МИНОБРНАУКИ РОССИИ

Юго-Западный государственный университет

УТВЕРЖДАЮ:
Заведующий кафедрой
программной инженерии

_____ А.В. Малышев
(подпись, инициалы, фамилия)

«12» мая 2022 г.

ОЦЕНОЧНЫЕ СРЕДСТВА
для текущего контроля успеваемости
и промежуточной аттестации обучающихся
по дисциплине

Проектирование и архитектура программных систем
(наименование дисциплины)

ОПОП ВО 09.03.04 Программная инженерия
код и наименование ОПОП ВО

Курск – 2022

1 ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ

1.1 ВОПРОСЫ ДЛЯ УСТНОГО ОПРОСА

Тема 1. Модели предметной области.

1. Что такое модель?
2. Что такое язык UML?
3. Какие существуют виды моделей?

Тема 2. Объектно-ориентированные модели.

1. Что такое объектно-ориентированный анализ?
2. Какие существуют виды диаграмм для моделирования?
3. Что такое архитектура программ?

Тема 3. Основные принципы проектирования программных систем.

1. В чем заключается суть проектирования программных систем?
2. Какие существуют общие принципы проектирования?
3. Какие существуют способы разделения на модули?

Тема 4. Архитектура программных систем.

1. Что такое архитектура программных систем?
2. Какие существуют виды архитектур?
3. Каким образом используются шаблоны архитектуры?

Тема 5. Проектирование пользовательского интерфейса.

1. Что такое пользовательский интерфейс?
2. Какие существуют виды пользовательского интерфейса?
3. Назовите основные этапы проектирования пользовательского интерфейса.

Тема 6. Проектирование компонентов программной системы.

1. Что такое компонент программной системы?
2. Какие существуют способы проектирования компонентов?
3. Назовите примеры компонентов программной системы.

Тема 7. Объектно-ориентированное проектирование программных систем.

1. Что такое объектно-ориентированное проектирование?
2. Какие существуют особенности объектно-ориентированного проектирования?
3. Назовите основные этапы объектно-ориентированного проектирования.

Тема 8. Метрики качества программной системы.

1. Что такое метрики качества программной системы?
2. Какие существуют виды метрик программы и процесса?
3. Назовите способы измерения метрик программной системы.

Тема 9. Тестирование и валидация проекта программной системы.

1. Что такое тестирование и валидация?
2. Какие существуют виды тестирования?
3. Назовите основные стратегии тестирования программной системы.

Шкала оценивания: 5-ти балльная.

Критерии оценивания:

Каждый ответ оценивается по дихотомической шкале:

правильно – 1 балл, неправильно – 0 баллов.

Применяется следующая шкала перевода баллов в оценку по 5-балльной шкале:

- 22-27 баллов соответствуют оценке «отлично»;
- 17-21 баллов – оценке «хорошо»;
- 12-16 баллов – оценке «удовлетворительно»;
- 11 баллов и менее – оценке «неудовлетворительно».

1.2 ВОПРОСЫ ДЛЯ СОБЕСЕДОВАНИЯ

Тема 1. Модели предметной области.

1. Приведите пример модели предметной области.
2. Какие диаграммы UML используются для моделирования?
3. Зачем применяется моделирование предметной области?

Тема 2. Объектно-ориентированные модели.

1. В каких случаях применяется объектно-ориентированный анализ?

2. Какие виды диаграмм используются для объектно-ориентированного анализа?

3. Что такое анализ предметной области?

Тема 3. Основные принципы проектирования программных систем.

1. Зачем используется этап проектирования программных систем?

2. Какие существуют критерии при проектировании?

3. Как происходит структурное проектирование программ?

Тема 4. Архитектура программных систем.

1. Что такое программная система?

2. Зачем используется клиент-серверная архитектура?

3. Назовите основные структурные шаблоны архитектуры?

Тема 5. Проектирование пользовательского интерфейса.

1. Что пользовательский интерфейс?

2. Когда применяется графический пользовательский интерфейс?

3. Назовите основные критерии качества пользовательского интерфейса.

