СОГЛАСОВАНО
Начальник отдела эксплуатации и внедрения информационных систем областного государственного автономного учреждения здравоохранения СОМИАЦ
_____ Я. А. Комиссаров
«31» 08 2023 г.

УТВЕРЖДАЮ
Заместитель директора по учебной работе
_____ И. В. Иванешко
«31» 08 2023 г.

Контрольно-оценочные материалы для промежуточной аттестации по
МДК 01.02Поддержка и тестирование программных модулей
для специальности 09.02.07 Информационные системы и программирование

Дифференцированный зачет является промежуточной формой контроля, подводит итог освоения МДК 01.02Поддержка и тестирование программных модулей.

В результате освоения МДК студент должен освоить следующие профессиональные компетенции:

ВД. Разработка модулей программного обеспечения для компьютерных систем
ПК 1.3. Выполнять отладку программных модулей с использованием специализированных программных средств
ПК 1.4. Выполнять тестирование программных модулей
ПК 1.5. Осуществлять рефакторинг и оптимизацию программного кода

Дифференцированный зачет по МДК проводится в форме тестирования. К сдаче дифференцированного зачета допускаются студенты, сдавшие все практические и лабораторные работы. Тест содержит 20 вопросов (суммарно тестовых позиций и теоретических вопросов с кратким ответом), выбираемых случайным образом программой из каждого блоков (состоящих первый блок 150 вопросов, второй блок 150 вопросов) заданий по 10 вопросов. Время тестирования – 80 минут для каждой подгруппы (по 3 минуты на каждый вопрос из первого блока, по 5 минуты на каждый вопрос закрытого типа). Для прохождения тестирования, студенты разбиваются на три подгруппы (по количеству персональных компьютеров в сдаваемой аудитории). Время на подготовку и проверку тестирования – 30 мин.

Шкала оценивания образовательных результатов:

Оценка	Критерии
«отлично»	Студент набрал 5 баллов (по весу критерия)
«хорошо»	Студент набрал 4 балла (по весу критерия)
«удовлетворительно»	Студент набрал 3 балла (по весу критерия)
«неудовлетворительно»	Студент набрал 0-2 балла (по весу критерия)

Первый блок заданий
Формируемые ПК1.3, ПК1.4, ПК 1.5

Проверяемая компетенция – ПК1.3.

1. Процесс локализации и исправления ошибок, обнаруженных при тестировании программного обеспечения это
   1) отладка
   2) локализация
   3) определение данных
   4) использование данных
2. Процесс определения оператора программы, выполнение которого вызвало нарушение нормального вычислительного процесса это
   1) локализация
   2) определение данных
   3) использование данных
   4) отладка
3. Проверка соответствия ПО требованиям, осуществляемая с помощью наблюдения за его работой в специальных, искусственно построенных ситуациях это
   1) тестирование
   2) локализация
   3) определение данных
   4) использование данных
4. Тестирование, предназначенное для проверки правильности отдельных модулей, вне зависимости от их окружения, это
   1) модульное тестирование
   2) интеграционное тестирование
   3) системное тестирование
   4) Регрессионное тестирование
5. Тестирование, предназначенное для проверки правильности взаимодействия модулей некоторого набора друг с другом, это
   1) интеграционное тестирование
   2) системное тестирование
   3) модульное тестирование
   4) Регрессионное тестирование
6. Тестирование, предназначенное для проверки правильности работы системы в целом, ее способности правильно решать поставленные пользователями задачи в различных ситуациях, это
   1) системное тестирование
   2) модульное тестирование
   3) интеграционное тестирование
   4) Регрессионное тестирование
7. Документ, который описывает конкретные шаги, условия и параметры, необходимые для анализа реализации тестируемой функции, это
   1) тестовый случай
   2) Таблица решений
   3) Тестовый сценарий
   4) описание теста
8. Алгоритм проверки некоторой функциональности программы в совокупности с ожидаемыми результатами это
   1) Тестовый сценарий
   2) тестовый случай

3) Лист проверки
4) Таблица решений

9. Шаги теста, которые переводят систему в состояние, пригодном для проведения проверки, это
1) Предусловия теста
2) тестовый случай
3) описание теста
4) постусловия теста

10. Шаги теста, которые переводят систему из состояния в состояние
1) описание теста
2) тестовый случай
3) Предусловия теста
4) постусловия теста

11. Шаги теста, которые переводят систему в изначальное положение
1) постусловия теста
2) тестовый случай
3) описание теста
4) Предусловия теста

12. Модуль, обеспечивающий вызов и передачу параметров модулю необходимых входных данных и обработку результатов, это
1) драйвер
2) заглушка
3) Таблица решений
4) Лист проверки

13. Модуль, имитирующий работы модулей, вызываемых тестируемым модулем, это
1) заглушка
2) драйвер
3) Таблица решений
4) Лист проверки

14. Набор тестов, который в совокупности должен обеспечить прохождение каждой команды не менее одного раза, соответствует условию критерия
1) тестирования команд (критерий С0)
2) тестирования ветвей (критерий С1)
3) тестирования путей (критерий С2)
4) покрытия вызовов

15. Набор тестов, который в совокупности должен обеспечить прохождение каждой ветви не менее одного раза, соответствует условию критерия
1) тестирования ветвей (критерий С1)
2) тестирования путей (критерий С2)
3) тестирования команд (критерий С0)
4) покрытия вызовов

16. Набор тестов, который в совокупности должен обеспечить прохождение каждого пути не менее 1 раз, соответствует условию критерия
1) тестирования путей (критерий С2)
2) тестирования ветвей (критерий С1)
3) тестирования команд (критерий С0)
4) покрытия вызовов

17. Набор тестов, по которому каждая функция программы должна быть вызвана хотя бы один раз, соответствует критерию
1) покрытия функций программы
2) покрытия вызовов

3) охвата всех DU-цепочек программы
4) тестирования ветвей

18. Набор тестов, по которому каждый вызов каждой функции в программе должен быть осуществлен хотя бы один раз, соответствует критерию
1) покрытия вызовов
2) покрытия функций программы
3) охвата всех DU-цепочек программы
4) тестирования ветвей

19. Набор тестов, по которому выполняется проверка всех составляющих логического условия (каждое атомарное булево выражение приняло значения и «истина», и «ложь»), это
1) Критерий охвата условий
2) Критерий охвата параметров
3) Критерий охвата циклов
4) покрытие вызовов

20. Набор тестов, по которому выполняется проверка всех значений параметров метода, это
1) Критерий охвата параметров
2) Критерий охвата условий
3) Критерий охвата циклов
4) покрытие вызовов

