Бюджетное профессиональное образовательное учреждение
Вологодское области
«Череповецкий лесомеханический техникум им. В. П.Чкалова»

«ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ»

Выполнили студенты
Сезёмин Артём
Дуденков Никита
По специальности 09.02.07.
Информационные системы и программирование
Руководитель: Гончарова Лана Николаевна

Г.Череповец

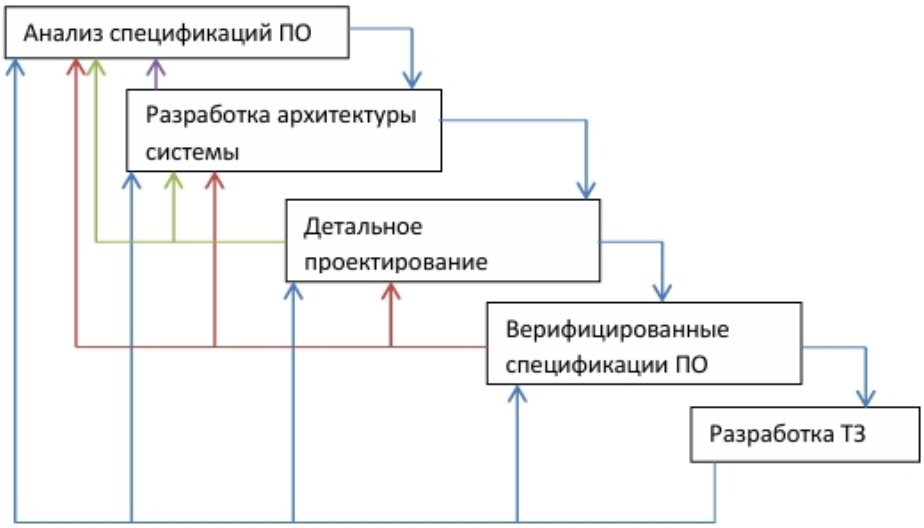
Содержание

1 практическая

Цель работы: изучить современные инструменты и средства разработки программного обеспечения.

Определение ПО: программный продукт — это совокупность программ и сопутствующей документации, предназначенных для решения прикладных задач и обеспечения работы аппаратных средств ЭВМ.

Технология программирования: включает методы и инструменты, используемые в процессе создания программного обеспечения.



1. Написание программы в соответствии с требованиями и дизайном, определёнными на предыдущих этапах.
2. Создание модулей, компонентов и функциональных частей программы.
3. Документирование кода — создание документации по коду с инструкциями для других разработчиков, а также руководства по функциям приложения для конечных пользователей.

Цикл проектирования ПО:

- Создание исходного кода: написание программ на выбранных языках с помощью текстовых редакторов.
- Перевод кода: использование компиляторов или интерпретаторов для преобразования исходного кода в машинный.

- Объединение компонентов: редактирование связей (линковка) для формирования исполняемого файла, объединяющего объектные модули и системные библиотеки.
- Отладка: использование отладчиков для поиска и устранения ошибок в коде.

Инструменты разработки:

1. Текстовые редакторы для написания кода.
2. Трансляторы (компиляторы и интерпретаторы).
3. Редакторы связей (линкеры) для сборки исполняемых файлов.
4. Отладчики для поиска ошибок.
5. Справочные системы для получения информации о языках программирования и инструментах.
6. Системы управления базами данных (СУБД).

Функции современных компиляторов:

- Перевод: преобразование кода в машинные инструкции, понятные компьютеру.
- Проверка: выявление синтаксических ошибок до запуска программы.
- Оптимизация: повышение скорости выполнения и снижение потребления ресурсов за счет оптимизации кода.

Современные средства программирования:

- **Python**: язык общего назначения, популярен в веб-разработке, анализе данных и машинном обучении благодаря простоте синтаксиса.
- **C++:** мощный язык для создания различных приложений, включая игры и системное программное обеспечение.
- **VSS (Visual SourceSafe):** устаревшая система контроля версий от Microsoft, также используется для создания снимков данных.
- **MS Visual Studio**: интегрированная среда разработки (IDE) для редактирования, отладки и сборки программ.