Тема 6. Проектирование компонентов программной системы.

1. Какие бывают компоненты программной системы?

2. Чем отличается проектирование компонентов от проектирования архитектуры?

3. Назовите типичные компоненты программной системы.

Тема 7. Объектно-ориентированное проектирование программных систем.

1. Когда используется объектно-ориентированное проектирование?

2. В чем заключается объектно-ориентированное проектирование?

3. Назовите достоинства и недостатки объектно-ориентированного проектирования.

Шкала оценивания: 5-ти балльная.

Критерии оценивания:

Каждый ответ оценивается по дихотомической шкале:

правильно – 1 балл, неправильно – 0 баллов.

Применяется следующая шкала перевода баллов в оценку по 5-балльной шкале:

- 18-21 баллов соответствуют оценке «отлично»;
- 14-17 баллов – оценке «хорошо»;
- 10-13 баллов – оценке «удовлетворительно»;
- 9 баллов и менее – оценке «неудовлетворительно».

1.3 ПРОИЗВОДСТВЕННЫЕ ЗАДАЧИ

Тема 1. Модели предметной области.

Производственная задача №1

Разработайте структурную модель программной системы для интегрированной среды разработки. Она должна включать в себя редактор кода, компилятор. Используйте язык моделирования UML.

Производственная задача №2

Разработайте модель данных для интернет-магазина. Она должна включать возможности размещения товаров любых категорий, поддерживать пользователей, обеспечивать доставку и оплату товара.

Производственная задача №3

Разработайте модель поведения виртуальной машины – интерпретатора. Машина производит арифметические вычисления, поддерживает функции и процедуры, обеспечивает ввод и вывод.

Тема 2. Объектно-ориентированные модели.

Производственная задача №1

Разработайте объектно-ориентированную модуль графического пользовательского интерфейса. Модель должна включать типичные элементы пользовательского интерфейса такие как окна, кнопки, поля ввода, списки, полосы прокрутки, меню. Также должна быть возможность создавать новые элементы

управления на основе имеющихся. Внешний вид элементов может быть изменен.

Производственная задача №2

Разработайте модель программной системы для поддержки компьютерной графики. Программная система должна поддерживать пользовательские мировые системы координат, аффинные преобразования, окна и области отсечения, спрайтовую анимацию.

Тема 4. Архитектура программных систем.

Производственная задача №1

Разработайте архитектуру программной системы для интегрированной среды разработки. Она должна включать в себя редактор кода, компилятор, отладчик. Спроектируйте компоненты программной системы.

Производственная задача №2

Разработайте архитектуру программной системы для поддержки компьютерной графики. Программная система должна поддерживать пользовательские мировые системы координат, аффинные преобразования, окна и области отсечения, спрайтовую анимацию, загрузку изображений.

Производственная задача №3

Разработайте архитектуру программной системы для интернет-магазина. Программная система должна включать возможности размещения товаров любых категорий, поддерживать пользователей, обеспечивать доставку и оплату товара, авторизацию пользователей.

Тема 5. Проектирование пользовательского интерфейса.

Производственная задача №1

Разработайте пользовательский интерфейс интернет-магазина. Он должен включать в себя: авторизацию и регистрацию пользователей, просмотр товаров, карточки товаров, корзину товаров, страницу оформления заказа.

Шкала оценивания: 5-ти балльная.

Критерии оценивания:

оценка **«отлично»** выставляется обучающемуся, если задача решена в полной мере, в программе выполняются все необходимые функции, программа работает корректно на всех тестовых входных данных;

оценка **«хорошо»** выставляется обучающемуся, если задача решена, в программе выполняются большинство функций, программа работает правильно на всех тестовых входных данных;

оценка **«удовлетворительно»** выставляется обучающемуся, если задача решена не полностью, в программе выполняются часть заданных функций, программа работает правильно на всех тестовых входных данных;

оценка **«неудовлетворительно»** выставляется обучающемуся, если в программе реализованы часть функций, программа работает неправильно на некоторых тестовых входных данных.

