**Оценочные материалы текущего контроля знаний и промежуточной аттестации по МДК 02.02 Инструментальные средства разработки программного обеспечения**

**Компетенция ПК 2.2**

|  |
| --- |
| 1. **Какие технологии разработки программ используются в современном программировании:** 2. **Визуальные** 3. **Событийные** 4. Структурные 5. **Объектно-ориентированные** 6. Модульные 7. Текстуальные 8. Графические 9. Машинно-ориентированное |
| 1. **Инструментальные средства разработки программ – это …** 2. **Средства создания новых программ** 3. Сервисные средства разработки ПО 4. Аналитические средства разработки ПО 5. **Программное обеспечение, предназначенное для разработки и отладки новых программ** 6. Средства отладки ПО 7. Средства тестирования ПО 8. **Аппаратные и программные инструменты разработки нового ПО**   Технические инструментальные средства разработки ПО |
| **3. Укажите правильную последовательность создания программного обеспечения:**   1. Анализ требований, проектирование, программирование, тестирование 2. Формулирование задачи, анализ требований, программирование, проектирование, тестирование 3. Анализ требований, программирование, проектирование, отладка, тестирование 4. **Формулирование задачи, анализ требований, проектирование, программирование, тестирование** |
| **4. В чем сущность модульного программирования:**   1. **в разбиении программы на отдельные функционально независимые части** 2. в разбиении программы на отдельные равные части 3. в разбиение программы на процедуры и функции |
| **5. Недостаток модульного программирования:**   1. увеличивает трудоемкость программирования; 2. снижает быстродействие программы; 3. не позволяет выполнять оптимизацию программы. 4. **усложняет процедуру комплексного тестирования** |
| **6. При структурном программировании задача выполняется:**   1. **поэтапным разбиением на более легкие задачи;** 2. без участия программиста; 3. объединением отдельных модулей программы. |
| **7. Модульное программирование применимо при:**   1. **проектировании сверху-вниз;** 2. проектирование снизу-вверх; |
| **8. Проектирование сверху-вниз это:**   1. **последовательное разбиение общих задач на более мелкие;** 2. составление из отдельных модулей большой программы. |
| **9. Проектирование снизу-вверх это:**   1. **составление из отдельных модулей большой программы;** 2. последовательное разбиение общих задач на более мелкие. |
| **10. Достоинство модульного программирования:**   1. не требует комплексного тестирования 2. уменьшает размер программы 3. **возможность приступить к тестированию до завершения написания всей программы** 4. повышает надежность программы |
| **11. Рекомендуемые размеры модулей:**   * 1. большие;   2. равные;   3. **небольшие;**   4. фиксированной длины. |
| **12. В чем заключается независимость модуля:**   1. **в написании, отладке и тестировании независимо от остальных модулей;** 2. в разработке и написании независимо от других модулей; 3. в независимости от работы основной программы. |
| **13. Основные характеристики модуля**   1. **Размер** 2. **Прочность** 3. **Сцепление** 4. **Рутинность** 5. Связность 6. Безопасность 7. Определенность |
| 1. **К основным понятиям объектно-ориентированного подхода относятся:** 2. Обобщение 3. **Полиморфизм** 4. **Инкапсуляция** 5. Реализация 6. Агрегирование 7. **Наследование** 8. Ассоциация 9. Композиция |
| **15. Какие методы сборки программы существуют**   1. монолитная 2. **пошаговая** 3. **одновременная** 4. постепенная |
| **16. Программное средство, обеспечивающее автоматизированную поддержку деятельности, выполняемой в рамках технологических операций.**   1. **Инструментальное средство (CASE-средство)** 2. Операционная система 3. Текстовый редактор 4. Язык программирования |
| **17. Что такое управление версиями:**   1. автоматизированный процесс трансформации исходных текстов ПО в пакет исполняемых модулей 2. **управление версиями файлов** 3. ручной процесс трансформации исходных текстов ПО в пакет исполняемых модулей |
| **18. Требования к информационной системе сформулированы в**   1. описании предметной области 2. техническом проекте 3. **техническом задании** 4. продукте моделирования |
| **19. Целью разработки архитектуры будущего программного обеспечения является**   1. разработка устройств основных компонент программного обеспечения. 2. разработка программного кода 3. тестирование 4. **разработка модели (описания) будущей системы, понятной для кодировщика** |