21. Набор тестов, по которому все циклы должны исполняться 0, 1, ..., N раз, это
1) Критерий охвата циклов
2) Критерий охвата условий
3) Критерий охвата параметров
4) покрытие вызовов

22. Действия, изменяющие элемент данных, это
1) определение данных
2) использование данных
3) покрытие вызовов
4) Предусловие теста

23. Признак определения —
1) имя элемента стоит в левой части оператора присваивания
2) имя элемента стоит в правой части оператора присваивания
3) имя элемента стоит справа относительно арифметического оператора
4) имя элемента стоит слева относительно арифметического оператора

24. Применение элемента в выражении, где происходит обращение к элементу данных, но не изменение элемента, это
1) использование данных
2) определение данных
3) Предусловие теста
4) покрытие функций программы

25. Признак использования —
1) имя элемента стоит в правой части оператора присваивания
2) имя элемента стоит в левой части оператора присваивания
3) имя элемента стоит справа относительно арифметического оператора
4) имя элемента стоит слева относительно арифметического оператора

26. Способ DU-тестирования требует
1) охвата всех DU-цепочек программы
2) покрытия функций программы
3) покрытия вызовов
4) последующего выполнения полной регрессия

27. Тестирование, которое проводится для проверки того, что новые изменения не влияют на существующие функции, функциональность приложения или программного обеспечения, это

1) Регрессионное тестирование
2) модульная регрессия
3) полная регрессия
4) частичная регрессия

28. Тип регрессионного тестирования, который выполняется во время модульного тестирования, это

1) модульная регрессия
2) полная регрессия
3) Регрессионное тестирование
4) частичная регрессия

29. Тип регрессионного тестирования, который выполняется для проверки того, что код работает нормально после внесения изменений в код и что код интегрирован с существующим кодом или неизмененными модулями, это

1) частичная регрессия
2) модульная регрессия
3) полная регрессия
4) Регрессионное тестирование

30. Тип регрессионного тестирования, который выполняется, когда есть много изменений в коде и количестве модулей, это

1) полная регрессия
2) модульная регрессия
3) Регрессионное тестирование
4) частичная регрессия

31. Эквивалентное разбиение – это

1) метод тестирования «черного ящика»
2) метод разбиения классов эквивалентности
3) Анализ граничных значений
4) Интеграция «сверху вниз»

32. Метод тестирования, который состоит в том, чтобы исключить набор входных данных, которые заставляют систему вести себя одинаково и давать одинаковый результат при тестировании программы, это

1) метод разбиения классов эквивалентности
2) метод тестирования «черного ящика»
3) Анализ граничных значений
4) Инкрементальный подход

33. Разработка тестов методом черного ящика, при котором тестовые сценарии проектируются на основании граничных значений, это

1) Анализ граничных значений
2) метод тестирования «черного ящика»
3) Инкрементальный подход
4) Интеграция «сверху вниз»

34. Входное значение или выходные данные, которое находится на грани эквивалентной области или на наименьшем расстоянии от обеих сторон грани, например, минимальное или максимальное значение области это

1) Граничное значение
2) Таблица решений
3) Тестовый сценарий
4) описание теста

35. Таблица, отражающая комбинации входных данных и/или причин с соответствующими выходными данными и/или действиям (следствиям), которая может быть использована для проектирования тестовых сценариев это

1) Таблица решений
2) Граничное значение
3) Анализ граничных значений
4) Подход Большого взрыва

36. Стратегия интеграционного тестирования, по которой все компоненты собираются вместе, а затем тестируются, это

1) Подход Большого взрыва
2) Сэндвич (гибридная интеграция)
3) Инкрементальный подход
4) Интеграция «снизу вверх»

37. Стратегия интеграционного тестирования, по которой тестирование выполняется путем объединения двух или более логически связанных модулей, затем добавляются другие связанные модули и проверяются на правильность функционирования, это

1) Инкрементальный подход
2) Подход Большого взрыва
3) Сэндвич (гибридная интеграция)
4) Интеграция «снизу вверх»

38. Стратегия интеграционного тестирования, по которой каждый модуль на более низких уровнях тестируется с модулями более высоких уровней, пока не будут протестированы все модули, это

1) Интеграция «снизу вверх»
2) Подход Большого взрыва
3) Сэндвич (гибридная интеграция)
4) Инкрементальный подход

39. Стратегия интеграционного тестирования, по которой тестирование выполняется сверху вниз, следуя потоку управления программной системы, это

1) Интеграция «сверху вниз»
2) Подход Большого взрыва
3) Сэндвич (гибридная интеграция)
4) Инкрементальный подход

40. Стратегия интеграционного тестирования, по которой верхнеуровневые модули тестируются с нижнеуровневыми, а нижнеуровневые модули интегрируются с верхнеуровневыми, соответственно, и тестируются, это

1) Сэндвич (гибридная интеграция)
2) Подход Большого взрыва
3) Инкрементальный подход
4) Анализ граничных значений

41. Этап системного тестирования, на котором тестируемая система анализируется (проверяется) на наличие определенных свойств, которым надо уделить особое внимание, и определяются соответствующие тестовые случаи, это

1) Анализ
2) Выполнение и анализ результатов
3) Построение
4) Интеграция «сверху вниз»

42. Этап системного тестирования, на котором выбранные на стадии анализа тестовые случаи переводятся на язык программирования, это

1) Построение
2) Анализ

3) Выполнение и анализ результатов
4) Интеграция «сверху вниз»

43. Этап системного тестирования, на котором производится выполнение тестовых случаев. Полученные результаты анализируются, чтобы определить, успешно ли прошла система испытания на тестовом наборе, это
1) Выполнение и анализ результатов
2) Анализ
3) Построение
4) Интеграция «снизу вверх»

44. Основной документ этапа тестирования, который описывает работы по тестированию, начиная с описания объекта, стратегии, расписания, критериев начала и окончания тестирования, до необходимого в процессе работы оборудования, специальных знаний, а также оценки рисков с вариантами их разрешения, это
1) План тестирования
2) Лист проверки
3) Тестовый сценарий
4) описание теста

45. Документ, описывающий что должно быть протестировано это
1) Лист проверки
2) План тестирования
3) Тестовый сценарий
4) описание теста

46. Тип тестирования, когда тестированию подвергается собственно система, являющаяся основным выпускаемым продуктом, это
1) Тестирование основной функциональности
2) Тестирование пользовательской документации
3) Тестирование инсталляции
4) Регрессионное тестирование

47. Тип тестирования, который включает тестирование сценариев первичной инсталляции системы, сценариев повторной инсталляции (поверх уже существующей копии), тестирование деинсталляции, тестирование инсталляции в условиях наличия ошибок в инсталлируемом пакете, в окружении или в сценарии и т. п., это
1) Тестирование инсталляции
2) Тестирование основной функциональности
3) Тестирование пользовательской документации
4) Регрессионное тестирование

48. Тип тестирования, который включает проверку полноты и понятности описания правил и особенностей использования продукта, наличие описания всех сценариев и функциональности, синтаксис и грамматику языка, работоспособность примеров и т. п., это
1) Тестирование пользовательской документации
2) Тестирование основной функциональности
3) Тестирование инсталляции
4) Регрессионное тестирование

49. Анализ исходного кода, производимый без его исполнения, это
1) Статический анализ кода
2) Динамическое тестирование
3) Анализ отдельного модуля
4) Создание кейс-теста

50. Тестовая деятельность, предусматривающая эксплуатацию (запуск) программного продукта это
1) Динамическое тестирование

2) Статический анализ кода
3) Анализ отдельного модуля
4) Создание кейс-теста

Проверяемая компетенция – ПК1.4.

