***Тест***

1. **Что такое технология разработки ПО?**

Технология разработки программного обеспечения — это совокупность методов, инструментов и процессов, используемых для проектирования, создания, тестирования и сопровождения программных систем. Она включает в себя выбор архитектуры, моделей жизненного цикла, языков программирования, методологий управления проектом и инструментов для автоматизации разработки.

1. **Достаточно ли при работе над проектом большой программной системы быть компетентным в области вычислительной техники и программировании. Почему?**

Нет, недостаточно. Помимо знаний в программировании, необходимы навыки системного анализа, управления проектами, тестирования, безопасности, а также понимание требований бизнеса и пользователей. Разработка крупной системы требует координации работы команды, управления изменениями и качеством.

1. **Может ли большая программная система быть отлажена до конца и почему?**

Полностью отладить большую программную систему практически невозможно, так как сложные системы содержат множество зависимостей, сценариев использования и потенциальных ошибок, которые трудно предусмотреть заранее. Кроме того, окружение и требования со временем меняются, создавая новые проблемы.

1. **При каких условиях созданный программный комплекс может быть назван программным продуктом?**

Программный комплекс становится программным продуктом, если он отвечает требованиям пользователей, имеет документацию, прошел тестирование, обладает коммерческой или практической ценностью и поддерживается разработчиками

1. **Что такое системное программное обеспечение?**

Системное программное обеспечение — это программы, управляющие аппаратными ресурсами и обеспечивающие работу других программ. Включает операционные системы, драйверы, утилиты и компиляторы.

1. **Что такое инструментарий технологии программирования?**

Инструментарий технологии программирования — это совокупность средств, используемых для разработки, тестирования, отладки, сопровождения и развертывания программного обеспечения. Он включает в себя:

Языки программирования – Python, Java, C++, JavaScript и другие.

Среды разработки (IDE) – Visual Studio, IntelliJ IDEA, PyCharm, Eclipse.

Компиляторы и интерпретаторы – GCC, Clang, CPython, JVM.

Системы контроля версий – Git, SVN, Mercurial.

Фреймворки и библиотеки – React, Django, Spring, TensorFlow.

Средства тестирования – JUnit, Selenium, PyTest.

Средства отладки и профилирования – GDB, Valgrind, Perf, Chrome DevTools.

Системы сборки и управления зависимостями – Maven, Gradle, CMake, npm.

Системы управления базами данных (СУБД) – MySQL, PostgreSQL, MongoDB.

Средства автоматизации развертывания и DevOps – Docker, Kubernetes, Jenkins, Ansible.

1. **Понятие жизненного цикла ПО. Что понимается под процессом жизненного цикла? Назовите основные группы процессов согласно ГОСТ Р ИСО/МЭК 12207-2010.**

Жизненный цикл программного обеспечения (ПО) – это последовательность стадий, через которые проходит программный продукт от момента возникновения идеи до завершения его эксплуатации. Это включает разработку, внедрение, сопровождение и, в конечном итоге, вывод из эксплуатации.

Процесс жизненного цикла

* Процесс жизненного цикла ПО – это совокупность действий, выполняемых на протяжении всего существования программного продукта. Он включает планирование, создание, тестирование, развертывание, эксплуатацию и утилизацию ПО.

Согласно стандарту ГОСТ Р ИСО/МЭК 12207-2010, процессы жизненного цикла ПО делятся на три основные группы:

1. Основные процессы (Primary Processes) – непосредственно связаны с разработкой и эксплуатацией ПО:

* Приобретение (Acquisition) – определение требований заказчика, выбор поставщика.
* Разработка (Development) – проектирование, кодирование, тестирование.
* Эксплуатация (Operation) – развертывание и использование ПО.
* Сопровождение (Maintenance) – исправление ошибок, обновление ПО.

1. Вспомогательные процессы (Supporting Processes) – обеспечивают качество и эффективность основных процессов:

* Документирование (Documentation).
* Управление конфигурацией (Configuration Management).
* Оценка качества (Quality Assurance).
* Верификация и валидация (Verification and Validation).
* Аудит (Audit).
* Решение проблем (Problem Resolution).