| **20. Требования к информационной системе реализованы в**   1. описании предметной области 2. **техническом проекте** 3. техническом задании 4. продукте моделирования |
| **21. Функциональные характеристики применения, являющиеся исходными данными для оценки и выбора технических средств для разработки программного обеспечения**   1. **среда функционирования** 2. удобство использования 3. совместимость с другими ТС ПО 4. соответствие технологическим стандартам |
| **22. К числу основных возможностей, обеспечиваемых современными инструментальными средствами, относятся:**   1. **графический анализ и проектирование;** 2. **интерактивное прототипирование;** 3. автоматическое тестирование и верификация программного обеспечения; 4. разработка руководства пользователей. |
| **23.** **Достоинство модульного программирования:**   1. не требует компоновки 2. **создание программы по частям в произвольном порядке** 3. всегда дает эффективные программы 4. снижает количество ошибок |
| **24. Что не входит в категорию «Инструментальные средства разработки программного обеспечения»:**   1. средства разработки приложений клиент-сервер; 2. средства разработки Internet/Intranet-приложений; 3. **СУБД (системы управления базами данных);** 4. средства автоматизации проектирования программного обеспечения. |
| **25. Что не относится к системному программному обеспечению ЭВМ включает:**   1. операционные системы; 2. **системы автоматизированного проектирования;** 3. утилиты; 4. драйвера; 5. операционные оболочки. |
| **26. Программное обеспечение ЭВМ можно классифицировать на следующие группы:**   1. **системное программное обеспечение;** 2. **прикладное программное обеспечение;** 3. операционные системы; 4. пакеты прикладных программ общего назначения; 5. **инструментальные средства разработки программного обеспечения.** |
| **27. Прикладное программное обеспечение включает:**   1. **системы автоматизированного проектирования;** 2. утилиты; 3. драйвера; 4. **текстовые редакторы;** 5. **программы автоматизации бухгалтерского учёта.** |
| **28. Модуль – это отдельная, функционально законченная программная единица, которая**   1. может быть частью программы, но не может применяться самостоятельно 2. не может применяться самостоятельно, либо быть частью программы 3. может применяться самостоятельно, но не может быть частью программы 4. **может применяться самостоятельно, либо быть частью программы** |
| **29. Стиль программирования - это**   1. собрание правил, которых обязан придерживаться каждый программист при написании приложений 2. четко определенный кодекс «статей» программирования на наиболее распространенных языках (C++, Java и т.п.) 3. **набор приемов и методов программирования, позволяющих создавать корректные, эффективные и доступные для чтения и понимания программы** 4. выбор определенного языка программирования разработчиком при постановке перед ним определенной задачи |

|  |
| --- |
| 1. **Интеграция программных продуктов с другими программами называется** 2. **коммуникативностью** 3. мобильностью 4. эффективностью 5. модифицируемостью |
| 1. **Программное обеспечение включает:** 2. Исходные тексты программ, исполнимый код 3. Техническое задание, исходные тексты программ 4. **Исполнимый код, программную документацию** |
| 1. **Репозиторий-портал** 2. **совмещает в себе функции каталога ПО, инструмента для организации сообществ по разработке ПО, среды информационного обмена для разработчиков, технической инфраструктуры разработки проектов по созданию ПО и инструмента для загрузки разработанного ПО.** 3. направлен на конечных пользователей ПО, который может выступать как хранилищем файлов, так и хранилищем ссылок на другие сайты, где можно загрузить соответствующие файлы. 4. служит для интеграции разных пакетов программ в единую систему. |
| **33. Репозиторий-каталог**   1. совмещает в себе функции каталога ПО, инструмента для организации сообществ по разработке ПО, среды информационного обмена для разработчиков, технической инфраструктуры разработки проектов по созданию ПО и инструмента для загрузки разработанного ПО. 2. **направлен на конечных пользователей ПО, который может выступать как хранилищем файлов, так и хранилищем ссылок на другие сайты, где можно загрузить соответствующие файлы.** 3. служит для интеграции разных пакетов программ в единую систему. |