2 практическая

Основные понятия и стандартизация требований к программному обеспечению.

Цель: Научится проводить анализ предметной области разрабатываемого ПО.

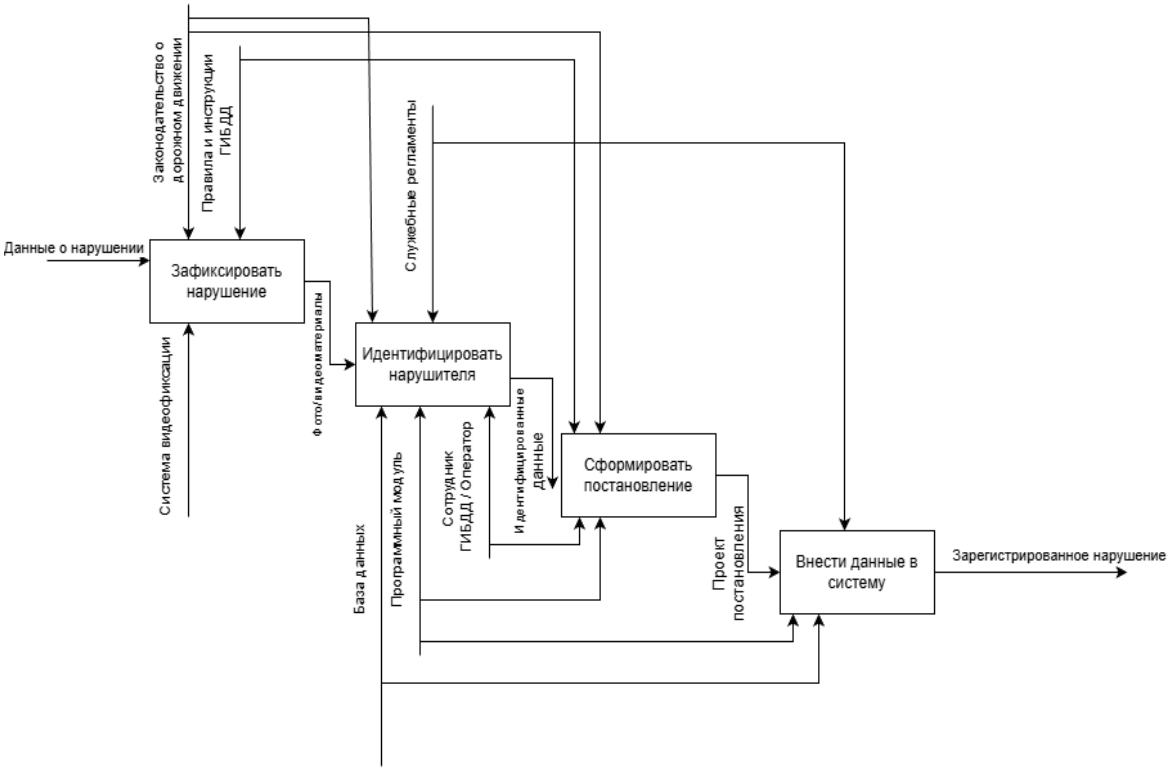
Анализ предметной области

Процесс организации работы с нарушителями правил дорожного движения с точки зрения работника милиции:

1. **Выявление нарушения. Сотрудник ДПС может выявить нарушение несколькими способами.**
   - Визуально, наблюдая за дорожным движением с патрульной машины или поста.
   - С помощью специальных технических средств (радаров, алкотестеров, камер видеонаблюдения).
   - Получив информацию от граждан или из автоматизированных систем видеофиксации.
2. **Остановка транспортного средства.**
3. **Взаимодействие с водителем.**
   - После остановки сотрудник подходит к автомобилю, представляется, предъявляет служебное удостоверение и сообщает причину остановки.
   - Происходит проверка документов - водительского удостоверения, свидетельства о регистрации транспортного средства (СТС) и, при необходимости, страхового полиса.
4. **Разбирательство и оформление нарушения.**
   - Если нарушение подтверждается, сотрудник составляет протокол об административном правонарушении. В протоколе указывается дата, время, место, вид нарушения и другие обстоятельства.

