Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

кафедра программного обеспечения информационных технологий

ОТЧЕТ

по индивидуальной работе

**«ОРГАНИЗАЦИЯ ТЕХНОЛОГИЧЕСКОГО ПРОЦЕССА РАЗРАБОТКИ ПРОГРАММНЫХ ПРОДУКТОВ»**

Выполнил:

Студент 1 курса КСиС ПИ

Кучмель Эдуард Русланович

Минск 2023

1. **Дайте определение терминам: приложение, программный продукт, утилита.**

**Приложение** - это программное обеспечение, созданное для выполнения определенных задач на устройстве, таком как компьютер, смартфон или планшет. Оно предоставляет пользователю интерфейс для взаимодействия с устройством и может выполнять различные функции, от обработки документов до игр и социальных сетей.

**Программный продукт** - это совокупность программ и данных, предназначенных для решения определенных задач или проблем. Он может включать одно или несколько приложений, а также дополнительные ресурсы, такие как библиотеки, конфигурационные файлы и документация. Программные продукты создаются, распространяются и поддерживаются разработчиками.

**Утилита** - это небольшая компьютерная программа, предназначенная для выполнения ограниченного набора специфических задач или функций. Утилиты часто используются для обеспечения системных операций и управления ресурсами компьютера, такими как архивация файлов, проверка на вирусы, оптимизация диска, резервное копирование данных и другие подобные задачи. Утилиты могут быть частью операционной системы или сторонними приложениями.

1. **«Жизненный цикл программного продукта» (ЖЦ ПП) – дайте определение и перечислите основные этапы.**

**Жизненный цикл программного продукта (ЖЦ ПП)** - это последовательность этапов и процессов, через которые проходит программный продукт с момента его создания и разработки до его вывода из эксплуатации. ЖЦ ПП охватывает всю жизненную историю программного продукта, включая его концепцию, проектирование, разработку, тестирование, внедрение, эксплуатацию, поддержку и, в конечном итоге, вывод из эксплуатации.

Основные этапы жизненного цикла программного продукта включают:

* Концепция. На этом этапе определяются цели и требования к программному продукту. Разрабатывается общее представление о том, что должен делать продукт и какие проблемы он должен решать.
* Планирование. Создается план разработки, который включает в себя определение бюджета, расписания, ресурсов и других аспектов управления проектом. Определяются сроки и этапы разработки.
* Проектирование. На этом этапе разрабатывается детальное техническое описание программного продукта. Проектируются архитектура, интерфейсы, базы данных и другие элементы продукта.
* Разработка. На этом этапе создаются код и компоненты программного продукта. Разработчики пишут программный код в соответствии с требованиями и дизайном, определенными на предыдущих этапах.
* Тестирование. Программный продукт подвергается различным видам тестирования, включая модульное, интеграционное, системное тестирование и тестирование на соответствие требованиям. Цель - выявить и устранить ошибки и дефекты.
* Внедрение. Программный продукт развертывается и внедряется в рабочей среде. Это может включать установку на сервера, распространение клиентских версий и обучение пользователей.
* Эксплуатация. Программный продукт используется конечными пользователями в реальных условиях. Осуществляется поддержка и обслуживание, включая решение проблем и обновления.
* Сопровождение. Продукт поддерживается в актуальном состоянии. В этот этап входят исправление ошибок, внесение улучшений и выпуск новых версий.
* Вывод из эксплуатации. Когда продукт устаревает или больше не требуется, он выводится из эксплуатации. Это включает в себя архивирование данных и завершение поддержки.

Эти этапы могут варьироваться в зависимости от методологии разработки и характеристик проекта, но представляют общий обзор жизненного цикла программного продукта.

1. **Дайте определение терминам: интерфейс, пользовательский интерфейс, графический интерфейс.**

**Интерфейс** (Interface) - это точка взаимодействия или область, где две системы, компоненты или объекта встречаются и взаимодействуют друг с другом. Это может быть набор методов, свойств или функций, которые определяют, как можно взаимодействовать с определенным объектом или системой.

**Пользовательский интерфейс** (User Interface, UI) - это часть программы или системы, через которую пользователь взаимодействует с компьютером или другим устройством. UI включает в себя элементы дизайна, такие как кнопки, поля ввода, меню, окна и другие элементы, которые позволяют пользователю взаимодействовать с программой или устройством.

**Графический интерфейс** (Graphical User Interface, GUI) - это форма пользовательского интерфейса, который предоставляет пользователю визуальные элементы для взаимодействия с программой или устройством. GUI включает в себя использование графических элементов, таких как иконки, кнопки, изображения и графические окна, чтобы облегчить навигацию и взаимодействие пользователя с системой. GUI обычно считается более интуитивным и доступным для неопытных пользователей по сравнению с текстовым или командным интерфейсом.

1. **Какова основная цель в составлении «Технического задания» (ТЗ)?**

Основная цель в составлении "Технического задания" (ТЗ) заключается в том, чтобы документировать и четко определить требования, характеристики и ожидания относительно разрабатываемого проекта или продукта. ТЗ играет важную роль в процессе разработки и является ключевым инструментом для успешного выполнения проекта.