1. Этап модульного тестирования, на котором разработчик изучает внутреннюю структуру кода, функционал и поведение исследуемого компонента это
   1) Анализ отдельного модуля
   2) Создание кейс-теста
   3) Тестирование модуля
   4) Статический анализ кода
2. Этап, на котором создается сценарий или модель, которые должны показать, как проверяемый модуль ведет себя в реальной обстановке, это
   1) Создание кейс-теста
   2) Анализ отдельного модуля
   3) Тестирование модуля
   4) Статический анализ кода
3. Этап, на котором проверяемый компонент, предварительно изолированный от ядра приложения и других модулей, запускается в кейс-тесте, это
   1) Тестирование модуля
   2) Создание кейс-теста
   3) Анализ отдельного модуля
   4) Динамическое тестирование
4. Сценарий, который создает искусственную среду, максимально близкую к реальной, но без привлечения внешних ресурсов, которые обычно задействуются в работе программного обеспечения (веб-серверов, баз данных и т.д.), это
   1) Кейс-тест
   2) Валидация
   3) Верификация
   4) Динамическое тестирование
5. Процесс оценки системы или её компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, сформированным в начале этого этапа, это
   1) Верификация
   2) Кейс-тест
   3) Валидация
   4) Создание кейс-теста
6. Определение соответствия разрабатываемого ПО ожиданиям и потребностям пользователя, требованиям к системе это
   5) Валидация
   6) Кейс-тест
   7) Верификация
   8) Создание кейс-теста
7. Подвид тестирования производительности, сбор показателей и определение производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной системе (устройству) это
   1) нагрузочноетестирование (Performance / Load Testing)
   2) Тестирование локализации
   3) стрессовое тестирование (StressTesting)
   4) кроссплатформенное тестирование (CrossPlanformTesting)

8. Проверка на соответствие локальным стандартам, языковым стандартам и пользовательского интерфейса это

1) Тестирование локализации
2) нагрузочноетестирование (Performance / Load Testing)
3) стрессовое тестирование (StressTesting)
4) тестирование на отказ и восстановление (FailoverandRecoveryTesting)

9. Исследование изменений свойств системы или объекта в нестандартных условиях это

1) стрессовое тестирование (StressTesting)
2) нагрузочноетестирование (Performance / Load Testing)
3) тестирование на отказ и восстановление (FailoverandRecoveryTesting)
4) кроссплатформенное тестирование (CrossPlanformTesting)

10. Проверка успешной инсталляции и настройки, а также обновления или удаления программного обеспечения это

1) тестирование установки (Installationtesting)
2) нагрузочноетестирование (Performance / Load Testing)
3) Тестирование локализации
4) тестирование на отказ и восстановление (FailoverandRecoveryTesting)

11. Метод тестирования, направленный на установление степени удобства использования, обучаемости, понятности и привлекательности для пользователей разрабатываемого продукта в контексте заданных условий это

1) тестирование удобства пользования (UsabilityTesting)
2) нагрузочноетестирование (Performance / Load Testing)
3) Тестирование локализации
4) стрессовое тестирование (StressTesting)

12. Подвид тестирования производительности, который проверяет тестируемый продукт с точки зрения способности противостоять и успешно восстанавливаться после возможных сбоев, возникших в связи с ошибками ПО, отказами оборудования или проблемами связи/сети это

1) тестирование на отказ и восстановление (FailoverandRecoveryTesting)
2) стрессовое тестирование (StressTesting)
3) кроссплатформенное тестирование (CrossPlanformTesting)
4) тестирование установки (Installationtesting)

13. Специальный вид тестирования, направленный на проверку работы ПОпри различных аппаратных и программных конфигурациях системы это

1) тестирование конфигурации (ConfigurationTesting)
2) тестирование на отказ и восстановление (FailoverandRecoveryTesting)
3) кроссплатформенное тестирование (CrossPlanformTesting)
4) стрессовое тестирование (StressTesting)

14. Тип тестирования, который проверяет, работает ли приложение так, как ожидается, в нескольких браузерах, операционных системах и устройствах это

1) кроссплатформенное тестирование (CrossPlanformTesting)
2) тестирование на отказ и восстановление (FailoverandRecoveryTesting)
3) стрессовое тестирование (StressTesting)
4) тестирование установки (Installationtesting)

15. Набор атрибутов, относящихся к сути набора функций и их конкретным свойствам. Функциями являются те, которые реализуют установленные или предполагаемые потребности это

1) Функциональные возможности
2) Надежность
3) Практичность
4) Эффективность

16. Набор атрибутов, относящихся к способности программного обеспечения сохранять свой уровень качества функционирования при установленных условиях за установленный период времени это

1) Надежность
2) Функциональные возможности
3) Практичность
4) Эффективность

17. Набор атрибутов, относящихся к объему работ, требуемых для использования и индивидуальной оценки такого использования определенным или предполагаемым кругом пользователей это

1) Практичность
2) Функциональные возможности
3) Надежность
4) Эффективность

18. Набор атрибутов, относящихся к соотношению между уровнем качества функционирования программного обеспечения и объемом используемых ресурсов при установленных условиях это

1) Эффективность
2) Функциональные возможности
3) Практичность
4) Сопровождаемость

19. Набор атрибутов, относящихся вобъему работ, требуемых для проведения конкретных изменений (модификаций) это

1) Сопровождаемость
2) Надежность
3) Эффективность
4) Мобильность

20. Набор атрибутов, относящихся к способности программного обеспечения быть перенесенными из одного окружения в другое это

1) Мобильность
2) Надежность
3) Практичность
4) Эффективность

21. Атрибут программного обеспечения, относящийся к наличию и соответствию набора функций конкретным задачам это

1) Пригодность
2) Надежность
3) Эффективность
4) Сопровождаемость

22. Атрибут программного обеспечения, относящиеся к обеспечению правильности или соответствия результатов или эффектов это

