

Залогом качественного выполнения лабораторных работ является самостоятельная подготовка к ним накануне путем повторения материалов лекций. Рекомендуется подготовить вопросы по неясным моментам и обсудить их с преподавателем в начале занятия.

Лабораторные работы выполняются с применением Microsoft Visual C++ 2022 Community и новых версий этого программного продукта.

Преподавателем запланировано применение на лабораторных занятиях технологий развивающейся кооперации, коллективного взаимодействия, разбора конкретных ситуаций.

Для текущего контроля успеваемости по очной форме обучения преподавателем используется балльно-рейтинговая система контроля и оценки академической активности. Поэтому настоятельно рекомендуется тщательно прора-

батывать материал дисциплины при самостоятельной работе, участвовать во всех формах обсуждения и взаимодействия, как на лекциях, так и на лабораторных и практических занятиях в целях лучшего освоения материала и получения высокой оценки по результатам освоения дисциплины.

Выполнение самостоятельной работы подразумевает самостоятельное изучение разделов дисциплины, подготовку к лабораторным и практическим занятиям, рубежным контролям (для очной формы) и экзамену, выполнению контрольной работы.

Рекомендуемая для очной формы обучения трудоёмкость самостоятельной работы представлена в таблице:

Рекомендуемый режим самостоятельной работы

Наименование вида самостоятельной работы	Рекомендуемая трудоемкость, акад. час.	
	очная форма обучения	заочная форма обучения
Самостоятельное изучение тем дисциплины:	**45**	**120**
Индустриальное тестирование	5	15
Альфа-тестирование	5	15
Бета-тестирование	5	15
Юзабилити-тестирование	6	15
Нагрузочное тестирование	6	15
Тестирование локализации	6	15
Позитивное тестирование	6	15
Негативное тестирование	6	15
Подготовка к лабораторным занятиям (по 1 часу на каждое занятие)	**14**	**3**
Подготовка к практическим занятиям (по 1 часу на каждое занятие)	**8**	**-**
Подготовка к рубежным контролям (по 2 часа на каждый рубеж)	**4**	**-**
Выполнение контрольной работы	**18**	**18**
Подготовка к экзамену	**27**	**27**
Всего:	**116**	**168**

6. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

6.1. Перечень оценочных средств

1. Балльно-рейтинговая система контроля и оценки академической активности студентов в КГУ (для очной формы обучения).
2. Отчеты студентов по лабораторным работам (для очной формы).
3. Отчеты студентов по практическим работам.
4. Банк заданий к рубежным контролям № 1, № 2 (для очной формы обучения).
5. Вопросы к экзамену.
6. Контрольная работа.

6.2. Система балльно-рейтинговой оценки работы студентов по дисциплине

Очная форма обучения

№	Наименование	Содержание						
1	Распределение баллов за семестры по видам учебной работы, сроки сдачи учебной работы *(доводятся до сведения студентов на первом учебном занятии)*	Распределение баллов, 7 семестр						
		Вид учебной работы:	Посещение лекций	Выполнение и защита отчетов по лабораторным работам	Контрольная работа	Рубежный контроль №1	Рубежный контроль №2	Экзамен
		Балльная оценка:	1б*8=8б	6б*6+5б*2=46б	6	5	5	30
2	Критерий пересчета баллов в традиционную оценку по итогам работы в семестре и экзамена	60 и менее баллов – неудовлетворительно; 61…73 – удовлетворительно; 74… 90 – хорошо; 91…100 – отлично						

3	Критерии допуска к промежуточной аттестации, возможности получения автоматически экзаменационной оценки по дисциплине, возможность получения бонусных баллов	Для допуска к промежуточной аттестации (экзамену) студент должен выполнить все лабораторные работы, контрольную работу и набрать не менее 50 баллов. Для получения экзаменационной оценки «автоматически» студенту необходимо набрать следующее минимальное количество баллов: - 68 для получения «автоматически» оценки «удовлетворительно». По согласованию с преподавателем студенту, набравшему минимум 68 баллов, могут быть добавлены дополнительные (бонусные) баллы за активность на консультациях, активное участие в научной и методической работе, оригинальность принятых решений в ходе выполнения лабораторных и практических работ, за участие в значимых учебных и внеучебных мероприятиях кафедры и выставлена за экзамен «автоматически» оценка «хорошо» или «отлично».
4	Формы и виды учебной работы для неуспевающих (восстановившихся на курсе обучения) студентов для получения недостающих баллов в конце семестра	В случае если к промежуточной аттестации (экзамену) набрана сумма менее 50 баллов, студенту необходимо набрать недостающее количество баллов за счет выполнения дополнительных заданий, до конца последней (зачетной) недели семестра. При этом необходимо проработать материал всех пропущенных лабораторных и практических работ и выполнить контрольную работу. Формы дополнительных заданий (назначаются преподавателем): - выполнение и защита пропущенной лабораторной и практической работы (при невозможности дополнительного проведения лабораторной и практической работы, преподаватель устанавливает форму дополнительного задания по тематике пропущенной лабораторной и практической работы самостоятельно) – до 8 баллов. Ликвидация академических задолженностей, возникших из-за разности в учебных планах при переводе или восстановлении, проводится путем выполнения дополнительных заданий, форма и объем которых определяется преподавателем.

6.3. Процедура оценивания результатов освоения дисциплины

Рубежные контроли проводятся в форме письменного тестирования, экзамен в устной форме в виде ответов на вопросы в билетах к экзамену.

Перед проведением рубежного контроля преподаватель прорабатывает со студентами основной материал соответствующих разделов дисциплины в форме краткой лекции-дискуссии.

Варианты заданий для рубежных контролей № 1, № 2 состоят из 20 вопросов. Для определения баллов при проверке рубежных контролей используются интервальные оценки, представленные в таблице

Количество правильных ответов	1-5	6-8	9-11	12-14	15-17	18-20
Количество баллов	0	1	2	3	4	5

На подготовку к рубежному контролю студенту отводится 1 академический час.