2 ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ

2.1 ТЕМЫ КУРСОВЫХ РАБОТ

1. Создание DNS сервера с использованием библиотеки socket на языке программирования Python.

2. Программная система ORM DB.

3. Проектирование программного обеспечения для агентства недвижимости.

4. Программа генерации и визуализации природных ландшафтов на основе шума Перлина

5. Программная система – социальная сеть.

6. Программа распознавания изображений с помощью нейронных сетей.

7. Почтовый сервер на основе SMTP протокола.

8. CRM система.

9. Программа сравнения и слияния файлов и папок.

10. Веб-сайт интернет-магазина.

11. Приложение для обмена сообщениями и файлами.

12. Программная система для просмотра Интернет страниц.

13. Мобильное приложение для удаленного управления устройством на платформе Android.

14. Сайт букмекерской конторы.

15. Интегрированная среда разработки.

16. Компилятор языка программирования.

17. Интерпретатор байт-кодов языка программирования.

18. Системный отладчик с графическим интерфейсом.

19. Библиотека графического пользовательского интерфейса.

20. Платформа для компьютерной графики.

21. Растровый графический редактор.

22. Векторный графический редактор.

23. Система автоматизированного проектирования.

24. Сервер для хранения и обмена файлов.

25. Программная система быстрого обмена сообщениями.

26. Облачное хранилище файлов.

27. Система управления базами данных.

28. Текстовый процессор.

29. Электронная таблица.

30. Программная система учета и планирования ресурсов предприятия.

2.2 БАНК ВОПРОСОВ И ЗАДАНИЙ В ТЕСТОВОЙ ФОРМЕ

Вопросы в закрытой форме.

1. Логическая модель это:

- Описание ключевых абстракций программного обеспечения (классы, интерфейсы и т. п.), т. е. средства, обеспечивающих требуемую функциональность
- Определение реальной организации программных модулей в среде разработки

- Отображение особенностей размещения программных компонентов на конкретном оборудовании

2. Модель использования это:

- Описание функциональности программного обеспечения с точки зрения пользователя
- Отображение особенностей размещения программных компонентов на конкретном оборудовании
- Представление разрабатываемого программного обеспечения в виде совокупности объектов, в процессе взаимодействия которых через передачу сообщений и происходит выполнение требуемых функций

3. Классы-сущности используют для:

- Моделирования последовательного поведения, заложенного в один или несколько вариантов использования
- Обеспечения взаимодействие между действующими лицами и внутренними элементами системы
- Представления сущностей реального мира или внутренних элементов системы, например структур данных

4. Процедурно-ориентированный интерфейс-меню:

- организует взаимодействие с пользователем в консольном режиме
- позволяет пользователю выбирать необходимые операции из специального списка, выводимого ему программой
- поддерживает концепцию интерактивного взаимодействия с программным обеспечением, осуществляя визуальную обратную связь с пользователем и возможность прямого манипулирования объектами и информацией на экране

5. За счет чего осуществляется минимизация рисков в гибких методологиях?

- за счет постоянного тестирования
- за счет специальных процессов управления рисками

- за счет сведения разработки к серии коротких циклов

6. Какой параметр не осложняет отладку программы?

- личный фактор программиста
- возможности взаимовлияния ошибок
- написания отдельных частей программы разными программистами

7. Сколько существует стадий тестирования?

- 2
- 3
- 1

8. Выберете средства ввода сообщений:

- Синтезаторы, звукогенераторы
- Принтеры, графопостроители
- Клавиатура, планшеты, сканеры

9. Какого метода при формировании тестовых наборов для функционального тестирования не существует?