| **34. Что делает команда git add?**   1. Создает файл с указанным именем и сразу добавляет его в Git 2. Добавляет локальный файл в удаленный репозиторий так, чтобы другие участники проекта могли его видеть 3. **Начинает отслеживать указанный файл или файлы** |
| **35. Что делает команда git status?**   1. **Показывает состояние проекта** 2. Показывает имя и email нашего пользователя 3. Показывает место, занимаемое репозиторием на жестком диске и количество выделенного под репозиторий месте 4. Такой команды нет, есть только команда git show |
| **36. Что такое Git Hub?**   1. Программа для работы с Git 2. Драйвер для Git 3. **Веб-сервис для хостинга IT-проектов и их совместной разработки, основанный на Git** 4. UI для работы с локальной версией Git |
| **37. Что такое ветка в репозитории Git?**   1. Это то же самое, что и коммит 2. Это минимум два коммита с одинаковым коммит-сообщением 3. **Это разные пути развития проекта, по сути разные последовательности коммитов** 4. Это механизм изменения конкретного файла |
| **38. Что такое коммит?**   1. **Это единица состояния проекта в Git** 2. Это результат вывода команды git diff 3. Это обобщающее название одного из статусов файла в выводе git status: untracked, new, deleted или modified 4. Это слово ничего не означает, его ввели только для того, чтобы путать новичков |
| **39. Что такое репозиторий Git?**   1. Любая директория/папка в моей ОС 2. Любая папка, находящаяся внутри Git 3. **Репозиторий Git представляет собой каталог файловой системы, в котором находятся файлы конфигурации репозитория, файлы журналов, хранящие операции, выполняемые над репозиторием, индекс, описывающий расположение файлов, и хранилище, содержащее собственно файлы** 4. Папка .git/ и все входящие в нее |
| **40. Головной модуль программной системы**   1. **управляет запуском программного продукта (существует в единственном числе);** 2. обеспечивает вызов других модулей на обработку; 3. выполняют функции обработки; 4. осуществляют обслуживающие функции. |
| **41. Управляющий модуль программной системы**   1. управляет запуском программного продукта (существует в единственном числе); 2. **обеспечивает вызов других модулей на обработку;** 3. выполняют функции обработки; 4. осуществляют обслуживающие функции. |
| **42. Протокол UDP**   1. **протокол не требующий установки соединения, не гарантирует доставку своих сообщений** 2. протокол требующий установки соединения, обеспечивает надежную транспортировку данных |
| **43. Системы контроля версий делятся на:**   1. **локальные** 2. **централизованные** 3. **распределенные** 4. примитивные |
| **44. В какой ситуации надо делать git status?**   1. Чем чаще, тем лучше 2. [Всегда при создании коммита](https://topuch.com/moskovskij-priborostroitelenij-tehnikum-v5/index.html) 3. Всегда после команды git pull 4. **Только если надо узнать, в каком статусе находится репозиторий, а так эта команда не является обязательной для любой манипуляции** |
| **45. Протокол TCP**   1. протокол не требующий установки соединения, не гарантирует доставку своих сообщений 2. **протокол требующий установки соединения, обеспечивает надежную транспортировку данных** |
| **46. Какие задачи выполняют уровни OSI в процессе передачи данных по сети:**   1. уровни выполняют одинаковые задачи, постоянно повторяя передающие сигналы по сети 2. **каждый уровень выполняет свою определенную задачу** 3. первых три уровня выполняют одинаковые задачи, последующие выполняют определенные задачи |
| **47. Протокол управления передачей данных – это**   1. IP 2. **TCP** 3. FTP 4. SMTP |
| **48. Протокол – это**   1. Специальная программа, передающая данные по сети. 2. **Набор соглашений, регулирующих передачу данных по сети.** 3. Специальное устройство компьютера, управляющее передачей данных. 4. Специальная область жесткого диска, через которую производится передача данных. |
| **49. Протокол TCP/IPотносится**   1. К аппаратному уровню сети Internet. 2. **К системному (сетевому или транспортному) уровню сети Internet.** 3. К сеансовому уровню сети Internet. 4. К прикладному уровню сети Internet. |
| **50. Процедуру условного форматирования можно применять только:**   1. только к числовым значениям 2. только к текстовым данным 3. только к ячейкам, содержащим формулу либо функцию 4. **ко всем перечисленным значениям** |

**Компетенция ПК 2.3**

|  |
| --- |