Преподаватель оценивает в баллах результаты рубежных контролей студентов по количеству правильных ответов и заносит в ведомость учета текущей успеваемости.

Билет к экзамену состоит из 2 вопросов. Вопросы к экзамену доводятся до студентов на последней лекции в семестре. На подготовку ответа по вопросам билета к экзамену студенту отводится 1 астрономический час. Каждый вопрос оценивается в 15 баллов.

Результаты текущего контроля успеваемости и экзамена заносятся преподавателем в экзаменационную ведомость, которая сдается в организационный отдел института в день экзамена, а также выставляются в зачетную книжку студента.

6.4 Примеры оценочных средств для рубежных контролей, экзамена

6.4.1 Примеры заданий для рубежного контроля №1

Вариант 1_1

1. Что называется тестированием ПО?

1. Процесс исследования ПО с целью получения информации о сложности.

2. Процесс исследования ПО с целью получения информации о надежности.

3. Процесс исследования ПО с целью получения информации о производительности.

4. Процесс исследования ПО с целью получения информации о качестве.

2. Что называется гейзенбагом?

1. Семантическая ошибка.

2. Плавающая ошибка.

3. Синтаксическая ошибка.

4. Логическая ошибка.

3. Какие причины возникновения гейзенбагов?

1. Низкоуровневые ошибки.

2. Ошибки проектирования и реализации программы.

3. Ошибки синтаксиса алгоритмического языка.

4. Ошибка деления на нуль.

4. В чём отличие между аттестацией и верификацией программного продукта?

1. Аттестация подтверждает, что созданный продукт соответствует предъявляемым требованиям необходимый заказчику, верификация, что создан продукт.

2. Отличий нет.

3. Аттестация подтверждает, что создан продукт, необходимый заказчику, верификация, что созданный продукт соответствует предъявляемым требованиям.

4. Отличия незначительные.

5. Тестирование – динамический метод верификации и аттестации ПО?

1. Да.
2. Нет

6. Какие части включает технология тестирования ПО?

1. Внешние воздействия, оценка реакция.
2. Внешние воздействия, реакция испытуемого объекта, оценка реакции, выводы.
3. Внешние воздействия, реакция испытуемого кода.
4. Внешние воздействия, выводы.

7. Какой главный признак соответствует функциональному тестированию?

1. Уровень автоматизации.
2. Степень изолированности.
3. Уровень готовности.
4. Объект тестирования.

8. Какой вид тестирования соответствует стратегии тестирования методами «белого ящика»?

1. По признаку позитивности сценариев.
2. По объекту тестирования.
3. По знанию системы.
4. По степени изолированности.

9. Какое тестирование имеет цель оценить приемлемость пользовательского интерфейса приложения?

1. Тестирование локализации (localization testing).
2. Модульное тестирование (unit testing).
3. Альфа-тестирование (alpha-testing).
4. Тестирование удобства использования (usability testing).

10. Какие уровни тестирования существуют?

1. Модульное.
2. Статистическое.
3. Интеграционное.
4. Системное

11. Какие виды тестирования проводятся при разработке массового («коробочного») программного обеспечения?

1. Альфа-тестирование (alpha-testing).
2. Бета-тестирование (beta-testing).
3. Приёмо-сдаточные испытания (acceptance testing).
4. Нагрузочное тестирование (performance load/stress testing).

12. Что называется покрытие кода тестами?

1. Время выполнения тестирования программного кода.

2. Мера, показывающая на сколько процентов исходный код программы был протестирован.

3. Мера, показывающая отличия тестируемого программного кода от спецификации.

4. Мера, показывающая сложность проведения тестирования программного кода.

13. Какие способы измерения покрытия кода тестами существуют?

1. Покрытие размера программы.

2. Покрытие операторов, покрытие условий.

3. Покрытие путей, покрытие функций.

4. Покрытие вход/выход.

14. Что называется отладкой?

1. Этап разработки программного обеспечения, в процессе которого происходит обнаружение, локализация и устранение ошибок в программе.

2. Этап обнаружения ошибок.

3. Этап устранения ошибок.

4. Этап обнаружения и локализации ошибок.

15. Чем отличается отладка от тестирования?

1. Обнаружение ошибок.

2. Степенью детализации ошибок.

3. Производительностью выполнения

4. Локализация и исправление ошибки.

16. Какие методы тестирования по принципу «белого ящика» существуют?

1. Покрытие операторов.

2. Покрытие объема.

3. Комбинаторное покрытие условий.

4. Комбинаторное покрытие решений

17. Какие методы тестирования по принципу «чёрного ящика» существуют?

1. Логистическое разбиение.

2. Эквивалентное разбиение.

3. Конгруэнтное разбиение.

4. Эвристическое разбиение.

18. Что понимается под граничными условиями в методе анализ граничных значений стратегии тестирования по принципу «черного ящика?»

1. Ситуации, возникающие на входе и выходе программы.

2. Непредсказуемые ситуации, возникающие при тестировании программы.

3. Граничные условия, налагаемые на выполнение операторов.

4. Ситуации, возникающие на высших и низших входных классов эквивалентности.

19. Какая цель метода покрытия решений?

1. Выполнение всех возможных комбинаций результатов условия в каждом решении по крайней мере один раз.

2. Выполнение результатов каждого условия решения по крайней мере один раз.

3. Выполнение каждого оператора программы хотя бы один раз.

4. Каждое направление перехода должно быть реализовано по крайней мере один раз.

20. Какие недостатки метода покрытия решений/условий?

1. Не всегда можно проверить все условия.

2. Невозможно проверить условия, которые скрыты другими условиями.

3. Не всегда можно проверить все решения.