- анализ граничных значений
- анализ причинно-следственных связей
- анализ разбиения

10. В какой модели жизненного цикла программного обеспечения переход на следующую стадию проектирования осуществляется только после того, как будет завершена работа на текущей стадии:

- спиральной
- каскадной
- пошаговой

11. Укажите, что из перечисленного ниже НЕ относится к принципам гибких методологий:

- Самый эффективный и продуктивный метод передачи информации команде разработчиков и обмена мнениями внутри неё - разговор лицом к лицу

- Гибкие процессы способствуют долгосрочной разработке с постоянным высоким темпом работ!
- Работающая программа - основной показатель прогресса в проекте

12. Рабочий элемент, который используется для фиксации в проекте событий или объектов, которые создают проблемы в выполнении проекта и должны быть устранены в ходе текущей или будущей итерации:

- Препятствие
- Ошибка
- Задача

13. Тестирование, при котором тестировщик не имеет заранее определенных тестовых сценариев и пытается интуитивно исследовать возможности программного продукта:

- Исследовательское тестирование
- Модульное тестирование
- Функциональное тестирование

14. Тестирование, которое представляет собой функциональные испытания, которые должны подтвердить то, что программный продукт соответствует требованиям и ожиданиям пользователей и заказчиков:

- Регрессионное тестирование
- Приемочное тестирование
- Функциональное тестирование

15. Какая задача не ставится на этапе анализа:

- Разработать алгоритм решения задачи
- Уточнить требуемое поведение разрабатываемого программного обеспечения
- Разработать концептуальную модель его предметной области с точки зрения поставленных задач

16. Модель реализации это:

- Определение реальной организации программных модулей в среде разработки
- Отображение особенностей размещения программных компонентов на конкретном оборудовании
- Описание ключевых абстракций программного обеспечения (классы, интерфейсы и т. п.), т. е. средства, обеспечивающих требуемую функциональность

17. Модель процессов это:

- Определение реальной организации программных модулей в среде разработки
- Отображение особенностей размещения программных компонентов на конкретном оборудовании
- Отображение организации вычислений и оперирование понятиями «процессы» и «нити», позволяющее оценить производительность, масштабируемость и надежность программного обеспечения

18. Модель развертывания это:

- Отображение особенностей размещения программных компонентов на конкретном оборудовании
- Определение реальной организации программных модулей в среде разработки
- Описание функциональности программного обеспечения с точки зрения пользователя

19. Основные варианты использования обеспечивают:

- Выполнение необходимых настроек системы и ее обслуживание (например, архивирование информации и т. п.)
- Требуемую функциональность разрабатываемого программного обеспечения
- Дополнительные удобства для пользователя (как правило, реализуются в том случае, если не требуют серьезных затрат каких-либо ресурсов ни при разработке, ни при эксплуатации)

20. Вспомогательные варианты использования обеспечивают:

- Требуемую функциональность разрабатываемого программного обеспечения
- Дополнительные удобства для пользователя (как правило, реализуются в том случае, если не требуют серьезных затрат каких-либо ресурсов ни при разработке, ни при эксплуатации)
- Выполнение необходимых настроек системы и ее обслуживание (например, архивирование информации и т. п.)

21. Дополнительные варианты использования обеспечивают:

- Выполнение необходимых настроек системы и ее обслуживание (например, архивирование информации и т. п.)
- Дополнительные удобства для пользователя (как правило, реализуются в том случае, если не требуют серьезных затрат каких-либо ресурсов ни при разработке, ни при эксплуатации)

22. Граничные классы используют для:

- обеспечения взаимодействие между действующими лицами и внутренними элементами системы
- представления сущностей реального мира или внутренних элементов системы, например структур данных
- моделирования последовательного поведения, заложенного в один или несколько вариантов использования

23. Управляющие классы используют для:

- представления сущностей реального мира или внутренних элементов системы, например структур данных
- обеспечения взаимодействие между действующими лицами и внутренними элементами системы
- моделирования последовательного поведения, заложенного в один или несколько вариантов использования

24. Примитивный процедурно-ориентированный интерфейс:

- поддерживает концепцию интерактивного взаимодействия с программным обеспечением, осуществляя визуальную обратную связь с пользователем и возможность прямого манипулирования объектами и информацией на экране
- позволяет пользователю выбирать необходимые операции из специального списка, выводимого ему программой
- организует взаимодействие с пользователем в консольном режиме

25. Объектная декомпозиция это:

- Описание функциональности программного обеспечения с точки зрения пользователя
- Описание ключевых абстракций программного обеспечения (классы, интерфейсы и т. п.), т. е. средства, обеспечивающих требуемую функциональность
- Представление разрабатываемого программного обеспечения в виде совокупности объектов, в процессе взаимодействия которых через передачу сообщений и происходит выполнение требуемых функций

26. Процедурно-ориентированный интерфейс со свободной навигацией:

- поддерживает концепцию интерактивного взаимодействия с программным обеспечением, осуществляя визуальную обратную связь с пользователем и возможность прямого манипулирования объектами и информацией на экране
- позволяет пользователю выбирать необходимые операции из специального списка, выводимого ему программой
- организует взаимодействие с пользователем в консольном режим

27. Проектирование абстрактных диалогов это:

- выбор основных и дополнительных устройств и проектирование процессов ввода-вывода для каждого диалога, а также уточнение передаваемых сообщений
- определение типа и формы каждого диалога, а также синтаксиса и семантики используемых языков

- определение множества необходимых диалогов, их основных сообщений и возможных сценариев

28. Проектирование конкретных диалогов это:

- определение множества необходимых диалогов, их основных сообщений и возможных сценариев
- определение типа и формы каждого диалога, а также синтаксиса и семантики используемых языков
- выбор основных и дополнительных устройств и проектирование процессов ввода-вывода для каждого диалога, а также уточнение передаваемых сообщений

29. Проектирование технических диалогов это:

- определение множества необходимых диалогов, их основных сообщений и возможных сценариев
- определение типа и формы каждого диалога, а также синтаксиса и семантики используемых языков
- выбор основных и дополнительных устройств и проектирование процессов ввода-вывода для каждого диалога, а также уточнение передаваемых сообщений

30. К логическим ошибкам некорректного использования переменных относятся:

- игнорирование системных соглашений, нарушение типов и последовательности при передаче параметров, несоблюдение единства единиц измерения формальных и фактических параметров, нарушение области действия локальных и глобальных переменных
- некорректные вычисления над неарифметическими переменными, некорректное использование целочисленной арифметики, некорректное преобразование типов данных в процессе вычислений, ошибки, связанные с незнанием приоритетов выполнения операций для арифметических и логических выражений, и т. п.

- неудачный выбор типов данных, использование переменных до их инициализации, использование индексов, выходящих за границы определения массивов, нарушения соответствия типов данных при использовании явного или неявного переопределения типа данных, расположенных в памяти при использовании нетипизированных переменных, открытых массивов, объединений, динамической памяти, адресной арифметики и т. п.

31. К логическим ошибкам межмодульного интерфейса относятся:

- некорректные вычисления над неарифметическими переменными, некорректное использование целочисленной арифметики, некорректное преобразование типов данных в процессе вычислений, ошибки, связанные с незнанием приоритетов выполнения операций для арифметических и логических выражений, и т. п.
- игнорирование системных соглашений, нарушение типов и последовательности при передаче параметров, несоблюдение единства единиц измерения формальных и фактических параметров, нарушение области действия локальных и глобальных переменных
- неудачный выбор типов данных, использование переменных до их инициализации, использование индексов, выходящих за границы определения массивов, нарушения соответствия типов данных при использовании явного или неявного переопределения типа данных, расположенных в памяти при использовании нетипизированных переменных, открытых массивов, объединений, динамической памяти, адресной арифметики и т. п.

32. К логическим ошибкам вычислений относятся:

- некорректные вычисления над неарифметическими переменными, некорректное использование целочисленной арифметики, некорректное преобразование типов данных в процессе вычислений, ошибки, связанные с незнанием приоритетов выполнения операций для арифметических и логических выражений, и т. п.

- игнорирование системных соглашений, нарушение типов и последовательности при передаче параметров, несоблюдение единства единиц измерения формальных и фактических параметров, нарушение области действия локальных и глобальных переменных
- неудачный выбор типов данных, использование переменных до их инициализации, использование индексов, выходящих за границы определения массивов, нарушения соответствия типов данных при использовании явного или неявного переопределения типа данных, расположенных в памяти при использовании нетипизированных переменных, открытых массивов, объединений, динамической памяти, адресной арифметики и т. п.