- В случае, если нарушение незначительное, возможно вынесение предупреждения.
- При серьезных нарушениях, например, при вождении в состоянии опьянения, могут быть применены более строгие меры (задержание, лишение прав).

5. **Назначение и контроль оплаты штрафа.**
   - После составления протокола или постановления, информация о нарушении вносится в единую базу данных.
   - Водитель получает квитанцию для оплаты штрафа.
   - Информация об оплате должна поступить в базу данных из системы межведомственного электронного взаимодействия. Если штраф оплачен, но в базе данных по-прежнему числится как неоплаченный, владельцу необходимо предоставить квитанцию об оплате в ГИБДД.


Законодательство о дорожном движении
Правила и инструкции ГИБДД
Служебные регламенты
Данные о нарушении
Зафиксировать нарушение
Идентифицировать нарушителя
Сформировать постановление
Внести данные в систему
Зарегистрированное нарушение
Система видеофиксации
Фото/видеоматериалы
Идентифицированные данные
Сотрудник ГИБДД / Оператор
Программный модуль
База данных
Проект постановления

3 практическая

Техническое задание

Учет нарушений правил дорожного движения

Введение

В условиях роста числа транспортных средств и повышения требований к безопасности дорожного движения возрастает нагрузка на сотрудников органов внутренних дел, отвечающих за контроль соблюдения Правил дорожного движения (ПДД). Ручной учёт нарушений и штрафов становится неэффективным и подвержен ошибкам. Автоматизация процесса учёта нарушений ПДД позволит повысить оперативность, достоверность данных и упростить взаимодействие между сотрудниками милиции и владельцами транспортных средств. Разрабатываемый модуль займёт ключевое место в системе административного контроля за дорожным движением.

Наименование и область применения

Наименование программы: «Учет нарушений правил дорожного движения».

Область применения: Программный модуль предназначен для использования сотрудниками подразделений Государственной автомобильной инспекции (ГАИ/ГИБДД) при оформлении, учёте и контроле оплаты административных штрафов за нарушения ПДД.

Объект применения: Автоматизированное рабочее место инспектора ДПС.

Основание для разработки

Разработка осуществляется на основании:

- Приказа Управления внутренних дел № 112-П от 15.03.2025 «О внедрении цифровых решений в деятельность подразделений ГИБДД»;
- Плана цифровой трансформации МВД РФ на 2024–2026 гг.;

- Тема разработки: «Автоматизация учёта нарушений ПДД и контроля оплаты штрафов».

Назначение разработки

Программный модуль предназначен для:

- регистрации транспортных средств и их владельцев;
- фиксации нарушений ПДД с указанием даты, времени, вида нарушения и суммы штрафа;
- хранения истории нарушений по каждому автомобилю;
- отслеживания статуса оплаты штрафов;
- автоматического удаления записи об автомобиле из базы после полной оплаты всех штрафов.

Технические требования к программе или программному изделию

Требования к функциональным характеристикам

Программа должна обеспечивать следующие функции:

- добавление нового автомобиля (гос. номер, марка, модель, ФИО владельца, контактные данные);
- регистрация нарушения: дата, время, тип нарушения (из справочника), сумма штрафа;
- просмотр списка нарушений по автомобилю или владельцу;
- отметка о частичной или полной оплате штрафа;
- автоматическое удаление записи об автомобиле из активной базы при полной оплате всех штрафов;
- поиск по гос. номеру, ФИО владельца, дате нарушения;
- формирование отчётов.