1) Правильность
2) Практичность
3) Сопровождаемость
4) Пригодность

23. Атрибут программного обеспечения, относящиеся к способности его взаимодействовать с конкретными системами это

1) Способность к взаимодействию
2) Эффективность
3) Мобильность
4) Пригодность

24. Атрибут программного обеспечения, которые заставляют программу придерживаться соответствующих стандартов или соглашений, или положений законов, или подобных рекомендаций это

1) Согласованность
2) Мобильность
3) Пригодность
4) Правильность

25. Атрибут программного обеспечения, относящиеся к его способности предотвращать несанкционированный доступ, случайный или преднамеренный, к программам и данным это

1) Защищенность
2) Пригодность
3) Правильность
4) Способность к взаимодействию

26. Атрибут программного обеспечения, относящиеся к частоте отказов при ошибках в программном обеспечении это

1) Стабильность
2) Мобильность
3) Правильность
4) Согласованность

27. Атрибут программного обеспечения, относящиеся к его способности поддерживать определенный уровень качества функционирования в случаях программных ошибок или нарушения определенного интерфейса это

1) Устойчивость к ошибке
2) Правильность
3) Способность к взаимодействию
4) Согласованность

28. Атрибут программного обеспечения, относящиеся к его возможности восстанавливать уровень качества функционирования и восстанавливать данные, непосредственно поврежденные в случае отказа, а также к времени и усилиям, необходимым для этого, это

1) Восстанавливаемость
2) Способность к взаимодействию
3) Согласованность
4) Защищенность

29. Атрибут программного обеспечения, относящиеся к усилиям пользователя по пониманию общей логической концепции и ее применимости это

1) Понятность
2) Способность к взаимодействию
3) Согласованность
4) Стабильность

30. Атрибут программного обеспечения, относящиеся к усилиям пользователя по обучению его применению (например, оперативному управлению, вводу, выводу) это

1) Обучаемость
2) Согласованность
3) Защищенность
4) Устойчивость к ошибке

31. Атрибут программного обеспечения, относящиеся к усилиям пользователя по эксплуатации и оперативному управлению это

1) Простота использования
2) Защищенность
3) Стабильность

4) Устойчивость к ошибке

32. Атрибут программного обеспечения, относящиеся к временам отклика и отработки и к скоростям выполнения его функций это

1) Характер изменения во времени
2) Защищенность
3) Стабильность
4) Устойчивость к ошибке

33. Атрибут программного обеспечения, относящиеся к объему используемых ресурсов и продолжительности такого использования при выполнении функции это

1) Характер изменения ресурсов
2) Стабильность
3) Устойчивость к ошибке
4) Восстанавливаемость

34. Атрибут программного обеспечения, относящиеся к усилиям, необходимым для диагностики недостатков или случаев отказов при определении составных частей для модернизации это

1) Анализируемость
2) Стабильность
3) Восстанавливаемость
4) Понятность

35. Атрибут программного обеспечения, относящиеся к усилиям, необходимым для модификации, устранению отказа или для изменения условий эксплуатации это

1) Изменяемость
2) Стабильность
3) Восстанавливаемость
4) Понятность

36. Атрибут программного обеспечения, относящиеся к риску от непредвиденных эффектов модификации это

1) Устойчивость
2) Понятность
3) Обучаемость
4) Простота использования

37. Атрибут программного обеспечения, относящиеся к усилиям, необходимым для проверки модифицированного программного обеспечения это

1) Тестируемость
2) Восстанавливаемость
3) Понятность
4) Обучаемость

38. Атрибут программного обеспечения, относящиеся к удобству его адаптации к различным конкретным условиям эксплуатации, из применения других действий или способов, кроме тех, что предназначены для этого в рассматриваемом программном обеспечении это

1) Адаптируемость
2) Восстанавливаемость
3) Обучаемость
4) Простота использования

39. Атрибут программного обеспечения, относящиеся к усилиям, необходимым для внедрения программного обеспечения в конкретное окружение это

1) Простота внедрения
2) Обучаемость
3) Простота использования
4) Характер изменения ресурсов

40. Атрибут программного обеспечения, которые заставляют программу подчиняться стандартам или соглашениям, относящимся к мобильности это
1) Соответствие
2) Характер изменения ресурсов
3) Устойчивость
4) Простота внедрения

41. Атрибут программного обеспечения, относящиеся к простоте и трудоемкости его применения вместо другого конкретного программного средства в среде этого средства это
1) Взаимозаменяемость
2) Обучаемость
3) Устойчивость
4) Адаптируемость

42. Тесты, которые проверяют наличие требуемых действий в тестируемом объекте и их работоспособность это
1) Позитивные тесты
2) Негативные тесты

43. Тесты, которые проверяют действия тестируемого объекта в случае некорректного начала (например, неправильные входные данные или неправильная последовательность вызовов) это
1) Негативные тесты
2) Позитивные тесты

44. Численное значение некоторого свойства кода это
1) Метрика кода ПО
2) Количественные метрики
3) Метрики сложности потока управления программы
4) Метрики сложности потока данных

45. Класс метрик, который позволяет оценить объем кода, соотношение компонент кода и, в результате, сложность понимания кода, это
1) Количественные метрики
2) Метрика кода ПО
3) Метрики сложности потока управления программы
4) Метрики сложности потока данных

46. Класс метрик, который оценивает управляющий граф программы, это
1) Метрики сложности потока управления программы
2) Метрика кода ПО
3) Количественные метрики
4) Метрики сложности потока данных

47. Класс метрик, который оценивает конфигурацию, размещение и использование данных в программе, это
1) Метрики сложности потока данных
2) Метрика кода ПО
3) Количественные метрики
4) Метрики сложности потока управления программы

48. Класс метрик, который устанавливает сложность структуры программы как на основе количественных подсчетов, так и на основе анализа управляющих структур, это
1) Метрики сложности потока управления и данных программы
2) Метрики сложности потока управления программы
3) Метрики сложности потока данных
4) Объектно-ориентированные метрики

49. Класс метрик, который специализирован на оценке использования методов объектно-ориентированного программирования, это

1) Объектно-ориентированные метрики
2) Метрики сложности потока данных
3) Метрики сложности потока управления и данных программы
4) Метрики надежности

50. Метрики, которые оценивают уровень дефектов кода, это
1) Метрики надежности
2) Метрики сложности потока управления и данных программы
3) Объектно-ориентированные метрики
4) Гибридные метрики

Проверяемая компетенция – ПК1.5.