4. Недостаточная чувствительность к ошибкам в логических выражениях.

Вариант 1_2

1. Что называется модульным тестированием?

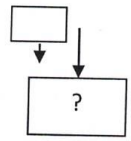
1. Пошаговое (по модульное) наращиванием комплекса программ с пошаговым тестированием собираемого комплекса.

2. Тестирование, характеризующееся одновременным объединением всех модулей в тестируемый комплекс.

3. Тестирование программы на уровне отдельно взятых модулей, функций, классов.

4. Сборка и тестирование системы начинаются с модулей нижнего уровня иерархии. Модули подключаются движением снизу вверх. Отсутствие заглушек.

2. Какая категория заглушки изображена на рисунке?



1. Заглушка A – отображает проходящий параметр.

2. Заглушка B – отображает трассируемое сообщение.

3. Заглушка C – возвращает величину из таблицы.

4. Заглушка D – выполняет табличный поиск по ключу и возвращает связанный с ним выходной параметр.

3. Что называется тестовым драйвером (test driver)?

1. Драйвер, управляющий тестированием кластеров.

2. Драйвер, управляющий заглушками.

3. Драйвер, классами эквивалентности.

4. Драйвер, управляющий диаграммами причинно-следственных связей.

4. Что называется инкрементальным тестированием?

1. Объединение модулей движением сверху вниз по управляющей иерархии, начиная с главного управляющего модуля. Подчиненные модули добавляются в структуру в результате поисков глубину или в ширину.

2. Тестирование программы на уровне отдельно взятых модулей, функций, классов.

3. Пошаговое (помодульное) наращиванием комплекса программ с пошаговым тестированием собираемого комплекса.

4. Тестирование, характеризующееся одновременным объединением всех модулей в тестируемый комплекс.

5. Какой недостаток метода восходящее тестирование интеграции?

1. Количество независимых маршрутов может быть велико.

2. Нельзя обнаружить ошибки, появление которых зависит от обрабатываемых данных.

3. Необходимость заглушек и связанные с ними трудности тестирования.

4. Система не существует как объект, пока не будет добавлен последний модуль.

6. Какой достоинство метода восходящее тестирование интеграции?

1. Отсутствие заглушек и упрощение разработки тестовых вариантов.

2. Гарантируется проверка всех независимых маршрутов программы.

3. Возможность раннего тестирования главных управляющих функций.

4. Анализируется правильность внутренних структур данных.

7. Что называется восходящим тестированием интеграции?

1. Тестирование, характеризующееся одновременным объединением всех модулей в тестируемый комплекс.

2. Тестирование программы на уровне отдельно взятых модулей, функций, классов.

3. Сборка и тестирование системы начинаются с модулей нижнего уровня иерархии. Модули подключаются движением снизу вверх. Отсутствие заглушек.

4. Пошаговое (помодульное) наращиванием комплекса программ с пошаговым тестированием собираемого комплекса.

8. Какие методы интеграционного тестирования разработаны?

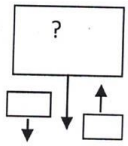
1. Эквивалентное разбиение.

2. Монолитное тестирование.

3. Покрытие операторов.

4. Предположение об ошибке.

9. Какой тип драйвера изображен на рисунке?



1. Драйвер A – вызывает подчиненный модуль.

2. Драйвер B – посылает элемент данных (параметр) из внутренней таблицы.

3. Драйвер C – Отображает параметр из подчиненного модуля

4. Драйвер D – является комбинацией драйверов B и C.

10. Что называется классом эквивалентности?

1. Набор разнотипных данных.

2. Набор взаимосвязанных данных.

3. Набор данных с общими свойствами.

4. Набор взаимоисключаемых данных.

11. Какой алгоритм нисходящей интеграции?

1. 1). Главный управляющий модуль – тестовый драйвер. Подчиненные модули – заглушки.

2) Заглушка заменяется реальным модулем.

3) Тестирование структуры.

4) Если в модуле драйвере нет заглушек, то производится смена модуля.

5) Переход на шаг 2).

2. 1) Модули уровня объединяются в кластеры.

2) Разрабатывается драйвер для координации ввода/вывода.

3) Тестируется кластер.

4) Драйверы удаляются а кластеры объединяются в структуру движением вверх.

3. 1). Главный управляющий модуль – тестовый драйвер. Подчиненные модули – заглушки.

2) Заглушка заменяется реальным модулем.

3) Тестирование структуры.

4. 1). Главный управляющий модуль – тестовый драйвер. Подчиненные модули – заглушки.

2) Заглушка заменяется реальным модулем.

3) Тестирование структуры.

4) Если в модуле драйвере нет заглушек, то производится смена модуля.

5) Переход на шаг 3).

12. Какой недостаток нисходящего тестирования интеграции?

1. Необходимость заглушек и связанные с ними трудности тестирования.

2. Количество независимых маршрутов может быть велико.

3. Нельзя обнаружить ошибки, появление которых зависит от обрабатываемых данных.

4. Система не существует как объект, пока не будет добавлен последний модуль.

13. Какой алгоритм восходящего тестирования интеграции?

1. 1) Модули уровня объединяются в кластеры.

2) Разрабатывается драйвер для координации ввода/вывода.

3) Тестируется кластер.

4) Драйверы удаляются а кластеры объединяются в структуру движением вверх.

2. 1). Главный управляющий модуль – тестовый драйвер. Подчиненные модули – заглушки.

2) Заглушка заменяется реальным модулем.

3) Тестирование структуры.

4) Если в модуле драйвере нет заглушек, то производится смена модуля.

5) Переход на шаг 2).

3. . 1). Главный управляющий модуль – тестовый драйвер. Подчиненные модули – заглушки.

2) Заглушка заменяется реальным модулем.

3) Тестирование структуры.

4. 1). Главный управляющий модуль – тестовый драйвер. Подчиненные модули – заглушки.