1. Организационные процессы (Organizational Processes) – направлены на управление и улучшение разработки ПО:

* Управление проектами (Project Management).
* Создание инфраструктуры (Infrastructure Management).
* Управление рисками (Risk Management).
* Обеспечение процессов (Process Assurance).
* Совершенствование процессов (Process Improvement).

Жизненный цикл программного обеспечения может быть реализован разными моделями (каскадная, спиральная, Agile, DevOps), в зависимости от требований проекта.

1. **Основная цель процесса анализа требований к программным средствам. Что является результатом успешного осуществления процесса?**

Анализ требований – это процесс выявления, формулирования, документирования, проверки и согласования требований к программному обеспечению (ПО).

Основная цель анализа требований – определить четкие, полные и непротиворечивые требования, которые будут служить основой для проектирования, разработки и тестирования ПО. Это помогает:

* минимизировать риски изменения требований на поздних этапах,
* улучшить понимание системы разработчиками, заказчиками и пользователями,
* обеспечить соответствие ПО бизнес-задачам и пользовательским ожиданиям.

Если анализ требований проведен правильно, то его результатом является:

1. Документированные требования – оформленные в виде Спецификации требований к ПО (Software Requirements Specification, SRS) или аналогичного документа.
2. Ясность и полнота требований – они однозначны, непротиворечивы и проверяемы.
3. Согласование требований – между заказчиком, пользователями, разработчиками и другими заинтересованными сторонами.
4. Определение критериев приемки – четкие условия, при которых система считается соответствующей требованиям.
5. Основы для проектирования и тестирования – требования служат базой для разработки архитектуры, кода и тест-кейсов.

Таким образом, успешный анализ требований значительно снижает риски, связанные с недопониманием, изменениями и ошибками в ПО.

1. **Процесс реализации. Какие виды деятельности и задачи входят в состав процесса реализации?**

Процесс реализации – это этап жизненного цикла программного обеспечения, на котором выполняется непосредственная разработка ПО, включая программирование, интеграцию и первоначальное тестирование.

Процесс реализации включает несколько ключевых видов деятельности:

1. Разработка архитектуры и проектирование системы

* Определение общей структуры программного обеспечения.
* Разработка модульной архитектуры, интерфейсов и компонентов.
* Подготовка технической документации.

1. Программирование (кодирование)

* Написание кода на выбранном языке программирования.
* Использование стандартов кодирования и оформления.
* Реализация алгоритмов и логики работы системы.

1. Интеграция модулей

* Объединение отдельных компонентов системы в единое целое.
* Проверка совместимости и взаимодействия между модулями.
* Разрешение конфликтов между компонентами.

1. Первичное тестирование (unit-тестирование, модульное тестирование)

* Тестирование отдельных частей программы на предмет корректности работы.
* Автоматизированное тестирование (если предусмотрено).
* Исправление выявленных ошибок.

1. Оптимизация кода и производительности

* Анализ и улучшение скорости работы системы.
* Оптимизация использования ресурсов (памяти, процессора и т. д.).
* Устранение узких мест в производительности.

1. Документирование кода и системы

* Создание документации для разработчиков (описание API, алгоритмов, комментарии в коде).
* Подготовка пользовательской документации (руководства, инструкции).

1. Контроль версий и управление изменениями

* Использование систем контроля версий (Git, SVN).
* Ведение истории изменений.
* Организация командной работы над кодом.

1. **Процесс проектирования архитектуры программных средств. Что является результатом успешной реализации процесса. Что понимается под базовой линией?**

Процесс проектирования архитектуры программных средств – это создание структуры ПО, определяющей компоненты, их взаимодействие и технологии. Включает выбор стека технологий, проектирование взаимодействий и модели данных.

Результат успешного процесса:

* Архитектурное описание ПО.
* Спецификация работы системы.
* Выбранный стек технологий.
* Архитектурные паттерны.
* Базовая линия архитектуры – утвержденная версия, которая служит основой для реализации.

Базовая линия – это утвержденная версия архитектуры или документа, которая фиксируется и становится основой для дальнейшей разработки.