| **1. Отладка ПС - это …**   1. **деятельность, направленная на обнаружение и исправление ошибок в ПС с использованием процессов выполнения его программ** 2. процесс выполнения программы на некотором наборе данных, для которого заранее известен результат применения или известны правила поведения этой программы 3. построение программы как композиции из нескольких типов управляющих конструкций, которые позволяют повысить понимаемость логики работы программы |
| 1. **Отладчик:** 2. Программа для создания системы защиты файла 3. Программа создания системы защиты от вирусных атак 4. **Программа, помогающая анализировать поведение отлаживаемой программы, обеспечивая ее трассировку** 5. Операционная оболочка для создания и управления файловыми структурами 6. Системное программное обеспечение для настройки операционной системы 7. Программа создания и редактирования графических файлов 8. **Программа, позволяющая выполнять остановы в заданных точках, просмотреть текущие значения переменных и изменять их значения** |
| 1. **Процесс локализации и исправления ошибок, обнаруженных при тестировании программного обеспечения.** 2. **Отладка** 3. Локализация 4. Тестирование |
| **4. На каком этапе создания программы могут появиться синтаксические ошибки**   1. **Программирование** 2. Проектирование 3. Тестирование 4. Анализ требований |
| **5. Отладка программ это:**   1. **локализация и исправление ошибок** 2. алгоритмизация программирования 3. компиляция и компоновка |
| **6. Вид ошибки с неправильным написанием служебных слов (операторов)**   1. **синтаксическая** 2. семантическая 3. логическая 4. символьная |
| **7. Вид ошибки с неправильным использованием служебных слов (операторов)**   1. синтаксическая 2. **семантическая** 3. логическая 4. символьная |
| **8. Причины синтаксических ошибок:**   1. ошибки в исходных данных; 2. ошибки, допущенные на более ранних этапах; 3. **плохое знание языка программирования;** 4. неправильное применение процедуры тестирования. |
| **9. Комплекс программных средств, предназначенных для кодирования, тестирования и отладки программного обеспечения. Имеет интерфейс, удобный пользователю.**   1. **Система программирования** 2. **Компилятор** 3. **Синтаксический анализатор** 4. Средства автоматизации сборки |
| **10.  Отладчик:**   1. **Программа, облегчающая программисту выполнение отладки разрабатываемых им программ** 2. Программа для создания системы защиты файла 3. Программа создания системы защиты от вирусных атак 4. **Программа, помогающая анализировать поведение отлаживаемой программы, обеспечивая ее трассировку** 5. Операционная оболочка для создания и управления файловыми структурами 6. Системное программное обеспечение для настройки операционной системы 7. Программа создания и редактирования графических файлов 8. **Программа, позволяющая выполнять остановы в заданных точках, просмотреть текущие значения переменных и изменять их значения** |
| **11. Последовательность этапов программирования**   1. **компилирование, компоновка, отладка** 2. компоновка, отладка, компилирование 3. отладка, компоновка, компилирование 4. компилирование, отладка, компоновка |
| **12. Вставьте пропущенное слово.**  \_\_\_\_\_\_\_\_\_\_\_\_\_, приложения, выполняющие программу в заданном режиме (например, пошаговом) с целью поиска, обнаружения и локализации ошибок. Используются на этапе компиляции.  Ответ: отладчики |
| **13. Выберите базовые функции отладчика:**   1. ввод и модификация текст программы 2. **пошаговое выполнение программы (режим трассировки) с отображением результатов** 3. **остановка в заранее определенных точках** 4. перевод исходного текста на входном языке в язык машинных кодов 5. **возможность остановки в некотором месте программы при выполнении некоторого условия** 6. **изображение и изменение значений переменных** 7. объединяют несколько объектных модулей, порождаемых компилятором, в одну программ |
| **14. Логические ошибки**  1. Это ошибки, обнаруженные компилятором. Их можно подразделить на категории в зависимости от того, какие правила языка он нарушают  2. Это ошибки, обнаруженные в ходе контрольных проверок выполняемого модуля.  3. **Это ошибки, найденные программистом в поисках причины неправильных результатов.**  4. Это ошибки, обнаруженные редактором связей при попытке объединить объектные файлы в выполняемый модуль |
| **15. Инструментальные средства отладки (Не правильный ответ)**   1. отладчики 2. трассировка 3. **компиляторы** |
| **16. Когда программист может проследить последовательность выполнения команд программы:**   1. при тестировании 2. **при трассировке** 3. при компиляции 4. при выполнении программы 5. при компоновке |