2) Заглушка заменяется реальным модулем.

3) Тестирование структуры.

4) Если в модуле драйвере нет заглушек, то производится смена модуля.

5) Переход на шаг 3).

14. Что называется нисходящим тестированием интеграции?

1. Объединение модулей движением снизу вверх по управляющей иерархии, начиная с главного управляющего модуля. Подчиненные модули добавляются в структуру в результате поисков глубину или в ширину.

2. Тестирование, рассматривающее систему в целом и оперирующее на уровне пользовательских интерфейсов.

3. Тестирование, характеризующееся одновременным объединением всех модулей в тестируемый комплекс.

4. Объединение модулей движением сверху вниз по управляющей иерархии, начиная с главного управляющего модуля. Подчиненные модули добавляются в структуру в результате поисков глубину или в ширину.

15. Что называется заглушкой (test stub)?

1. Фиктивная подпрограмма, не имеющая точку входа и выхода.

2. Фиктивная подпрограмма, имитирующая одну или несколько функций отсутствующего модуля программного продукта. Имеет точку входа и точку выхода.

3. Фиктивная подпрограмма, имеющая точку входа, но не имеющая точку выхода.

4. Фиктивная подпрограмма ограниченного размера.

16. Что называется монолитным тестированием?

1. Тестирование, рассматривающее систему в целом и оперирующее на уровне пользовательских интерфейсов.

2. Объединение модулей движением сверху вниз по управляющей иерархии, начиная с главного управляющего модуля. Подчиненные модули добавляются в структуру в результате поисков глубину или в ширину..

3. Тестирование, характеризующееся одновременным объединением всех модулей в тестируемый комплекс.

4. Объединение модулей движением снизу вверх по управляющей иерархии, начиная с главного управляющего модуля. Подчиненные модули добавляются в структуру в результате поисков глубину или в ширину.

17. Что такое диаграмма причинно-следственных связей?

1. Последовательность выполняемых действий.

2. Способ проектирования тестовых вариантов, использующий формальную запись логических условий и действий.

3. Взаимосвязь между элементами программы.

4. Координация действий в процессе тестирования.

18. Какое достоинство нисходящего тестирования интеграции?

1. Возможность раннего тестирования главных управляющих функций.

2. Анализируется правильность внутренних структур данных.

3. Гарантируется проверка всех независимых маршрутов программы.

4. Отсутствие заглушек и упрощение разработки тестовых вариантов.

19. Какая цель интеграционного тестирования?

1. Проверка функциональных требований к программе.

2. Выявление локализованных в модуле ошибок в реализации алгоритмов и определение степени готовности системы к переходу на следующий уровень разработки.

3. Выявление дефектов, связанных с работой системы в целом

4. Поиск дефектов, связанных с ошибками в реализации и интерпретации интерфейсного взаимодействия между модулями.

20. Какая цель модульного тестирования?

1. Выявление локализованных в модуле ошибок в реализации алгоритмов и определение степени готовности системы к переходу на следующий уровень разработки.

2. Выявление дефектов, связанных с работой системы в целом.

3. Проверка функциональных требований к программе.

4. Поиск дефектов, связанных с ошибками в реализации и интерпретации интерфейсного взаимодействия между модулями.

6.4.2 Примеры заданий для рубежного контроля №2

Вариант 2_1

1. Что называется системным тестированием?

1. Тестирование логики программы.

2. Тестирование функций программы.

3. Тестирование системы в целом, оперирующим на уровне пользовательских интерфейсов

4. Тестирование программ при сборке проекта

2. Какая основная задача системного тестирования?

1. Выявление дефектов, связанных с работой системы в целом.

2. Выявление дефектов, связанных с логикой программы.

3. Выявление дефектов, связанных с функций программы.

4. Выявление дефектов, связанных с изменениями в программе.

3. Что относится к категории тестов системного тестирования?

1. Корректность операторов.

2. Корректность переходов.

3. Корректность условий.

4. Корректность документации.

4. Что называется регрессионным тестированием?

1. Тестирование логики программы.

2. Тестирование, проводимое при внесении изменений на фазе системного тестирования или сопровождения продукта.

3. Тестирование функций программы.

4. Тестирование системы в целом, оперирующим на уровне пользовательских интерфейсов.

5. Что называется автоматизированным тестированием?

1. Тестирование, использующее программные средства для выполнения тестов и проверки результатов выполнения, приводящее к сокращению времени тестирования и упрощению процесса.

2. Тестирование системы в целом, оперирующим на уровне пользовательских интерфейсов.

3. Тестирование логики программы.

4. Тестирование, проводимое при внесении изменений на фазе системного тестирования или сопровождения продукта.

6. Сколько подходов к автоматизации тестирования существует?

1. 5
2. 3
3. 4
4. 2

7. Какие подходы к автоматизации тестирования существуют?

1. Минимизация объема программы.
2. Тестирование на уровне кода.
3. Тестирование пользовательского интерфейса.
4. Максимизация производительности программы.

8. Какая главная проблема автоматизированного тестирования?

1. Трудоемкость.
2. Энергозатратность.
3. Малая производительность.
4. Большой объем.

9. Что называется регрессионной ошибкой?

1. Ошибка, приводящая к сбою работ программы после подписания технического задания.

2. Ошибка, приводящая к сбою работ программы после интеграционного тестирования.

3. Ошибка, приводящая к сбою работ программы после структурного тестирования.

4. Ошибка, приводящая к сбою работ программы после внесения изменений.

10. Какая цель регрессионного тестирования?

1. Проверка правильности работы программы после системного тестирования.

2. Проверка правильности работы программы после внесения изменений.

3. Проверка правильности работы программы после функционального тестирования.

4. Проверка правильности работы программы после структурного тестирования.

11. Какие типы сопровождения в регрессионном тестировании существуют?