33. Какого метода отладки программного обеспечения не существует?

- обратного прослеживания
- индукции
- прямого прослеживания

34. Какого метода и средства получения дополнительной информации не существует?

- отладочный вывод
- интегрированные средства отладки
- зависимые отладчики

35. Что не позволяют делать средства отладки программы?

- отображать содержимое любых переменных при пошаговом выполнении
- находить логические ошибки в коде
- выполнять программу по шагам, причем как с заходом в подпрограммы, так и выполняя их целиком

36. На чем основывается структурный подход тестирования?

- структура программного обеспечения известна («стеклянный ящик»)!
- структура программного обеспечения узнается на ходе тестирования
- структура программного обеспечения не известна («черный ящик»)

37. Что не предполагает минимальное тестирование?

- тестирование работы логики программы
- тщательную проверку руководства
- тестирование граничных значений

38. Выберете средства вывода сообщений:

- Принтеры, графопостроители, Синтезаторы, звукогенераторы
- Клавиатура, планшеты
- Сканеры

39. Что такое диаграмма последовательностей системы?

- графическая модель, которая для определенного сценария варианта использования показывает генерируемые действующими лицами события и их порядок
- графическая модель, которая показывает изменение значения переменной во время работы программы
- графическая модель, которая показывает нагрузку системы во время работы программы

40. Какого подхода для формирования тестов для тестирования не существует?

- Логический
- Функциональный
- Структурный

41. Какой стадии тестирования не существует?

- автономное тестирование компонентов программного обеспечения
- логическое тестирование разрабатываемого программного обеспечения
- комплексное тестирование разрабатываемого программного обеспечения

42. Какого основного принципа тестирования не существует?

- предполагаемые результаты должны быть известны до тестирования
- количество тестов должно быть ограничено
- необходимо досконально изучать результаты каждого теста

43. Основным методами ручного контроля не является:

- сквозные просмотры
- прогнозирование результатов теста
- инспекции исходного текста

44. Что делают объекты-данные?

- снабжают пользователя информацией
- могут манипулировать своими внутренними объектами
- часто представляют устройства, существующие в реальном мире: телефоны, факсы, принтеры и т. д.

45. Где применяются диаграммы компонентов?

- при проектировании физической структуры разрабатываемого программного обеспечения
- при написании программного обеспечения
- при отладке программного обеспечения

46. Объектно-ориентированные пользовательские интерфейсы:

- Акцент делается на входные данные и результаты
- Обеспечивают пользователей функциями, необходимыми для выполнения задач
- Пиктограммы представляют приложения, окна или операции

47. Технология разработки программного обеспечения это...

- комплекс организационных мер, направленных на разработку программных продуктов
- операции и приемы разработки программных продуктов
- вычислительные комплексы

48. Жизненный цикл программного обеспечения это...

- период времени эксплуатации программного продукта
- период времени, который начинается с момента разработки программного продукта и заканчивается в момент передачи его в эксплуатацию

- период времени, который начинается с момента принятия решения о необходимости создания программного продукта и заканчивается в момент его полного изъятия из эксплуатации

49. На чем основывается функциональный подход тестирования?

структура программного обеспечения не известна («черный ящик»)

структура программного обеспечения известна («стеклянный ящик»)

структура программного обеспечения узнается на ходе тестирования

50. Какого критерия для формирования тестовых наборов для тестирования маршрутов не существует?

- покрытие условий
- покрытие тестов
- покрытие операторов

51. Что определяет технология технология Drag and Drop («перетащил и бросил»)?

- основные принципы отладки
- основные принципы тестирования
- основные принципы прямого манипулирования

52. Что делают объекты-контейнеры?

- снабжают пользователя информацией
- могут манипулировать своими внутренними объектами
- часто представляют устройства, существующие в реальном мире: телефоны, факсы, принтеры и т. д.

53. Что делают объекты-устройства?