1. Метрики класса, которые основываются на более простых метриках и представляют собой их взвешенную сумму, это
1) Гибридные метрики
2) Метрики сложности потока управления и данных программы
3) Объектно-ориентированные метрики
4) Метрики надежности

2. Тестирование, которое заключается в том, что каждый модуль тестируется отдельно, это
1) Монолитное тестирование
2) Пошаговое тестирование

3. Тестирование, при котором модули тестируются не изолированно, а подключаются поочередно к набору уже оттестированных модулей, это
1) Пошаговое тестирование
2) Монолитное тестирование

4. Ошибка характеризуется критической важностью, если
1) произошел сбой либо отказ системы, и дальнейшая работа с программой невозможна
2) не работает основная функциональность программы (например, добавление записи в базу данных)
3) не работает второстепенная функциональность либо имеются недочеты в работе основной функциональности
4) имеется мелкая ошибка (например, ошибка интерфейса либо орфографическая ошибка)

5. Ошибка характеризуется серьезной важностью, если
1) не работает основная функциональность программы (например, добавление записи в базу данных)
2) произошел сбой либо отказ системы, и дальнейшая работа с программой невозможна
3) не работает второстепенная функциональность либо имеются недочеты в работе основной функциональности
4) имеется мелкая ошибка (например, ошибка интерфейса либо орфографическая ошибка)

6. Ошибка характеризуется средней важностью, если
1) не работает второстепенная функциональность либо имеются недочеты в работе основной функциональности
2) произошел сбой либо отказ системы, и дальнейшая работа с программой невозможна
3) не работает основная функциональность программы (например, добавление записи в базу данных)

4) имеется мелкая ошибка (например, ошибка интерфейса либо орфографическая ошибка)

7. Ошибка характеризуется низкой важностью, если
   1) имеется мелкая ошибка (например, ошибка интерфейса либо орфографическая ошибка)
   2) произошел сбой либо отказ системы, и дальнейшая работа с программой невозможна
   3) не работает основная функциональность программы (например, добавление записи в базу данных)
   4) не работает второстепенная функциональность либо имеются недочеты в работе основной функциональности
8. Ошибка обладает очень высоким приоритетом, если
   1) ошибка должна быть исправлена немедленно
   2) ошибка может быть исправлена в последнюю очередь
   3) ошибка может быть исправлена, когда появится свободное время
   4) ошибка должна быть исправлена как можно скорее
9. Ошибка обладает высоким приоритетом, если
   1) ошибка должна быть исправлена как можно скорее;
   2) ошибка может быть исправлена в последнюю очередь
   3) ошибка может быть исправлена, когда появится свободное время
   4) ошибка должна быть исправлена немедленно
10. Ошибка обладает средним приоритетом, если
   1) ошибка может быть исправлена, когда появится свободное время;
   2) ошибка должна быть исправлена как можно скорее
   3) ошибка может быть исправлена в последнюю очередь
   4) ошибка должна быть исправлена немедленно
11. Ошибка обладает низким приоритетом, если
   1) ошибка может быть исправлена в последнюю очередь.
   2) ошибка может быть исправлена, когда появится свободное время
   3) ошибка должна быть исправлена как можно скорее
   4) ошибка должна быть исправлена немедленно
12. Свойство ошибки, которое связано с частотой и условиями воспроизведения, это
   1) Воспроизводимость
   2) Приоритет
   3) Симптом
   4) Важность
13. Свойство ошибки, которое связано со скоростью исправления ошибки, это
   1) Воспроизводимость
   2) Приоритет
   3) Симптом
   4) Важность
14. Метод отладки, при котором выполняется всесторонний анализ за столом исходного кода и алгоритма программы, выходных результатов и сообщений компилятора, выполнение тестируемой программы вручную, используя тестовый набор, при работе с которым была обнаружена ошибка это
   1) Метод ручного тестирования
   2) Метод индукции
   3) Метод дедукции
   4) Метод обратного прослеживания

15. Метод отладки, по которому фрагмент проявления ошибки вычисляют, исходя из последних полученных результатов и действий пользователя, это

1) Метод индукции
2) Метод ручного тестирования
3) Метод дедукции
4) Метод обратного прослеживания

16. Метод отладки, по которому формируется множество причин, которые могли бы вызвать данное проявление ошибки, а затем анализируются причины и исключаются те, которые противоречат имеющимся данным, это

1) Метод дедукции
2) Метод ручного тестирования
3) Метод обратного прослеживания
4) Метод индукции

17. Метод отладки, когда отладка начинается с точки программы, где обнаружен неверный ожидаемый результат, с целью определить место между оператором, где результат выполнения программы соответствовал ожидаемому, и оператором, в котором появились расхождения, это

1) Метод обратного прослеживания
2) Метод ручного тестирования
3) Метод дедукции
4) Метод индукции

18. Способ рефакторинга, при котором преобразуется фрагмент кода в метод, название которого поясняет его назначение, это

1) Извлечение метода
2) Введение поясняющей переменной
3) Замена временной переменной запросом
4) Расщепление временной переменной

19. Способ рефакторинга, при котором помещается тело метода в код, который его вызывает, и удаляется метод, это

1) Встраивание метода
2) Извлечение метода
3) Введение поясняющей переменной
4) Замена временной переменной запросом

20. Способ рефакторинга, при котором выполняется замена всех ссылок на переменную выражением, это

1) Встраивание временной переменной
2) Введение поясняющей переменной
3) Извлечение метода
4) Замена временной переменной запросом

21. Способ рефакторинга, при котором выполняется замена всех ссылокна временную переменную новым методом, который после этого может быть использован и в других методах, это

1) Замена временной переменной запросом
2) Введение поясняющей переменной
3) Извлечение метода
4) Расщепление временной переменной

22. Способ рефакторинга, при котором результат выражения или его части помещается во временную переменную, имя которой поясняет его назначение, это

1) Введение поясняющей переменной
2) Извлечение метода
3) Замена временной переменной запросом
4) Расщепление временной переменной

23. Способ рефакторинга, при котором для каждого присваивания создается отдельная временная переменная, это

1) Расщепление временной переменной
2) Введение поясняющей переменной
3) Извлечение метода
4) Удаление присваиваний параметрам

24. Способ рефакторинга, при котором в случае, если код выполняет присваивание параметру, то вместо этого используется временная переменная, это

1) Удаление присваиваний параметрам
2) Замена подкласса полями
3) Введение поясняющей переменной
4) Расщепление временной переменной

25. Способ рефакторинга, при котором преобразовывается метод в отдельный объект так, чтобы локальные переменные стали его полями, после этого метод раскладывается на несколько методов того же объекта, это

1) Замена метода объектом методов
2) Замена подкласса полями
3) Введение поясняющей переменной
4) Расщепление временной переменной

26. Способ рефакторинга, при котором заменяется тело метода новым алгоритмом, это