1. Адаптивное.
2. Гибкое.
3. Универсальное.

4. Настраиваемое.

12. Какие типы регрессионного тестирования существуют?

1. Прогрессивное.
2. Регрессивное.
3. Регрессионное.
4. Оптимальное.

13. Какие методы регрессионного тестирования существуют?

1. Максимизации.
2. Минимизации.
3. Оптимизации.
4. Рационализации.

14. Что называется случайным методом?

1. Выбор тестов определенным образом.
2. Выбор тестов предопределенным образом.
3. Выбор тестов оптимальным образом.
4. Выбор тестов случайным образом.

15. Какое обозначение случайных методов?

1. randomize(x)
2. srandom(x)
3. random(x).
4. rand(x)

16. Что называется безопасным методом?

1. Метод, обеспечивающий выбор всех тестов, по определенным правилам.

2. Метод, обеспечивающий выбор всех тестов, обнаруживающих изменения.

3. Метод, обеспечивающий выбор всех тестов, случайным образом.
4. Метод, обеспечивающий выбор всех тестов, оптимальным образом.

17. В чем значимость методов, основанных на покрытии кода?

1. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия С, использовавшегося при создании последнего набора тестов.

2. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия С, использовавшегося при создании первоначального набора тестов.

3. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия С, использовавшегося при создании минимального набора тестов.

4. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия С, использовавшегося при создании максимального набора тестов.

18. Что называется регрессией багов (bug regression)?

1. Попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

2. Попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок.

3. Попытка доказать, что исправленная ошибка на самом деле исправлена.

4. Попытка доказать, что исправленная ошибка на самом деле не исправлена.

19. Что называется регрессией побочного эффекта (side effect regression)?

1. Попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок.

2. Попытка доказать, что исправленная ошибка на самом деле не исправлена.

3. Попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

4. Попытка доказать, что исправленная ошибка на самом деле исправлена.

20. Что называется регрессией старых багов (old bugs regression)?

1. Попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок.

2. Попытка доказать, что исправленная ошибка на самом деле не исправлена.

3. Попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

4. Попытка доказать, что исправленная ошибка на самом деле исправлена.

Вариант 2_2

1. Что называется регрессионным тестированием?

1. Тестирование логики программы.

2. Тестирование, проводимое при внесении изменений на фазе системного тестирования или сопровождения продукта.

3. Тестирование функций программы.

4. Тестирование системы в целом, оперирующим на уровне пользовательских интерфейсов.

2. Что называется автоматизированным тестированием?

1. Тестирование, использующее программные средства для выполнения тестов и проверки результатов выполнения, приводящее к сокращению времени тестирования и упрощению процесса.

2. Тестирование системы в целом, оперирующим на уровне пользовательских интерфейсов.

3. Тестирование логики программы.

4. Тестирование, проводимое при внесении изменений на фазе системного тестирования или сопровождения продукта.

3. Какая цель интеграционного тестирования?

1. Проверка функциональных требований к программе.

2. Выявление локализованных в модуле ошибок в реализации алгоритмов и определение степени готовности системы к переходу на следующий уровень разработки.

3. Выявление дефектов, связанных с работой системы в целом

4. Поиск дефектов, связанных с ошибками в реализации и интерпретации интерфейсного взаимодействия между модулями.

4. Какая цель модульного тестирования?

1. Выявление локализованных в модуле ошибок в реализации алгоритмов и определение степени готовности системы к переходу на следующий уровень разработки.

2. Выявление дефектов, связанных с работой системы в целом.

3. Проверка функциональных требований к программе.

4. Поиск дефектов, связанных с ошибками в реализации и интерпретации интерфейсного взаимодействия между модулями.

5. Что называется безопасным методом?

1. Метод, обеспечивающий выбор всех тестов, по определенным правилам.

2. Метод, обеспечивающий выбор всех тестов, обнаруживающих изменения.

3. Метод, обеспечивающий выбор всех тестов, случайным образом.

4. Метод, обеспечивающий выбор всех тестов, оптимальным образом.

6. В чем значимость методов, основанных на покрытии кода?

1. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия C, использовавшегося при создании последнего набора тестов.

2. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия C, использовавшегося при создании первоначального набора тестов.

3. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия C, использовавшегося при создании минимального набора тестов.

4. Сохранение выбранным набором тестов требуемой степени покрытия элементов P'' относительно критерия структурного покрытия C, использовавшегося при создании максимального набора тестов.

7. Что называется системным тестированием?

1. Тестирование логики программы.

2. Тестирование функций программы.

3. Тестирование системы в целом, оперирующим на уровне пользовательских интерфейсов

4. Тестирование программ при сборке проекта

8. Какая основная задача системного тестирования?

1. Выявление дефектов, связанных с работой системы в целом.

2. Выявление дефектов, связанных с логикой программы.

3. Выявление дефектов, связанных с функций программы.

4. Выявление дефектов, связанных с изменениями в программе.

9. Что относится к категории тестов системного тестирования?

1. Корректность операторов.

2. Корректность переходов.

3. Корректность условий.

4. Корректность документации.

10. Что называется инкрементальным тестированием?

1. Объединение модулей движением сверху вниз по управляющей иерархии, начиная с главного управляющего модуля. Подчиненные модули добавляются в структуру в результате поисков глубину или в ширину.

2. Тестирование программы на уровне отдельно взятых модулей, функций, классов.

3. Пошаговое (помодульное) наращиванием комплекса программ с пошаговым тестированием собираемого комплекса.

4. Тестирование, характеризующееся одновременным объединением всех модулей в тестируемый комплекс.

11. Какой недостаток метода восходящее тестирование интеграции?

1. Количество независимых маршрутов может быть велико.

2. Нельзя обнаружить ошибки, появление которых зависит от обрабатываемых данных.