1. **Что понимается под моделью ЖЦ ПО? Назовите существующие модели ЖЦ ПО.**

Модель жизненного цикла ПО (ЖЦ ПО) — это абстрактное представление процесса разработки программного обеспечения, которое описывает его стадии и последовательность действий. Модели помогают организовать процесс разработки, улучшить управление и повысить качество ПО.

Основные модели ЖЦ ПО:

1. Каскадная модель (Waterfall) — последовательное выполнение стадий: анализ требований, проектирование, кодирование, тестирование, развертывание, поддержка.
2. Спиральная модель — итеративный подход, который сочетает элементы каскадной модели с рисковыми анализами и многократным циклом разработки.
3. Инкрементная модель — ПО разрабатывается поэтапно, каждый этап представляет собой часть конечного продукта, который постепенно дополняется новыми функциями.
4. Модель V — расширение каскадной модели, где каждая стадия разработки соответствует стадии тестирования.
5. Agile (Гибкая модель) — адаптивный подход, ориентированный на гибкость и итеративную разработку с частыми релизами и тесным взаимодействием с заказчиком.
6. DevOps — интеграция разработки и эксплуатации ПО с целью ускорения выпуска обновлений и улучшения взаимодействия между командами.
7. **Чем модель ЖЦ ПО отличается от методологии разработки ПО? Назовите существующие гибкие методологии разработки ПО.**

Модель жизненного цикла ПО (ЖЦ ПО) описывает этапы и последовательность процессов, через которые проходит программное обеспечение в ходе своего существования — от планирования до вывода из эксплуатации. Модели фокусируются на том, как и в каком порядке выполняются эти этапы (например, каскадная модель, Agile, спиральная и т. д.).

Методология разработки ПО — это набор принципов, практик, техник и методов, которые используются для управления процессом разработки. Методология определяет, как будет осуществляться разработка ПО, а также взаимодействие команды, заказчика и других заинтересованных сторон. Она более конкретно фокусируется на подходах к решению задач разработки, а не только на этапах жизненного цикла.

Разница:

* Модель ЖЦ ПО фокусируется на последовательности этапов (как протекает процесс).
* Методология разработки ПО определяет подходы, методы и практики, используемые на этих этапах (как выполняются задачи на каждом этапе).

Гибкие методологии разработки ПО:

1. Agile (Гибкая методология) — ориентирована на итеративную разработку с быстрым реагированием на изменения и тесным взаимодействием с заказчиком.
2. Scrum — фреймворк для управления проектами, который делит работу на небольшие итерации (спринты), с регулярными встречами для отслеживания прогресса.
3. Kanban — методология управления процессом разработки, фокусирующаяся на визуализации работы и управлении потоком задач с минимальными задержками.
4. Extreme Programming (XP) — фокусируется на высококачественном кодировании с частыми релизами, тесным взаимодействием с заказчиком и непрерывным тестированием.
5. Lean Software Development — методология, направленная на минимизацию потерь и повышение эффективности разработки путем оптимизации процессов.
6. **Назовите основные особенности и стадии «Каскадной модели».**

Каскадная модель — это классическая модель жизненного цикла ПО, где процесс разработки разделен на четко определенные последовательные этапы. Каждый этап должен быть завершен перед переходом к следующему, что делает модель линейной и последовательной.

Основные особенности каскадной модели:

1. Последовательность этапов: каждый этап жизненного цикла выполняется строго после завершения предыдущего.
2. Документирование на каждом этапе: на каждом этапе создаются и утверждаются документы, которые служат основой для следующего шага.
3. Ригидность: изменения на поздних этапах (после завершения этапа) сложно вносить, что делает модель менее гибкой.
4. Подходит для проектов с четкими и стабильными требованиями: каскадная модель хорошо работает, когда требования известны заранее и не будут сильно изменяться в процессе разработки.