1) Замена алгоритма
2) Извлечение метода
3) Расщепление временной переменной
4) Удаление присваиваний параметрам

27. Способ рефакторинга, при котором создается новый метод с аналогичным телом в том классе, который чаще всего им используется(тело прежнего метода заменяется простым делегированием или полностью удаляется), это

1) Перенос метода
2) Замена подкласса полями
3) Замена временной переменной запросом
4) Удаление присваиваний параметрам

28. Способ рефакторинга, при котором создается в целевом классе новое поле и изменяется код всех его использований, так поле используется (или будет использоваться) классом чаще, чем классом, в котором оно определено, это

1) Перенос поля
2) Замена подкласса полями
3) Извлечение метода
4) Замена временной переменной запросом

29. Способ рефакторинга, при котором создается новый класс и перемещаются в него соответствующие поля и методы из старого класса, это

1) Извлечение класса
2) Замена подкласса полями
3) Введение поясняющей переменной
4) Замена временной переменной запросом

30. Способ рефакторинга, при котором перемещается функциональность класса в другой класс и удаляется исходный, это

1) Встраивание класса
2) Замена подкласса полями
3) Извлечение метода
4) Расщепление временной переменной

31. Способ рефакторинга, при котором создаются на сервере методы, скрывающие делегирование, это

1) Сокрытие делегирования
2) Введение поясняющей переменной
3) Извлечение метода
4) Удаление присваиваний параметрам

32. Способ рефакторинга, при котором клиент обращается непосредственно к делегату, это

1) Удаление посредника
2) Замена подкласса полями
3) Замена временной переменной запросом

33. Способ рефакторинга, при котором создается в классе клиента метод, которому в качестве первого аргумента передается экземпляр класса сервера, это

1) Введение внешнего метода
2) Введение поясняющей переменной
3) Извлечение метода
4) Удаление посредника

34. Способ рефакторинга, который используется в случае, если используемому классу сервера требуется несколько дополнительных методов, но класс недоступен для модификации, поэтому создается новый класс с необходимыми дополнительными методами, это

1) Введение локальноrо расширения
2) Введение поясняющей переменной
3) Извлечение метода
4) Удаление посредника

35. Способ рефакторинга, при котором создаются методы получения и установки значения поля и обращение к полю выполняется только через них, это

1) Самоинкапсуляция поля
2) Введение поясняющей переменной
3) Извлечение метода
4) Введение локальноrо расширения

36. Способ рефакторинга, при котором преобразуется элемент данных в объект, это

1) Замена значения данных объектом
2) Введение поясняющей переменной
3) Введение локальноrо расширения
4) Самоинкапсуляция поля

37. Способ рефакторинга, при котором объект превращается в объект-ссылку, это

1) Замена значения ссылкой
2) Замена подкласса полями
3) Самоинкапсуляция поля
4) Замена значения данных объектом

38. Способ рефакторинга, при котором объект-ссылка превращается в объект-значение, это

1) Замена ссылки значением
2) Замена подкласса полями
3) Извлечение метода
4) Удаление посредника

39. Способ рефакторинга, при котором массив заменяется объектом, в котором есть поле для каждого элемента, это

1) Замена массива объектом

2) Введение поясняющей переменной
3) Удаление посредника
4) Введение локального расширения

40. Способ рефакторинга, при котором копируются данные в объект предметной области,создается объект-наблюдатель, который будет синхронизировать эти два набора данных, это

1) Дублирование видимых данных
2) Замена подкласса полями
3) Удаление посредника
4) Самоинкапсуляция поля

41. Способ рефакторинга, при котором добавляются обратные указатели и изменяются модификаторы так, чтобы два класса, взаимно использующие функциональность один другого, но между ними есть связь только в одном направлении, обновляли оба множества, это

1) Замена однонаправленной связи двунаправленной
2) Введение поясняющей переменной
3) Введение локального расширения
4) Замена значения данных объектом

42. Способ рефакторинга, при котором для двунаправленной связи удаляется ненужный конец связи, это

1) Замена двунаправленной связи однонаправленной
2) Замена подкласса полями
3) Введение поясняющей переменной
4) Самоинкапсуляция поля

43. Способ рефакторинга, при котором создается константа, присваивается ей имя в соответствии со смыслом константы и заменяется ею число, это

1) Замена магического числа символической константой
2) Замена подкласса полями
3) Извлечение метода
4) Замена значения данных объектом

44. Способ рефакторинга, при котором открытое поле превращаетсяв закрытое, это

1) Инкапсуляция поля
2) Замена подкласса полями
3) Замена кода типа состоянием/стратегией
4) Замена кода типа подклассами

45. Способ рефакторинга, при котором возвращаемая коллекция становится доступной только для чтения, это

1) Инкапсуляция коллекции
2) Введение поясняющей переменной
3) Самоинкапсуляция поля
4) Замена однонаправленной связи двунаправленной

46. Способ рефакторинга, который используется, когда необходим интерфейс для структуры записи в традиционной программной среде, это

1) Замена записи классом данных
2) Замена однонаправленной связи двунаправленной
3) Инкапсуляция коллекции
4) Инкапсуляция поля

47. Способ рефакторинга, который используется, когда у класса есть код числового типа, который не влияет на его поведение, это

1) Замена кода типа классом
2) Инкапсуляция коллекции

3) Инкапсуляция поля
4) Замена однонаправленной связи двунаправленной

48. Способ рефакторинга, который используется, когда имеется неизменяемый код типа, влияющий на поведение класса, это

1) Замена кода типа подклассами
2) Замена подкласса полями
3) Инкапсуляция коллекции
4) Инкапсуляция поля

49. Способ рефакторинга, который используется, когда имеется код типа, влияющий на поведение класса, но применить создание подклассов невозможно, это

1) Замена кода типа состоянием/стратегией
2) Замена подкласса полями
3) Замена значения данных объектом
4) Замена однонаправленной связи двунаправленной

50. Способ рефакторинга, который используется, когда имеются подклассы, различающиеся только методами, возвращающими константные данные, это

1) Замена подкласса полями
2) Самоинкапсуляция поля
3) Замена значения данных объектом
4) Замена однонаправленной связи двунаправленной

Второй блок заданий
Формируемые ПК1.3, ПК1.4, ПК 1.5

Проверяемая компетенция ПК1.3:

1. Какой способ рефакторинга используется, когда имеется сложная условная цепочка проверок (if-then-else)?
2. Какой способ рефакторинга используется, когда имеется ряд проверок условий, дающих одинаковый результат?
3. Какой способ рефакторинга используется, когда один и тот же фрагмент кода присутствует во всех ветвях условного выражения?
4. Какой способ рефакторинга используется, когда имеется переменная, действующая в качестве управляющего флага для ряда логических выражений?
5. Какой способ рефакторинга используется, когда метод использует условное поведение, которое не дает возможности понять нормальный путь выполнения?
6. Какой способ рефакторинга используется, когда имеется условная инструкция, обеспечивающая разное поведение в зависимости от типа объекта?
7. Какой способ рефакторинга используется, когда в исходном тексте имеются многократные проверки на нулевое значение?
8. Какой способ рефакторинга используется, когда некоторая часть кода предполагает определенное состояние программы?
9. Какой способ рефакторинга используется, когда имя метода не раскрывает его назначения?
10. Какой способ рефакторинга используется, когда метод нуждается в дополнительной информации от вызывающего метода?
11. Какой способ рефакторинга используется, когда параметр больше не используется телом метода?
12. Какой способ рефакторинга используется, когда имеется метод, не только возвращающий значение, но еще и изменяющий состояние объекта?
13. Какой способ рефакторинга используется, когда несколько методов выполняют сходные действия, но с разными значениями, содержащимися в теле метода?
14. Какой способ рефакторинга используется, когда имеется метод, выполняющий разный код в зависимости от значения параметра перечислимого типа?
15. Какой способ рефакторинга используется, когда от объекта получено несколько значений, которые затем передаются при вызове метода в качестве параметров?
16. Какой способ рефакторинга используется, когда объект вызывает метод, а затем передает полученный результат в качестве параметра метода?
17. Какой способ рефакторинга используется, когда есть группа параметров, естественным образом связанных между собой?
18. Какой способ рефакторинга используется, когда значение поля должно быть установлено в момент создания объекта и в дальнейшем не изменяться?
19. Какой способ рефакторинга используется, когда метод не используется никаким другим классом?
20. Какой способ рефакторинга используется, когда при создании объекта требуетсявыполнить нечто большее, чем простое конструирование?
21. Какой способ рефакторинга используется, когда метод возвращает объект, к которому вызывающий код должен применить нисходящее приведение типа?
22. Какой способ рефакторинга используется, когда метод возвращает код, указывающий происшедшую ошибку?
23. Какой способ рефакторинга используется, когда выполняется генерация исключения при выполнении условия, которое может быть проверено вызывающим кодом?

24. Какой способ рефакторинга используется, когда в подклассах есть методы с идентичными результатами работы?
25. Какой способ рефакторинга используется, когда имеются конструкторы подклассов с практически идентичными телами?
26. Какой способ рефакторинга используется, когда в суперклассе имеется поведение, относящееся только к некоторым из его подклассов?
27. Какой способ рефакторинга используется, когда имеется поле, используемое лишь некоторыми подклассами?
28. Какой способ рефакторинга используется, когда в классе есть возможности, используемые только в некоторых экземплярах?
29. Какой способ рефакторинга используется, когда имеются два класса с аналогичной функциональностью?
30. Какой способ рефакторинга используется, когда несколько клиентов используют одно и то же подмножество интерфейса класса, или в двух классах часть интерфейса является общей?
31. Какой способ рефакторинга используется, когда суперкласс и подкласс не слишком различаются?
32. Какой способ рефакторинга используется, когда в подклассах есть два метода, выполняющие аналогичные шаги в одинаковом порядке, однако сами шаги различны?
33. Какой способ рефакторинга используется, когда подкласс использует только часть интерфейса суперкласса или не желает наследовать данные?
34. Какой способ рефакторинга используется, когда используется делегирование и часто пишется много простых делегирующих методов для всего интерфейса?
35. Какой способ рефакторинга используется, когда имеется иерархия наследования, которая решает одновременно две задачи?
36. Какой способ рефакторинга используется, когда имеется код, написанный в процедурном стиле?
37. Какой способ рефакторинга используется, когда имеются классы графического интерфейса пользователя, содержащие логику предметной области?
38. Какой способ рефакторинга используется, когда имеется класс, выполняющий слишком большую работу, как минимум частично реализованную с помощью многочисленных условных инструкций?
39. Что такое тестовый случай?
40. Что такое Тестовые данные?
41. Что такое метод тестирования?
42. Что такое план тестирования?
43. Что такое отчет о тестировании?
44. Что такое тестируемость?
45. Что такое тестовое покрытие?
46. Что такое отладка?
47. Какие ошибки являются синтаксическими?
48. Какие ошибки являются алгоритмическими?
49. Какие ошибки являются структурными?
50. Какие ошибки являются концептуальными?

Проверяемая компетенция ПК1.4:

1. Как выполняется отладка по методу ручного тестирования?
2. Укажите особенности метода ручного тестирования при отладке.
3. На чем основан метод индукции при отладке?

4. Как локализуют ошибку по методу индукции при отладке?
5. Что необходимо сделать, чтобы выдвинуть гипотезу по методу индукции при отладке?
6. Что выполняется по методу индукции при отладке в случае подтверждения или опровержения гипотезы?
7. Что необходимо выполнить по методу дедукции при отладке?
8. Как подтверждается истинность гипотезы по методу дедукции при отладке?
9. В каких случаях эффективен метод обратного отслеживания при отладке?
10. Как выполняется метод обратного прослеживания при отладке?
11. Что такое тестирование?
12. Что такое объект тестирования?
13. Что такое условия тестирования?
14. Что такое вид тестирования?
15. Что такое модель для тестирования?
16. Что такое метод тестирования?
17. Что такое методика тестирования?
18. Что такое средство тестирования?
19. Что такое объём тестирования?
20. Что такое данные тестирования?
21. Что такое результат тестирования?
22. Что такое протокол тестирования?
23. Какое тестирование относится к исследовательскомутестированию?
24. Какое тестирование относится к сравнительномутестированию?
25. Какое тестирование относится к определительномутестированию?
26. Какое тестирование относится к контрольномутестированию?
27. Какое тестирование относится к нормальномутестированию?
28. Какое тестирование относится к ускоренномутестированию?
29. Какое тестирование относится к сокращенномутестированию?
30. Какое тестирование относится к полномутестированию?
31. Какое тестирование относится к ручномутестированию?
32. Какое тестирование относится к автоматизированномутестированию?
33. Какое тестирование относится к автоматическомутестированию?
34. Какое тестирование относится к модульномутестированию?
35. Какое тестирование относится к интеграционномутестированию?
36. Какое тестирование относится к системномутестированию?
37. По каким типам разделяют тестирование программных продуктов?
38. Из каких стадий состоит процесс разработки ПО?
39. Что выполняется при общем планировании и анализе требований при тестировании ПО?
40. Что выполняется при уточнении критериев приёмки в процессе тестирования?
41. Что выполняется в процессе уточнения стратегии тестирования в процессе тестирования?
42. Что выполняется на стадии разработки тест-кейсов в процессе тестирования?
43. Какие выделяют типы метрик кода?
44. Преднамеренное прерывание, при котором выполняется вызов отладчика, это
45. В техническом задании наименование, краткая характеристика области применения программы или программного изделия и объекта, в котором используют программу или программное изделие, указываются в разделе