3. Необходимость заглушек и связанные с ними трудности тестирования.

4. Система не существует как объект, пока не будет добавлен последний модуль.

12 Какие методы структурного тестирования программного обеспечения?

1. Методы структурного тестирования:

- покрытие операторов;
- покрытие объёма;

- покрытие условий;
- покрытие решений/условий;
- комбинаторное покрытие условий.

2. Методы структурного тестирования:

- покрытие операторов;
- покрытие количества;
- покрытие условий;
- покрытие решений/условий;
- комбинаторное покрытие условий.

3. Методы структурного тестирования:

- покрытие операндов;
- покрытие решений;
- покрытие условий;
- покрытие решений/условий;
- комбинаторное покрытие условий.

4. Методы структурного тестирования:

- покрытие операторов;
- покрытие решений;
- покрытие условий;
- покрытие решений/условий;
- комбинаторное покрытие условий.

13 Какие функциональные критерии?

1. Виды функциональных критериев:

- тестирование проектов спецификации;
- тестирование классов входных данных;
- тестирование исключений;
- тестирование классов выходных данных;
- тестирование функций;
- комбинированные критерии для программ и спецификаций.

2. Виды функциональных критериев:

- тестирование проектов спецификации;
- тестирование классов входных данных;
- тестирование правил;
- тестирование классов выходных данных;
- тестирование функций;
- комбинированные критерии для программ и спецификаций.

3. Виды функциональных критериев:

- тестирование проектов спецификации;
- тестирование классов входных данных;
- тестирование правил;
- тестирование классов выходных данных;
- тестирование функций;
- комбинированные условия для программ и спецификаций.

4. Виды функциональных критериев:

- тестирование проектов спецификации;
- тестирование классов входных данных;
- тестирование условий;
- тестирование классов выходных данных;
- тестирование функций;
- комбинированные критерии для программ и спецификаций.

14. Что называется регрессией багов (bug regression)?

1. Попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

2. Попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок.

3. Попытка доказать, что исправленная ошибка на самом деле исправлена.

4. Попытка доказать, что исправленная ошибка на самом деле не исправлена.

15. Что называется регрессией побочного эффекта (side effect regression)?

1. Попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок.

2. Попытка доказать, что исправленная ошибка на самом деле не исправлена.

3. Попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

4. Попытка доказать, что исправленная ошибка на самом деле исправлена.

16. Что называется регрессией старых багов (old bugs regression)?

1. Попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок.

2. Попытка доказать, что исправленная ошибка на самом деле не исправлена.

3. Попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

4. Попытка доказать, что исправленная ошибка на самом деле исправлена.

17. Какая цель регрессионного тестирования?

1. Проверка правильности работы программы после системного тестирования.

2. Проверка правильности работы программы после внесения изменений.

3. Проверка правильности работы программы после функционального тестирования.

4. Проверка правильности работы программы после структурного тестирования.

18. Какие типы сопровождения в регрессионном тестировании существуют?

1. Адаптивное.
2. Гибкое.
3. Универсальное.
4. Настраиваемое.

19. Какие типы регрессионного тестирования существуют?

1. Прогрессивное.
2. Регрессивное.
3. Регрессионное.
4. Оптимальное.

20. Какие методы регрессионного тестирования существуют?

1. Максимизации.
2. Минимизации.
3. Оптимизации.
4. Рационализации.

6.4.3 Таблица ответов

№ вопроса	Правильные ответы			
	Рубежный контроль №1		Рубежный контроль №2	
	Вариант 1_1	Вариант 1_2	Вариант 2_1	Вариант 2_2
1	4	3	3	2
2	2	2	1	1
3	1	1	4	4
4	3	3	2	1
5	1	4	1	2

6	2	1	4	2
7	4	3	2, 3	3
8	3	2	1	1
9	4	4	4	4
10	1,3,4	3	2	3
11	1,2,3	1	1	4
12	2	1	1	4
13	2,3,4	1	2	2
14	1	4	4	4
15	4	2	3	3
16	1,3	3	2	1
17	2	2	2	2
18	4	1	4	1
19	4	4	3	1
20	1,2,4	1	1	2

6.4.4 Примерный перечень вопросов для экзамена

1. Цель, задачи тестирования. Основные понятия и определения. Этапы тестирования программ.
2. Классификация методов тестирования программ.
3. Основные принципы тестирования программ.
4. Соотношение качества программы и времени разработки.
5. Процессы тестирования и их связь с процессами проектирования.
6. Концепция тестирования программ. Методы обоснования истинности формул.
7. Идеальный критерий тестирования. Требования к идеальному критерию тестирования. Классы критериев тестирования.
8. Структурные критерии тестирования.
9. Функциональные критерии тестирования.
10. Стохастические критерии тестирования.
11. Мутационный критерий тестирования.
12. Оценка тестированности программы. Плоская модель управляющего графа программы.
13. Оценка тестированности программы. Иерархическая модель управляющего графа программы.
14. Интегральная оценка тестированности программ. Методика интегральной оценки тестированности.
15. Методы ручное тестирования. Инспекции исходного текста.
16. Методы ручного тестирования. Сквозные просмотры.
17. Методы ручного тестирования. Проверка за столом.
18. Методы ручного тестирования. Оценка посредством просмотра.
19. Тестирование путем покрытия логики программы. Покрытие операторов. Пример.

20. Структурное тестирование. Покрытие решений. Пример.

21. Структурное тестирование. Покрытие решений/условий. Пример.

22. Структурное тестирование. Комбинаторное покрытие условий. Пример.

23. Функциональное тестирование. Эквивалентное разбиение. Предположение об ошибке.

24. Функциональное тестирование. Анализ граничных значений.

25. Функциональное тестирование. Метод диаграмм причинно-следственных связей.

26. Интеграционное тестирование программы. Метод нисходящего тестирования.