Требования к надежности

- Обеспечение целостности данных при сбоях питания или аварийном завершении работы;
- Резервное копирование базы данных не реже одного раза в сутки;

Условия эксплуатации

- Работа в помещениях с температурой от +10 °C до +35 °C;
- Относительная влажность — до 80 %;
- Эксплуатация на стандартных АРМ инспектора ДПС;
- Обслуживание — 1 сотрудник (инспектор), прошедший краткий инструктаж.

Требования к составу и параметрам технических средств

- Компьютер: процессор не ниже Intel Core i3, ОЗУ ≥ 4 ГБ, HDD ≥ 128 ГБ;
- ОС: Windows 10/11 или Linux (Ubuntu 20.04+);
- Наличие сетевого подключения для синхронизации с центральной базой (опционально).

Требования к информационной и программной совместимости

- Язык программирования: Python 3.9+ или C# (.NET 6+);
- СУБД: SQLite
- Форматы обмена данными: JSON, CSV (для отчётов);
- Совместимость с существующими АИС ГИБДД (через API в будущем).

Технико-экономические показатели

- Сокращение времени оформления нарушения на 30 %;
- Снижение количества ошибок при учёте штрафов на 90 %;
- Годовая потребность: до 500 экземпляров (по числу АРМ в регионе);
- Экономический эффект: сокращение административных издержек на 15 % по сравнению с ручным учётом.

Стадии и этапы разработки

1. Анализ требований

- Сбор и формализация требований, согласование с заказчиком.
- Срок: 5 рабочих дней

2. Проектирование

- Разработка архитектуры, БД, интерфейсов
- Срок: 7 рабочих дней

3. Реализация

- Написание кода, модульное тестирование
- Срок: 15 рабочих дней

4. Тестирование

- Интеграционное и приёмочное тестирование
- Срок: 5 рабочих дней

5. Внедрение

- Установка, обучение пользователей
- Срок: 3 рабочих дней

Порядок контроля и приемки

Приёмка осуществляется комиссией УВД по результатам:

- демонстрации работоспособности всех функций;
- проверки соответствия требованиям настоящего ТЗ;
- успешного прохождения тестовых сценариев (не менее 20).
- Программа считается принятой после подписания акта приёмки.

Приложения

- КоАП РФ (ст. 12.1–12.37);
- Приказ МВД № 664 от 23.08.2022 «О порядке оформления нарушений ПДД».

4 практическая

Выработка требований к программному обеспечению и программному модулю

ЖЦ ПП, критерии качества ПП, виды ПО, стадии разработки ПП.

ЖЦ ПО - период от принятия решения о создании ПО до его снятия с эксплуатации.

Основные модели: Каскадная, Спиральная, Итеративная, Agile (Scrum, Kanban).

Критерии качества

- Функциональность, надежность, удобство использования, эффективность, сопровождаемость, портируемость.

Виды ПО:

- Системное (ОС, драйверы).
- Прикладное (офисные приложения, игры).
- Инструментальное (IDE, компиляторы).

Стадии разработки:

1. Анализ требований: сбор и детальное изучение потребностей заказчика и пользователей, определение целей системы, формирование функциональных и нефункциональных требований.
2. Проектирование: создание архитектуры системы, определение структуры модулей, интерфейсов, баз данных и алгоритмов.
3. Реализация (кодирование): написание программного кода в соответствии с проектной документацией/
4. Тестирование: проверка системы на соответствие требованиям, выявление и устранение ошибок.
5. Развертывание: установка и настройка системы в рабочей среде заказчика.
6. Сопровождение: поддержка системы после запуска в эксплуатацию.

Разработка требований (определение, виды работ).

Определение: Процесс сбора, анализа, документирования и управления ожиданиями заинтересованных лиц.

Виды работ:

- Сбор требований (интервью, анкетирование).
- Анализ и приоритизация.
- Документирование (спецификации).
- Валидация (проверка корректности).
- Управление изменениями.

Пользовательские требования, системные требования, проектная системная спецификация - определения.