Стадии каскадной модели:

1. Анализ требований — сбор и документирование всех требований к системе.
2. Проектирование системы — создание архитектуры и проектирование всех компонентов системы.
3. Разработка (кодирование) — написание исходного кода по проекту.
4. Тестирование — проверка работоспособности, исправление ошибок и проверка соответствия требований.
5. Развертывание — установка и настройка системы в рабочей среде.
6. Эксплуатация и обслуживание — поддержка системы, исправление багов, обновления.
7. **Назовите основные особенности и стадии «Эволюционной модели».**

Эволюционная модель — это итеративный подход к разработке ПО, где система создается и улучшаетс я через несколько циклов с учетом обратной связи.

Особенности:

* Разработка происходит через несколько итераций.
* Каждая версия — улучшение предыдущей.
* Легкость в учете изменений требований на каждом цикле.
* Регулярная обратная связь с пользователями.

Стадии:

1. Планирование и анализ — определяются начальные требования.
2. Проектирование и разработка — создается первая рабочая версия.
3. Тестирование — проверка и выявление проблем.
4. Обратная связь — сбор мнений пользователей.
5. Улучшение — доработка системы на основе полученной информации.
6. Повторение цикла — процесс повторяется до достижения окончательной версии
7. **Методология Scrum. Что такое Спринт в рамках методологии Scrum? Какие группы ролей определены в данной методологии**

Методология Scrum — это гибкий фреймворк для управления проектами, который основывается на итеративном подходе и тесном взаимодействии между командой и заказчиком.

Спринт в Scrum:

Спринт — это ограниченный по времени период (обычно 2–4 недели), в течение которого команда выполняет определенный набор задач и доставляет рабочую версию продукта.

Роли в Scrum:

1. Product Owner (Владелец продукта) — отвечает за управление продуктовым бэклогом и приоритетами задач.
2. Scrum Master — помогает команде следовать процессам Scrum, устраняет препятствия и поддерживает эффективность.
3. Development Team (Команда разработки) — группа специалистов, которые непосредственно выполняют работу над продуктом.
4. **Перечислите основные стандарты, описывающие оценку качества программного обеспечения.**

Основные стандарты для оценки качества программного обеспечения:

1. ISO/IEC 9126 — стандарт, описывающий модели качества ПО, включая характеристики, такие как функциональность, надежность, удобство использования и производительность.
2. ISO/IEC 25010 — обновленный стандарт ISO/IEC 9126, определяет качество ПО через 8 характеристик: функциональность, производительность, совместимость, удобство использования, безопасность, поддерживаемость, переносимость и тестируемость.
3. IEEE 829 — стандарт, описывающий процессы тестирования ПО и формат документации для тестирования.
4. CMMI (Capability Maturity Model Integration) — модель для оценки зрелости процессов разработки и управления качеством ПО.
5. ISO/IEC 12207 — стандарт, определяющий процессы жизненного цикла ПО, включая процессы, связанные с обеспечением качества.
6. **Назовите факторы качества программного обеспечения.**

Основные факторы качества программного обеспечения:

1. Функциональность — способность ПО выполнять требуемые функции.
2. Надежность — устойчивость ПО к ошибкам и сбоям.
3. Производительность — эффективность работы системы (скорость, использование ресурсов).
4. Удобство использования — простота и удобство взаимодействия с ПО.
5. Безопасность — защита от несанкционированного доступа и утечек данных.
6. Поддерживаемость — легкость в модификации и исправлении ошибок.
7. Совместимость — способность ПО работать в различных средах и с другими системами.
8. Переносимость — способность ПО работать на разных платформах.
9. **Что такое модель зрелости? Особенности.**

Модель зрелости — это система оценки процессов разработки и управления ПО, которая помогает определить уровень их зрелости и улучшения.

Особенности модели зрелости:

* Этапы развития — модель делит процессы на несколько уровней зрелости, от начального до оптимизированного.
* Постепенное улучшение — на каждом уровне внедряются новые практики для повышения эффективности.
* Цель — улучшение качества разработки ПО, управление рисками и повышение производительности.
* Оценка процессов — помогает организациям определить, какие процессы нужно улучшать и как.

Пример: CMMI (Capability Maturity Model Integration), где выделяют 5 уровней зрелости: от начального (неупорядоченные процессы) до оптимизированного (постоянные улучшения).