27. Интеграционное тестирование программы. Метод восходящего тестирования.

28. Интеграционное тестирование. Монолитная сборка модулей. Сравнение монолитного и интегрального подходов.

29. Особенности интеграционного тестирования для процедурного программирования.

30. Особенности интеграционного тестирования для объектно-ориентированного программирования.

31. Модульное тестирование программы. Принципы тестирования.

32. Системное тестирование программ. Категории тестов системного тестирования программ.

33. Автоматизация тестирования программ. Структура инструментальной системы автоматизации тестирования.

34. Тестовые фреймворки CppUnit, Boost, t2c. Задачи и особенности. Достоинства и недостатки. Пример.

35. Интегрированное тестирование программных продуктов с использованием Visual Studio 2019 Community. Платформs тестирования для модульного теста Google Test, Boost Test, CTest.

36. Тестирование готового программного продукта. Тестирование документации

6.4.5 Примерный перечень тем контрольной работы

6.4.5.1 Назначение, цели и задачи контрольной работы

Разрабатываемое приложение контрольной работы предназначено для формализации решения задачи тестирования программы методом базового пути.

Цель выполнения контрольной работы – закрепление теоретических знаний и повышение практических навыков в разработке программы тестирования методом базового пути на языке C++.

Задачи, решаемые студентом в процессе выполнения контрольной работы:

- изучение теоретического обоснования предметной области;

- разработка алгоритма решения задачи;
- разработка программного приложения, формализующего решения задачи;
- документирование контрольной работы.

6.4.5.2 Требования к контрольной работе

Контрольная работа включает визуальное приложение на языке C++ и пояснительную записку, содержащую разделы:
- введение;
- постановка задачи;
- алгоритм решения задачи;
- результаты работы приложения (скриншоты);
- заключение;
- список использованных источников;
- содержание;

К пояснительной записке прилагается комплект документации:
- спецификация (ГОСТ 19.202-78);
- описание программы (ГОСТ 19.402-78);
- руководство пользователя (ГОСТ 19.505-79);
- руководство программиста (ГОСТ 19.504-79).

6.4.5.3 Варианты контрольной работы

Разработать визуальное приложение на языке C++, формализующее алгоритм метода структурного тестирования базового пути.

Вариант 1

$$y=\begin{cases} a*(x+c)^2-b, при\ x=0, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x}-\mathrm{a}}{-\mathrm{c}}, при\ x=0\ и\ \mathrm{b}=0 \\ a+\dfrac{x}{c}, в\ остальных\ случаях \end{cases}$$

Вариант 2

$$y=\begin{cases} a*x^2-cx+b, при\ x+10<0, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x}-\mathrm{a}}{\mathrm{x}-\mathrm{c}}, при\ x+10>0\ и\ b=0 \\ \dfrac{-x}{a-c}, в\ остальных\ случаях \end{cases}$$

Вариант 3

$$y=\begin{cases} a*x^3+b*x^2, при\ x<0, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x-a}}{\mathrm{x-c}}, при\ x>0\ и\ b=0 \\ \dfrac{x+5}{c(x-10)}, в\ остальных\ случаях \end{cases}$$

Вариант 4

$$y=\begin{cases} a*(x+7)^2-b, при\ x<5, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x-c*d}}{\mathrm{a*x}}, при\ x>5\ и\ b=0 \\ \dfrac{x}{c}, в\ остальных\ случаях \end{cases}$$

Вариант 5

$$y=\begin{cases} -\dfrac{2*x-c}{c*x-a}, при\ x<0, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x-a}}{\mathrm{x-c}}, при\ x>0\ и\ b=0 \\ -\dfrac{x}{c}+\dfrac{-b}{2*c}, в\ остальных\ случаях \end{cases}$$

Вариант 6

$$y=\begin{cases} a*x^3+b^2+c, при\ x<0, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x-a}}{\mathrm{c}}, при\ x>0\ и\ c\neq 0 \\ \dfrac{15*x}{c}, в\ остальных\ случаях \end{cases}$$

Вариант 7

$$y=\begin{cases} a*x^2+2*b*x+5, при\ x<5, \mathrm{b}\neq 0 \\ \dfrac{3*\mathrm{x-a}}{\mathrm{c}}, при\ x\geq 5\ и\ c\neq 0 \\ \dfrac{10*x}{c+6}, в\ остальных\ случаях \end{cases}$$

Вариант 8

$$y=\begin{cases}(a+2)*x^2+5*b, при\ x<10, \mathrm{b}\neq 0\\ \dfrac{\mathrm{x}+\mathrm{a}}{7*\mathrm{c}}, при\ x\geq 10\ и\ c\neq 0\\ \dfrac{x}{c+14}, в\ остальных\ случаях\end{cases}$$

Вариант 9

$$y=\begin{cases}a*x^4+b*x^2, при\ x<0, \mathrm{b}\neq 0\\ \dfrac{\mathrm{x}+\mathrm{a}}{\mathrm{c}}, при\ x\geq 0\ и\ c\neq 0\\ \dfrac{15*x}{c+9}, в\ остальных\ случаях\end{cases}$$

Вариант 10

$$y=\begin{cases}a*x^2+25*b*x, при\ x<0, \mathrm{b}\neq 0\\ \dfrac{\mathrm{x}-\mathrm{a}}{\mathrm{c}}, при\ x\geq 0\ и\ c\neq 0\\ \dfrac{16*x}{c+8}, в\ остальных\ случаях\end{cases}$$

6.5. Фонд оценочных средств

Полный банк заданий для текущего, рубежных контролей и промежуточной аттестации по дисциплине, показатели, критерии, шкалы оценивания компетенций, методические материалы, определяющие процедуры оценивания образовательных результатов, приведены в учебно-методическом комплексе дисциплины.