Пользовательские (User Requirements):

- Описание функций системы на языке пользователя (например: «Система должна позволять оплачивать штрафы онлайн»).

Системные (System Requirements):

- Технические детали реализации (например: «API для интеграции с платежной системой Stripe»).

Проектная системная спецификация (Software Design Specification):

- Детальное описание архитектуры, компонентов, алгоритмов и интерфейсов ПО.

Виды требований к ПП.

1. Бизнес-требования (цели проекта).
2. Пользовательские требования.
3. Функциональные требования.
4. Нефункциональные требования.
5. Требования к интерфейсам.

Функциональные требования, нефункциональные требования - определения.

Функциональные:

- Описывают действия системы (например: «Регистрировать нарушения ПДД с указанием даты и штрафа»).

Нефункциональные:

- Описывают качество системы (производительность, безопасность, удобство).
    - Пример: «Система должна обрабатывать 1000 запросов в секунду».

Сравнение моделей разработки

№ п/п	Модель разработки	Особенности	+	-
1	SADT-модели	Структурный анализ и проектирование; функциональные блоки, стрелки (данные/управление)	Наглядность, четкое разделение функций, подходит для анализа бизнес-процессов	Не подходит для ООП, сложность масштабирования для крупных систем
2	CASE-средства	Автоматизация этапов разработки (моделирование, кодогенерация, документация)	Повышение производительности, снижение ошибок, интеграция с методами (SADT, UML)	Высокая стоимость, требует обучения, избыточность для малых проектов
3	Объектно-ориентированный анализ и проектирование	Акцент на классы, объекты, наследование, инкапсуляция, полиморфизм	Гибкость, повторное использование кода, легкость модификации	Сложность для новичков, риски перепроектирования, избыточная абстракция
4	UML	Стандартный язык моделирования с диаграммами (Use Case, классов, последовательностей)	Универсальность, поддержка ООП, визуализация архитектуры системы	Избыточность (не все диаграммы нужны), сложность изучения
5	Группа разработчиков	Распределение ролей (аналитики,	Разнообразие экспертиз,	Коммуникационные сложности,

		программисты, тестировщики), командная работа	параллельная разработка, снижение рисков	конфликты, необходимость координации

Вывод:

Я выбрал бы UML как основную модель для разработки. Это обусловлено его универсальностью и статусом де-факто стандарта в индустрии.

5 практическая

Проектирование ПО для решения прикладных задач Построение структуры программного продукта Кодирование программного обеспечения

Определения основных понятий

Проектирование ПО - процесс создания проекта программного обеспечения, включающий определение внутренних свойств системы и детализацию ее внешних (видимых) свойств на основе требований заказчика. Это процесс преобразования требований в архитектурную и детальную модель будущей системы.

Концептуальная архитектура - высокоуровневое представление системы, определяющее основные компоненты, их взаимодействие и ключевые принципы построения без детализации технических аспектов. Это "голубая печать" системы, показывающая общую логику организации.

Архитектурный стиль - набор принципов организации систем (или их компонентов) в рамках определенной архитектурной модели. Примеры: клиент-серверный, многослойный, сервис-ориентированный, микро сервисный. Стиль определяет структуру системы на высоком уровне абстракции.

Пилотная/базовая архитектура - первая работающая версия архитектуры системы, подтверждающая жизнеспособность выбранных подходов и технологий. Используется для проверки критически важных рисков и демонстрации технических возможностей системы.

Модуль - логически завершенная часть программы, реализующая определенную функциональность, имеющая четко определенные входы и выходы, и способная быть разработанной, протестированной и поддерживаемой независимо от других частей системы.

Компонент - самостоятельная, независимо развертываемая и заменяемая часть системы, инкапсулирующая реализацию и

предоставляющая функциональность через интерфейсы. Компоненты обычно имеют более высокий уровень абстракции, чем модули.