| 1. **Синтаксические ошибки** 2. **ошибки, фиксируемые компилятором (транслятором, интерпретатором) при выполнении синтаксического и частично семантического анализа программы;** 3. ошибки, обнаруженные компоновщиком (редактором связей) при объединении модулей программы; 4. ошибки, обнаруженные операционной системой, аппаратными средствами или пользователем при выполнении программы. |
| **18. Ошибки компоновки -**   1. ошибки, фиксируемые компилятором (транслятором, интерпретатором) при выполнении синтаксического и частично семантического анализа программы; 2. **ошибки, обнаруженные компоновщиком (редактором связей) при объединении модулей программы;** 3. ошибки, обнаруженные операционной системой, аппаратными средствами или пользователем при выполнении программы. |
| **19. Ошибки выполнения**   1. ошибки, фиксируемые компилятором (транслятором, интерпретатором) при выполнении синтаксического и частично семантического анализа программы; 2. **ошибки, обнаруженные операционной системой, аппаратными средствами или пользователем при выполнении программы.** 3. ошибки, обнаруженные компоновщиком (редактором связей) при объединении модулей программы |
| 1. **Попытка найти ошибки при выполнении программы в реальной среде:** 2. отладка 3. контроль 4. **испытание** |
| 1. **Процесс локализации и исправления ошибок, обнаруженных при тестировании ПО называют:** 2. **отладкой** 3. локализацией 4. инициализацией |
| **22. Что относится к ошибкам кодирования:**   1. ошибки выполнения 2. **ошибки некорректного использования переменных, ошибки вычислений, ошибки взаимодействия модулей, игнорирование особенностей конкретного языка программирования** 3. логические ошибки |
| **23. Какой метод отладки программ описан в тексте: «Самый простой и естественный способ отладки программы. Метод эффективен, но не применим для программ со сложными вычислениями, для больших программ, а также в случаях, когда ошибка связана с неверным представлением программиста о выполнении операций»:**   1. метод индукции 2. **метод ручного тестирования** 3. метод обратного прослеживания |
| **24. Какой метод отладки программ описан в тексте: «Сначала формируют множество причин, которые могли бы вызвать данное проявление ошибки. Затем, анализируя причины, исключают те, которые противоречат имеющим данным.»:**   1. метод индукции 2. метод ручного тестирования 3. **метод дедукции** |
| **25. Какой метод отладки программ описан в тексте: «Метод основан на тщательном анализе симптомов ошибки, которые могут проявляться как неверные результаты вычислений или как сообщение об ошибке.»**   1. **метод индукции** 2. метод ручного тестирования 3. метод дедукции |
| **26. Какой метод отладки программ описан в тексте: «Начинается проверка с точки вывода неправильного результата. Для этой точки строится гипотеза о значениях основных переменных, которые могли бы привести к получению имеющегося результата»**   1. метод индукции 2. метод ручного тестирования 3. **метод обратного прослеживания** |
| 1. **27. Последовательность этапов отладки** 2. локализация ошибок, определение причины ошибки, исправление ошибки, изучение проявление ошибки 3. Изучение проявление ошибки, определение причины ошибки, локализация ошибок, исправление ошибки 4. Изучение проявление ошибки, локализация ошибок, повторное тестирование, определение причины ошибки, исправление ошибки 5. **изучение проявление ошибки, локализация ошибок, определение причины ошибки, исправление ошибки, повторное тестирование** |
| **28. Сложный алгоритм пишется или прототипируется на одном языке программирования (с наличием всех доступных инструментальных средств), а потом исходный код отлаженного алгоритма транслируется вручную или автоматически в другой язык программирования (целевой системы), для которого отсутствуют необходимые инструментальный средства**   1. Отладка методом RPC 2. **Отладка трансляцией кода** 3. Профилирование кода 4. Отладка разработкой интерпретатора |
| **29. Логирование кода**   1. выполнение автоматических unit test-ов в более простых сценариях для функций и автоматическое выявление проблемных участков кода 2. **вывод в файл входных, выходных аргументов функций, промежуточных состояний в процессе исполнения программы** 3. поиск причин возникновения дефекта с помощью анализа исходного кода программы, проблемного контента, конфигурации, состояния базы данных и т.п. 4. изолирование проблемы используя ручное или автоматическое тестирование |