- могут манипулировать своими внутренними объектами
- часто представляют устройства, существующие в реальном мире: телефоны, факсы, принтеры и т. д.
- снабжают пользователя информацией

54. Что такое метафора?

- Мысленный перенос свойств или признаков одного объекта на другой, чем-то аналогичный первому
- Аналогия технологии Drag and Drop
- Оборот речи, состоящий в употреблении слов и выражений в переносном смысле на основе какой-н. аналогии, сходства, сравнения

55. Какого варианта использования не существует?

- Основные
- Вспомогательные
- Логические

56. Какого вида сообщений не существует?

- Дополнительные
- Выходные
- Входные

57. Какого подхода к разработке интерфейсов не существует?

- процедурно-ориентированный
- графический
- объектно-ориентированный

58. Какого вида процедурно-ориентированного интерфейса не существует?

- Сложной навигацией
- Меню
- «Примитивные»

59. Процедурно-ориентированные пользовательские интерфейсы:

- Обеспечивают пользователям возможность взаимодействия с объектами
- Акцент делается на входные данные и результаты
- Пиктограммы представляют приложения, окна или операции

60. Классическими моделями жизненного цикла программного обеспечения является:

- водопадная

- спиральная
- итерационная

61. Какие схемы определяют функциональность системы и описывают с точки зрения пользователей их возможные действия с программным продуктом?

- Схемы (диаграммы) вариантов использования UML
- Схемы (диаграммы) последовательностей UML
- Схемы (диаграммы) активности UML

62. Какие схемы описывают бизнес-процесс или программный процесс в виде потока работ через последовательные действия:

- Схемы (диаграммы) вариантов использования UML
- Схемы (диаграммы) активности UML
- Схемы (диаграммы) последовательностей UML

63. Какие схемы описывают распределение программных составляющих приложения, позволяя наглядно отобразить на высоком уровне структуру компонентов и служб:

- Схемы (диаграммы) активности UML
- Схемы (диаграммы) слоев UML
- Схемы (диаграммы) компонентов UML

64. Рабочий элемент, который представляет собой требование, которое необходимо выполнить при реализации проекта:

- Пользовательское описание функциональности
- Задача
- Тестовый случай

65. Рабочий элемент, который создается в проекте для назначения и выполнения работы:

- Препятствие
- Задача
- Ошибка

66. Рабочий элемент, который используется для отслеживания и мониторинга проблем в программном продукте:

- Препятствие
- Задача
- Ошибка

67. Рабочий элемент, который используется для фиксации в проекте событий или объектов, которые создают проблемы в выполнении проекта и должны быть устранены в ходе текущей или будущей итерации:

- Препятствие
- Ошибка
- Задача

68. Тестирование, при котором проверяется корректная совместная работа компонентов программного продукта:

- Исследовательское тестирование
- Интеграционное тестирование

69. Тестирование, при котором осуществляется проверка конкретных требований к ПО, и которое проводится после добавление к системе новых функций:

- Интеграционное тестирование
- Приемочное тестирование
- Функциональное тестирование

70. Тестирование, которое применяется при внесении изменений в программное обеспечение, с целью проверки корректности работы компонентов системы:

- Интеграционное тестирование
- Регрессионное тестирование
- Функциональное тестирование

Шкала оценивания результатов тестирования: в соответствии с действующей в университете балльно-рейтинговой системой оценивание результа-

тов промежуточной аттестации обучающихся осуществляется в рамках 100-балльной шкалы, при этом максимальный балл по промежуточной аттестации обучающихся по очной форме обучения составляет 36 баллов, по очно-заочной и заочной формам обучения – 60 баллов (установлено положением П 02.016).

Максимальный балл за тестирование представляет собой балл по промежуточной аттестации для данной формы обучения (36 или 60).

Соответствие 100-балльной и пятибальной шкал:

Сумма баллов по 100-балльной шкале	*Оценка по дихотомической шкале*
100-85	отлично
84-70	хорошо
69-50	удовлетворительно
49 и менее	неудовлетворительно