1. **Для решения каких основных задач предназначаются «Системы контроля версий» (СКВ)?**

**Управление версиями кода**: СКВ позволяют разработчикам отслеживать изменения в исходном коде программы. Они сохраняют историю изменений, позволяя переходить между разными версиями кода, восстанавливать предыдущие версии и сравнивать изменения.

**Совместная разработка**: СКВ облегчают совместную работу нескольких разработчиков над одним проектом. Они предоставляют механизмы для слияния изменений, резервирования ресурсов и координации работы.

**Работа в разных ветках**: СКВ позволяют создавать разные ветки (branches) кода для параллельной разработки различных функций или исправлений ошибок. Это помогает избегать конфликтов и обеспечивает изоляцию изменений.

**Отслеживание ошибок и задач**: СКВ могут интегрироваться с системами управления ошибками и задачами, что облегчает отслеживание и решение проблем в процессе разработки.

**Автоматизация процесса сборки и развертывания:** СКВ могут использоваться для автоматической сборки и развертывания приложений. Это позволяет упростить процесс развертывания на тестовых и производственных серверах.

**Аудит и анализ изменений**: СКВ сохраняют историю всех изменений в проекте, что позволяет проводить аудит изменений и анализировать, кто и когда внес какие изменения.

**Восстановление после сбоев**: СКВ обеспечивают возможность восстановления кода и данных после сбоев или потери информации.

**Управление доступом**: СКВ позволяют управлять доступом к репозиторию кода, определяя, кто имеет доступ на чтение, запись или администрирование.

**Документация и комментарии**: СКВ поддерживают добавление комментариев и документации к изменениям, что помогает документировать процесс разработки.

**Сравнение версий**: Разработчики могут сравнивать разные версии кода для выявления изменений и решения конфликтов.

1. **Перечислите известные вам СКВ?**

**Git, SVN**, **Mercurial, Perforce**

1. **Для решения каких задач предназначаются Системы управления дефектами (СУД)?**

**Отслеживание ошибок и дефектов**: Основная задача СУД - отслеживание и документирование ошибок, дефектов и неполадок в программном обеспечении. Это позволяет разработчикам и тестировщикам вести учет всех обнаруженных проблем.

**Управление задачами и заданиями**: СУД позволяют создавать, назначать и отслеживать задачи и задания, связанные с разработкой программного обеспечения. Это включает в себя задачи по исправлению ошибок, добавлению новых функций, тестированию и другие задания.

**Приоритизация и классификация задач**: СУД позволяют устанавливать приоритеты для задач и классифицировать их по различным критериям, таким как серьезность ошибки, сроки исправления и другие.

**Отчетность и анализ**: СУД предоставляют возможность создавать отчеты и анализировать статистику по ошибкам и задачам. Это помогает управляющим и разработчикам принимать информированные решения.

**Совместная работа и коммуникация**: СУД обеспечивают средства для совместной работы и коммуникации в команде разработки. Разработчики, тестировщики и другие участники могут обмениваться информацией и комментариями к задачам.

**Управление жизненным циклом задач**: СУД позволяют отслеживать статусы задач на разных этапах их жизненного цикла, начиная с создания и заканчивая завершением.

**Интеграция с другими инструментами**: Многие СУД могут интегрироваться с другими инструментами разработки, такими как системы контроля версий, системы сборки и непрерывной интеграции, что облегчает автоматизацию процессов.

**Управление релизами и версиями**: СУД могут помогать в управлении релизами программного обеспечения, отслеживая, какие задачи включены в каждую версию и какие исправления внесены.

**Улучшение качества продукта**: СУД помогают обнаруживать, исправлять и тестировать ошибки, что способствует повышению качества программного продукта.

**Служба поддержки и обратная связь пользователей**: СУД могут использоваться для управления запросами на поддержку и обратной связи от пользователей, что позволяет оперативно реагировать на проблемы и потребности пользователей.

1. **Перечислите известные вам СУД?**

**JIRA, Trello, GitHub Issues.**

1. **Что такое IDE? Расшифруйте и дайте определение.**

**IDE -** это сокращение от "Integrated Development Environment" (Интегрированная Среда Разработки). Это программное обеспечение, предназначенное для облегчения и ускорения процесса разработки программного обеспечения. IDE объединяет в себе различные инструменты и функции, необходимые разработчику, чтобы создавать, отлаживать и управлять проектами. Основной целью IDE является увеличение продуктивности разработчика и снижение сложности создания программ.

1. **Опишите используемую Вами IDE. Особенности и преимущества.**

**Rad Studio 11:**

**Особенности:** Rad Studio 11 - это интегрированная среда разработки, обеспечивающая создание многоплатформенных приложений для различных ОС с помощью Delphi, C++Builder и HTML5 Builder. Предлагает широкий набор инструментов разработки, визуальные компоненты, отладку, профилирование и обновления для эффективной работы разработчиков.

**Преимущества:** Его многоплатформенность, разнообразие языков программирования и обширные наборы визуальных компонентов обеспечивают удобство и скорость разработки. Инструменты отладки и профилирования помогают эффективно управлять кодом и улучшать производительность приложений. Постоянные обновления и поддержка позволяют быть в курсе последних технологических трендов и решать проблемы оперативно.