| **30. Ди­на­ми­че­ские ана­ли­за­торы**   1. **кон­тро­ли­ру­ют вре­мя вы­пол­не­ния уча­ст­ков ко­да про­грам­мы, на­хо­дят точ­ки (об­лас­ти) не­кор­рект­ной ра­бо­ты с па­мя­тью и объ­ек­та­ми опе­ра­ци­он­ной сис­те­мы и др.** 2. вы­яв­ля­ют ошиб­ки вы­хо­да (об­ра­ще­ния) за гра­ни­цы мас­си­вов, по­тен­ци­аль­ные про­бле­мы безо­пас­но­сти, т. н. утеч­ки па­мя­ти, не­кор­рект­ность ис­поль­зо­ва­ния сис­тем­ных ре­сур­сов и др. |
| **31. Ста­ти­че­ские ана­ли­за­то­ры**   1. кон­тро­ли­ру­ют вре­мя вы­пол­не­ния уча­ст­ков ко­да про­грам­мы, на­хо­дят точ­ки (об­лас­ти) не­кор­рект­ной ра­бо­ты с па­мя­тью и объ­ек­та­ми опе­ра­ци­он­ной сис­те­мы и др. 2. **вы­яв­ля­ют ошиб­ки вы­хо­да (об­ра­ще­ния) за гра­ни­цы мас­си­вов, по­тен­ци­аль­ные про­бле­мы безо­пас­но­сти, т. н. утеч­ки па­мя­ти, не­кор­рект­ность ис­поль­зо­ва­ния сис­тем­ных ре­сур­сов и др.** |
| **32. Дефект**   1. **участок программного кода, выполнение которого при определенных условиях приводит к неожиданному поведению системы** 2. кратковременная самоустраняющаяся утрата работоспособности технического устройства 3. это проявление ошибки в программной системе и выражается в утрате работоспособности |
| **33. Отказ**   1. участок программного кода, выполнение которого при определенных условиях приводит к неожиданному поведению системы 2. **проявление ошибки в программной системе и выражается в утрате работоспособности** 3. кратковременная самоустраняющаяся утрата работоспособности технического устройства |
| **34. Сбой**   1. участок программного кода, выполнение которого при определенных условиях приводит к неожиданному поведению системы 2. **кратковременная самоустраняющаяся утрата работоспособности технического устройства** 3. проявление ошибки в программной системе и выражается в утрате работоспособности |
| **35. Верификация**   1. **процесс оценки системы или её компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, сформированным в начале этого этапа** 2. определение соответствия, разрабатываемого ПО ожиданиям и потребностям пользователя, требованиям к системе 3. процесс изменения программной системы таким образом, что её внешнее поведение не изменяется, а внутренняя структура улучшается |
| **36. Валидация**   1. процесс оценки системы или её компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, сформированным в начале этого этапа 2. **определение соответствия, разрабатываемого ПО ожиданиям и потребностям пользователя, требованиям к системе** 3. процесс изменения программной системы таким образом, что её внешнее поведение не изменяется, а внутренняя структура улучшается |
| **37. Рефакторинг**   1. процесс оценки системы или её компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, сформированным в начале этого этапа. 2. определение соответствия, разрабатываемого ПО ожиданиям и потребностям пользователя, требованиям к системе 3. **процесс изменения программной системы таким образом, что её внешнее поведение не изменяется, а внутренняя структура улучшается** |
| **38. Тестирование, предназначенное для проверки правильности функционирования методов классов ПО**   1. **Модульное тестирование** 2. Исследовательское тестирование 3. Функциональное тестирование 4. Нагрузочное тестирование 5. Регрессионное тестирование |
| **39. Тестирование, предназначенное для проверки работоспособности программного продукта при предельной входной нагрузке**   1. Модульное тестирование 2. Исследовательское тестирование 3. Функциональное тестирование 4. **Нагрузочное тестирование** 5. Регрессионное тестирование |
| **40. Тестирование, которое применяется при внесении изменений в программное обеспечение с целью проверки корректности работы компонентов системы, которые потенциально могут взаимодействовать с измененным компонентом**   1. Модульное тестирование 2. Исследовательское тестирование 3. Функциональное тестирование 4. Нагрузочное тестирование 5. **Регрессионное тестирование** |
| **41. Обработчик событий - это...**   1. метод программирования, основанным на управлении событиями 2. визуальное отображение окна приложения 3. **процедура (или функция) инициируемая событием** 4. совокупность данных и методов работы с ними |
| **42. Исключительные ситуации делятся**   1. Контролируемые 2. **Исправимые** 3. **Неисправимы**е |
| **43. Исключения делятся на**   1. **Аппаратные** 2. Пользовательские 3. **Программные** |