46. В техническом задании документы, на основании которых ведется разработка, организация, утвердившая этот документ, и дата его утверждения, наименование или условное обозначение темы разработки указываются в разделе

47. В техническом задании функциональное и эксплуатационное назначение программы или программного изделия указываются в разделе

48. В техническом задании требования к составу выполняемых функций, организации входных и выходных данных, временным характеристикам указываются в подразделе

49. В техническом задании требования к обеспечению надежного функционирования, устойчивого функционирования, контроль входной и выходной информации, время восстановления после отказа указываются в подразделе

50. В техническом задании условия эксплуатации, при которых должны обеспечиваться заданные характеристики, а также вид обслуживания, необходимое количество и квалификация персонала указываются в подразделе

Проверяемая компетенция ПК 1.5:

1. В техническом задании необходимый состав технических средств с указанием их основных технических характеристик указываются в подразделе

2. В техническом задании требования к информационным структурам на входе и выходе и методам решения, исходным кодам, языкам программирования и программным средствам, используемым программой указываются в подразделе

3. В техническом задании предварительный состав программной документации и, при необходимости, специальные требования к ней указываются в разделе

4. В техническом задании ориентировочная экономическая эффективность, предполагаемая годовая потребность, экономические преимущества разработки по сравнению с лучшими отечественными и зарубежными образцами или аналогами указываются в разделе

5. В техническом задании необходимые стадии разработки, этапы и содержание работ (перечень программных документов, которые должны быть разработаны, согласованы и утверждены), а также сроки разработки и исполнители указываются в разделе

6. В техническом задании виды испытаний и общие требования к приемке работы указываются в разделе

7. В каком документе указываются сведения для обеспечения процедуры общения оператора с вычислительной системой в процессе выполнения программы?

8. В каком документе указываются сведения для эксплуатации программы?

9. В каком документе указываются сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения?

10. В каком документе указываются сведения о назначении программы, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств?

11. В каком документе указывается перечень эксплуатационных документов на программу?

12. В каком документе указывается схема алгоритма, общее описание алгоритма и (или) функционирования программы, а также обоснование принятых технических и технико-экономических решений?

13. В каком документе указываются сведения о логической структуре и функционировании программы?

14. В каком документе указывается запись программы с необходимыми комментариями?

15. Требование, для выполнения которого требуется, чтобы спецификации должны однозначно воспринимались как заказчиком, так и разработчиком, называется

16. Требование, для выполнения которого требуется чтобы спецификации содержали всю существенную информацию и не содержали несущественной информации, чтобы не препятствовать разработчику в выборе наиболее эффективных решений, называется

Ответ: требование полноты

17. Спецификации, которые описывают функции разрабатываемого ПО, называются

18. Спецификации, которые определяют требования к техническим средствам, надежности, безопасности, называются

19. Схема, отражающая состав и взаимодействие по управлению частей разрабатываемого ПО, называется

20. В каком разделе руководства оператора указываются сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации?

21. В каком разделе руководства оператора указываются условия, необходимые для выполнения программы?

22. В каком разделе руководства оператора указывается последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузки и управляет выполнением программы, а также ответы программы на эти команды?

23. В каком разделе руководства оператора указываются тексты сообщений, выдаваемых в ходе выполнения программы, описание их содержания и соответствующие действия оператора?

24. В каком разделе руководства программиста указываются назначение и функции, выполняемые программой, условия, необходимые для выполнения программы?

25. В каком разделе руководства программиста указывается описание основных характеристик и особенностей программы?

26. В каком разделе руководства программиста указывается описание процедур вызова программы?

27. В каком разделе руководства программиста указывается описание организации используемой входной и выходной информации и, при необходимости, ее кодирования?

28. В каком разделе руководства программиста указываются тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям?

29. В каком разделе описания программы указывается обозначение и наименование программы; программное обеспечение, необходимое для функционирования программы; языки программирования, на которых написана программа?

30. В каком разделе описания программы указываются классы решаемых задач и назначение программы и сведения о функциональных ограничениях на применение?

31. В каком разделе описания программы указывается алгоритм программы; используемые методы; структура программы с описанием функций составных частей и связи между ними; связи программы с другими программами. Описание логической структуры программы выполняют с учетом текста программы на исходном языке?

32. В каком разделе описания программы указываются типы электронно-вычислительных машин и устройств, которые используются при работе программы?

33. В каком разделе описания программы указывается способ вызова программы с соответствующего носителя данных; входные точки в программу.

Допускается указывать адреса загрузки, сведения об использовании оперативной памяти, объем программы?

34. В каком разделе описания программы указывается характер, организация и предварительная подготовка входных данных; формат, описание и способ кодирования входных данных?

35. В каком разделе описания программы указывается характер и организация выходных данных; формат, описание и способ кодирования выходных данных?

36. В каком разделе программы испытаний указывается наименование, область применения и обозначение испытуемой программы?

37. В каком разделе программы испытаний указывается цель проведения испытаний?

38. В каком разделе программы испытаний указываются требования, подлежащие проверке во время испытаний и заданные в техническом задании на программу?

39. В каком разделе программы испытаний указывается состав программной документации, предъявляемой на испытания, а также специальные требования, если они заданы в техническом задании на программу?

40. В каком разделе программы испытаний указываются технические и программные средства, используемые во время испытаний, а также порядок проведения испытаний?

41. В каком разделе программы испытаний указываются описания используемых методов испытаний?

42. Из каких стадий состоит разработка программной документации в соответствии с ГОСТ 19.102-77?

43. Из каких этапов состоит создание технического задания в соответствии с ГОСТ 19.102-77?

44. Из каких этапов состоит создание эскизного проекта в соответствии с ГОСТ 19.102-77?

45. Из каких этапов состоит создание технического проекта в соответствии с ГОСТ 19.102-77?

46. Из каких этапов состоит создание рабочего проекта в соответствии с ГОСТ 19.102-77?

47. Из каких этапов состоит утверждение эскизного проекта при создании технического проекта в соответствии с ГОСТ 19.102-77?

48. Из каких этапов состоит разработка эскизного проекта при создании эскизного проекта в соответствии с ГОСТ 19.102-77?

49. Из каких этапов состоит утверждение технического проекта при создании технического проекта в соответствии с ГОСТ 19.102-77?

50. Из каких этапов состоят испытания программы при создании рабочего проекта в соответствии с ГОСТ 19.102-77?

Составил преподаватель Веремьев В.О.