Фреймворк - программная платформа, определяющая каркас для построения приложений и предоставляющая стандартный способ решения определенных задач. Фреймворк использует принцип "инверсии управления" - разработчик пишет код, который вызывается фреймворком, а не наоборот.

Слабая связанность - принцип проектирования, при котором компоненты системы минимизируют зависимости между собой. Это позволяет модифицировать один компонент с минимальным влиянием на другие, упрощает тестирование и повторное использование кода.

Сквозная функциональность - функциональность, которая затрагивает несколько модулей или компонентов системы.

- **Примеры**: логирование, аутентификация, кэширование, обработка ошибок. Обычно реализуется через аспектно-ориентированное программирование или паттерны типа "цепочка обязанностей".

Портирование ПО - процесс адаптации программного обеспечения для работы в другой операционной системе, на другом аппаратном обеспечении или в ином программном окружении без значительного изменения исходного кода.

Программный код - текст, написанный на языке программирования, содержащий инструкции для компьютера. Код преобразуется компилятором или интерпретируется средой выполнения для получения исполняемой программы.

Структура кода - организация исходного кода в систему пакетов, модулей, классов и функций, отражающая архитектуру приложения. Хорошая структура кода способствует читаемости, поддерживаемости и расширяемости системы.

6 практическая

Определения терминов

Комплексное тестирование (system testing) - процесс тестирования всего программного продукта как единого целого для проверки соответствия функциональным и нефункциональным требованиям, выявления ошибок в работе интегрированной системы.

Отладка (debugging) - процесс локализации, анализа и устранения программных ошибок (багов) в коде для обеспечения корректной работы программы.

Тест - проверка определенного аспекта программного обеспечения, включающая выполнение программы с заданными входными данными и сравнение фактического результата с ожидаемым.

Верификация - процесс проверки соответствия результатов этапа разработки исходным требованиям этого этапа ("правильно ли мы делаем?").

Валидация - процесс подтверждения того, что продукт удовлетворяет требованиям пользователя и пригоден для использования ("делаем ли мы правильный продукт?").

- Этапы процесса тестирования:
- Планирование тестирования
- Анализ требований и проектирование тестовых случаев
- Разработка тестовой документации
- Выполнение тестов
- Анализ результатов и отчетность
- Закрытие тестирования

Цикл тестирования - итеративный процесс, включающий планирование, проектирование тестов, их выполнение, анализ результатов и подготовку отчетов, который повторяется при каждом внесении изменений в ПО.

Модульное тестирование - проверка отдельных компонентов (модулей, функций, классов) программы в изолированном режиме.

Интеграционное тестирование - проверка взаимодействия между различными модулями системы для выявления ошибок в их стыковке.

Системное тестирование - комплексное тестирование всей системы как единого целого для подтверждения соответствия требованиям.

Выходное тестирование (приемочное) - финальная проверка системы перед передачей заказчику для подтверждения ее готовности к эксплуатации.

Программная ошибка (баг) - несоответствие поведения программы ожидаемому или описанному в требованиях, приводящее к некорректным результатам.

Регрессионное тестирование - повторное тестирование уже проверенных функций после внесения изменений в код для убеждения, что новые изменения не нарушали существующую функциональность.

Тестирование "черного ящика" (black box) - метод тестирования, при котором проверяется функциональность системы без знания внутреннего устройства, основываясь только на спецификациях входов и выходов.

Тестирование "белого ящика" (white box) - метод тестирования, при котором проверка основывается на знании внутренней структуры кода, проверяются все пути выполнения и логические условия.