1. **Какие модели качества процессов разработки вы знаете?**

Основные модели качества процессов разработки:

1. CMMI (Capability Maturity Model Integration) — модель зрелости процессов, оценивающая и улучшая процессы разработки ПО через 5 уровней зрелости.
2. ISO/IEC 15504 (SPICE) — стандарт для оценки процессов в разработке ПО, который помогает улучшить процесс разработки и управления качеством.
3. ISO 9001 — международный стандарт, устанавливающий требования к системам менеджмента качества, применяемые в разработке ПО.
4. Six Sigma — методология, ориентированная на минимизацию дефектов и улучшение процессов, основываясь на статистическом анализе.
5. ITIL (Information Technology Infrastructure Library) — фреймворк для управления IT-услугами, включая улучшение процессов разработки и эксплуатации ПО.
6. **Назовите основные цели, преследуемые при анализе требований в проектах.**

Основные цели анализа требований в проектах:

1. Выявление потребностей заказчика — точное определение, что нужно от системы.
2. Документирование требований — создание четкой и понятной спецификации, которая служит основой для разработки.
3. Устранение противоречий — идентификация и разрешение несоответствий и конфликтов в требованиях.
4. Определение приоритетов — установление важности и срочности различных требований.
5. Обеспечение согласования — согласование требований с заказчиком и другими заинтересованными сторонами.
6. Основы для тестирования — создание критериев приемки и проверки качества ПО.
7. **Перечислите типы требований.**

Основные типы требований:

1. Функциональные требования — описывают, какие функции и задачи система должна выполнять.
2. Нефункциональные требования — определяют качества системы, такие как производительность, безопасность, удобство использования.
3. Требования к интерфейсам — описывают взаимодействие системы с другими системами или пользователями.
4. Требования к данным — касаются структуры, хранилищ и управления данными.
5. Ограничения — условия, которые ограничивают решение, например, бюджетные или временные ограничения.
6. Бизнес-требования — определяют цели и задачи, которые должны быть достигнуты для удовлетворения бизнеса.
7. **Назовите методы выявления требований.**

Основные методы выявления требований:

1. Интервью — беседы с заказчиками, пользователями и экспертами для сбора информации.
2. Опросы и анкеты — использование вопросов для получения обратной связи от пользователей.
3. Наблюдение — наблюдение за тем, как пользователи взаимодействуют с текущей системой.
4. Анализ документации — изучение существующих документов, таких как отчеты, спецификации и стандарты.
5. Мозговые штурмы — коллективное обсуждение для выявления идей и требований.
6. Прототипирование — создание модели системы для получения отзывов и уточнений требований.
7. Рабочие группы — обсуждения с заинтересованными сторонами для выявления требований и ожиданий.
8. Использование случаев — анализ конкретных сценариев использования системы.
9. **Перечислите задачи, которые решаются на стадии анализа требований.**

Задачи, решаемые на стадии анализа требований:

1. Выявление потребностей заказчика — уточнение и понимание требований и ожиданий.
2. Документирование требований — создание четкой и структурированной документации.
3. Анализ и уточнение требований — выявление неполных, противоречивых или неясных требований.
4. Приоритизация требований — определение важности и приоритетности для разработки.
5. Определение ограничений — выявление технических, временных и ресурсных ограничений.
6. Согласование требований — утверждение требований с заказчиком и другими заинтересованными сторонами.
7. Создание критериев приемки — разработка критериев для проверки выполнения требований на всех этапах.
8. **Аналитик требований. Перечислите основные задачи аналитика требований.**

Основные задачи аналитика требований:

1. Выявление требований — сбор информации от заказчиков и пользователей.
2. Документирование требований — создание четкой, понятной и структурированной документации.
3. Анализ требований — проверка требований на полноту, согласованность и выполнимость.
4. Приоритизация требований — определение, какие требования наиболее важны для реализации.
5. Моделирование требований — создание моделей (диаграмм, прототипов) для визуализации требований.
6. Согласование требований — подтверждение требований с заказчиком и заинтересованными сторонами.
7. Управление изменениями — отслеживание и документирование изменений требований на протяжении проекта.