7. ОСНОВНАЯ И ДОПОЛНИТЕЛЬНАЯ УЧЕБНАЯ ЛИТЕРАТУРА

7.1. Основная учебная литература

1. Джеф Решка, Джон Пол, Элфрид Дастин. Тестирование программного обеспечения. Внедрение, управление и автоматизация. – М.: Лори, 2012. – 600 с.

2. Кори Сандлер, Том Баджет, Гленфорд Майерс. Искусство тестирования программ. – М.: Вильямс, 2012. – 272 с.

3. Левинсон Дж. Тестирование ПО с помощью Visual Studio 2010. – М.: ЭКОМ Паблишерз, 2012. – 336 с.

7.2. Дополнительная учебная литература

1. Канер Сэм, Фолк Джек. Тестирование программного обеспечения. – М.: ДиаСофт, 2001. – 538 с.

2. Тампе Луиза. Введение в тестирование программного обеспечения. – М.: Вильямс, 2003. –320 с.

3. Элфрид Дастин, Джеф Решка, Джон Пол. Автоматизированное тестирование программного обеспечения. – М.: Лори, 2003. – 592 с.

4. Степанченко И.В. Методы тестирования программного обеспечения: Учебное пособие. - Волгоград: ВолгГТУ, 2006. - 74 с.

http://window.edu.ru/resource/765/45765(дата обращения: 06.03.2017).

8. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

1. Семахин А.М. Управление качеством и тестирование программного обеспечения. Методические указания к выполнению лабораторных работ для студентов направления подготовки 09.03.04 «Программная инженерия». Курган, КГУ, 2017. – 43 с..

2. Семахин А.М. Управление качеством и тестирование ПО. Методические указания к выполнению практических и контрольных работ для студентов направления подготовки 09.03.04 «Программная инженерия». Курган, КГУ, 2017. – 36 с. (электронный).

3. Семахин А.М. Методы верификации и оценки качества программного обеспечения : учебное пособие. – Курган : Изд-во КГУ, 2018. – 150 с.

9. РЕСУРСЫ СЕТИ «ИНТЕРНЕТ», НЕОБХОДИМЫЕ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

1. Федеральный портал «Российское образование» URL: http://www.edu.ru/

2. Сайт дистанционного обучения в НОУ «ИНТУИТ». URL: http://www.intuit.ru/

10. ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ И ИНФОРМАЦИОННЫЕ СПРАВОЧНЫЕ СИСТЕМЫ

При чтении лекций используются слайдовые презентации.

Минимальные требования к операционной системе и программному обеспечению компьютера, используемого при показе слайдовых презентаций:

Windows XP, Foxit PDF Reader 12.1, 13.12.2022 г., свободное программное обеспечение (free software), GNU License.

11. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Материально-техническое обеспечение включает в себя учебные лаборатории и классы, оснащенные современными компьютерами (рабочими станциями локальной вычислительной сети) с доступом в Интернет, мультимедийное оборудование (переносной персональный компьютер, мультимедийный проектор, мультимедийный экран).

Программные средства обеспечения учебного процесса включают лицензионное программное обеспечение: операционную систему Windows 10, интегрированную среду программирования Microsoft Visual Studio 2022 Community, язык программирования Microsoft Visual C++.

12. ДЛЯ ОБУЧАЮЩИХСЯ С ИСПОЛЬЗОВАНИЕМ ДИСТАНЦИОННЫХ ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ

При использовании электронного обучения и дистанционных образовательных технологий (далее ЭО и ДОТ) занятия полностью или частично проводятся в режиме онлайн. Объем дисциплины и распределение нагрузки по видам работ соответствует п. 4.1. Распределение баллов соответствует п. 6.2 либо может быть изменено в соответствии с решением кафедры, в случае перехода на ЭО и ДОТ в процессе обучения. Решение кафедры об используемых технологиях и системе оценивания достижений обучающихся принимается с учетом мнения ведущего преподавателя и доводится до обучающихся.

ПРИЛОЖЕНИЕ 1

Аннотация к рабочей программе дисциплины

«Управление качеством и тестирование программного обеспечения»

образовательной программы высшего образования –
программы бакалавриата
09.03.04 – Программная инженерия
Направленность:
Программное обеспечение автоматизированных систем

Трудоемкость дисциплины: 5 ЗЕ (180 академических часа)
Семестр: 7 (очная, 6 заочная)
Форма промежуточной аттестации: экзамен

Содержание дисциплины

Концепция тестирования. Подходы к обоснованию истинности формул, программ и их связь с тестированием. Понятие тестирования и отладки. Классификация видов тестирования. Требования к идеальному критерию тестирования. Классы критериев.

Особенности тестирования «белого ящика». Методы тестирования по принципу «белого ящика». Достоинства и недостатки тестирования «белого ящика». Графовые модели проекта. Плоская модель. Иерархическая модель. Назначение метода тестирования базового пути. Цикломатическая сложность.

Функциональное назначение, цель и задачи тестирования «чёрного ящика». Эквивалентное разбиение. Анализ граничных значений. Особенности диаграммы причинно-следственных связей и предположения об ошибке.

Системное тестирование. Назначение, цели и задачи. Категории тестов системного тестирования. Регрессионное тестирование. Особенности. Достоинства и недостатки.

Функциональное назначение, цели и задачи. Методы сборки модулей: монолитный и инкрементальный. Особенности нисходящего тестирования и восходящего тестирования.

Автоматический прогон тестов. Структура тестового прогона. Инструментальная система автоматизации тестирования. Модульное тестирование. Основные принципы модульного тестирования. Требования к системам тестирования. Тестовые фреймворки CppUnit, Boost, t2c. Задачи и особенности. Достоинства и недостатки. Пример.

Интегрированное тестирование программных продуктов с использованием Visual Studio 2022 Community. Платформа тестирования для модульного теста Google Test, Boost Test, CTest. Ручное тестирование программ.