| **44. … - это отклонение программы от функционирования или невозможность программы выполнять функции, определенные требованиями и ограничениями, что рассматривается как событие, способствующее переходу программы в неработоспособное состояние из-за ошибок, скрытых в ней дефектов или сбоев в среде функционирования.**   1. Ошибка (error) 2. **Отказ (failure)** 3. Дефект (fault) |
| **45. … - состояние программы, при котором выдаются неправильные результаты, причиной которых являются изъяны в операторах программы или в технологическом процессе ее разработки, что приводит к неправильной интерпретации исходной информации, следовательно, и к неверному решению.**   1. **Ошибка (error)** 2. Отказ (failure) 3. Дефект (fault) |
| **46. … - следствие ошибок разработчика на любом из этапов разработки, которая может содержаться в исходных или проектных спецификациях, текстах кодов программ, эксплуатационной документация и т.п.**   1. Ошибка (error) 2. Отказ (failure) 3. **Дефект (fault)** |
| **47. Способы оценки надежности**   1. **Тестирование** 2. Сравнение с аналогами 3. Трассировка 4. Оптимизация |
| **48. Когда можно обнаружить синтаксические ошибки**   1. при отладке 2. при тестировании 3. **при компиляции** 4. при эксплуатации 5. на этапе проектирования |
| **49. Ошибки компоновки заключается в том, что**   1. **Указано внешнее имя, но не объявлено** 2. Составлено неверное выражение 3. Неправильно использовано зарезервированное слово 4. Указан неверный тип переменной |
| **50. Программа для просмотра значений переменных при выполнении программы:**   1. **Отладчик** 2. Компилятор 3. Интерпретатор |

**Компетенция ПК 2.5**

|  |
| --- |
| **1. Отладка программ это:**   1. **локализация и исправление ошибок** 2. алгоритмизация программирования 3. компиляция и компоновка |
| 1. **Компилятор – это:** 2. Прикладное программное обеспечение 3. Специальная утилита системного ПО 4. Операционная оболочка 5. Программное обеспечение, используемое в издательских системах 6. **Программа, которая переводит программу, написанную на языке программирования высокого уровня в программу на машинном языке** |
| **3. Отладчик:**   1. Программа для создания системы защиты файла 2. Программа создания системы защиты от вирусных атак 3. **Программа, помогающая анализировать поведение отлаживаемой программы, обеспечивая ее трассировку** 4. Операционная оболочка для создания и управления файловыми структурами 5. Системное программное обеспечение для настройки операционной системы 6. Программа создания и редактирования графических файлов 7. **Программа, позволяющая выполнять остановы в заданных точках, просмотреть текущие значения переменных и изменять их значения** |
| 1. **Процесс локализации и исправления ошибок, обнаруженных при тестировании программного обеспечения.** 2. **Отладка** 3. Локализация 4. Тестирование |
| **5. Определение оператора/операторов программы, выполнение которого вызвало нарушение вычислительного процесса.**   1. Отладка 2. **Локализация** 3. Тестирование |
| **6. Когда программист может проследить последовательность выполнения команд программы:**   1. при тестировании 2. **при трассировке** 3. при компиляции 4. при выполнении программы 5. при компоновке |
| **7. Процесс исполнения программы с целью обнаружения ошибок:**   * 1. кодирование   2. сопровождение   3. **тестирование**   4. проектирование |
| 1. **На языке программирования составляется:** 2. **исходный код** 3. исполняемый код 4. объектный код 5. алгоритм |
| 1. **Одно из преимуществ автоматизации программирования:** 2. **наглядное программирование с визуальным контролем** 3. получение стандартной программы 4. создание программы с оптимальным кодом |
| 1. **В чем отличия формальной инспекции от тестирования?** 2. **не происходит выполнения программного кода** 3. может применяться как к требованиям, так и к коду 4. более быстрый процесс 5. не фиксируются найденные проблемы |
| 1. **На каком этапе создания программы могут появиться синтаксические ошибки** 2. **Программирование** 3. Проектирование 4. Тестирование 5. Анализ требований |
| 1. **Верификация это** 2. **процесс проверки соответствия поведения системы требованиям** 3. процесс устранения ошибок в программном обеспечении 4. процесс взаимодействия с пользователем, направленный на улучшение его понимания принципов работы программной системы 5. процесс уточнения требований по результатам обсуждения с пользователем |
| **13. Вид ошибки с неправильным использованием служебных слов (операторов)**   1. синтаксическая 2. **семантическая** 3. логическая 4. символьная |