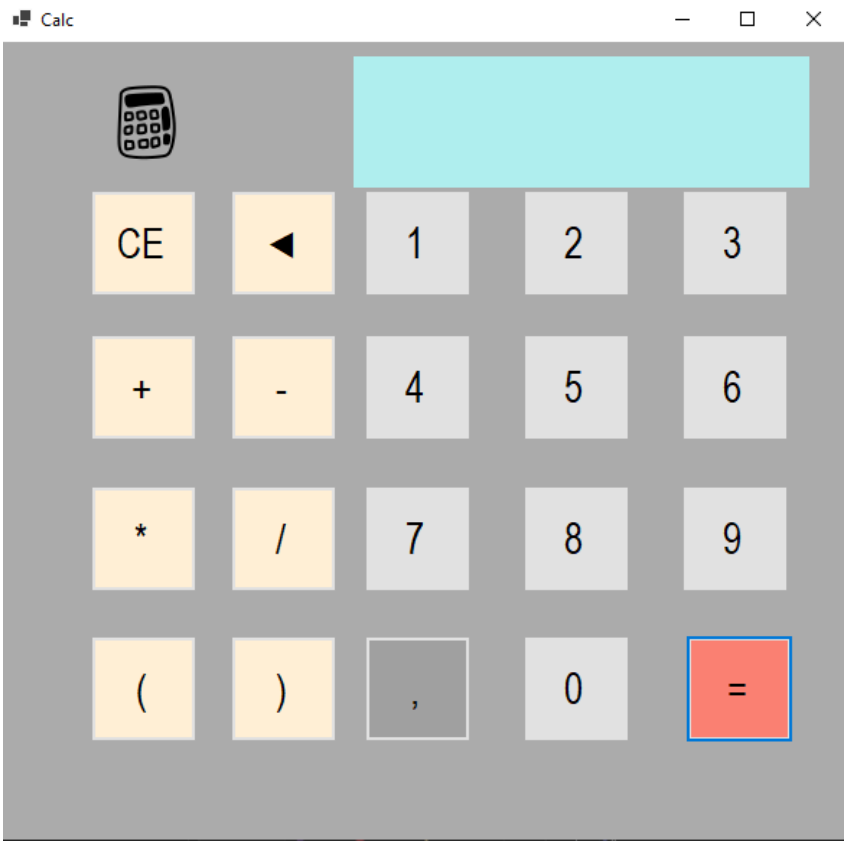
Трассировка - метод отладки, при котором фиксируется последовательность выполнения операторов программы для анализа ее поведения.

Тестовые сценарии - документированные последовательности действий, описывающие шаги для проверки конкретной функциональности системы с указанием входных данных, ожидаемых результатов и критериев успешного выполнения.

Три закона программотехники

1. Закон сохранения сложности: Сложность системы не может быть полностью устранена, она может только быть перераспределена между различными компонентами системы.

2. Закон демпфирования: Для обеспечения стабильности системы необходимо изолировать нестабильные компоненты от стабильных, создавая барьеры между зонами изменчивости и зонами стабильности.
3. Закон устойчивости конструкции: Надежность и качество системы определяются не количеством написанного кода, а качеством архитектурных решений и соблюдением принципов проектирования.



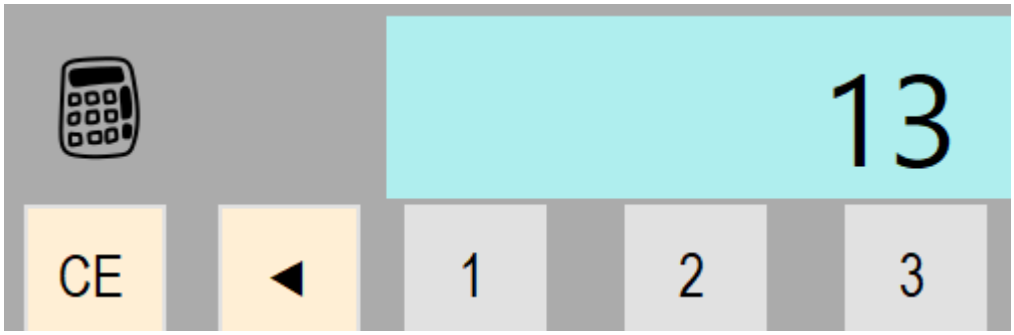
```
var dt = new DataTable();
var res = dt.Compute(label1.Text, null);
label1.Text = res.ToString();
```

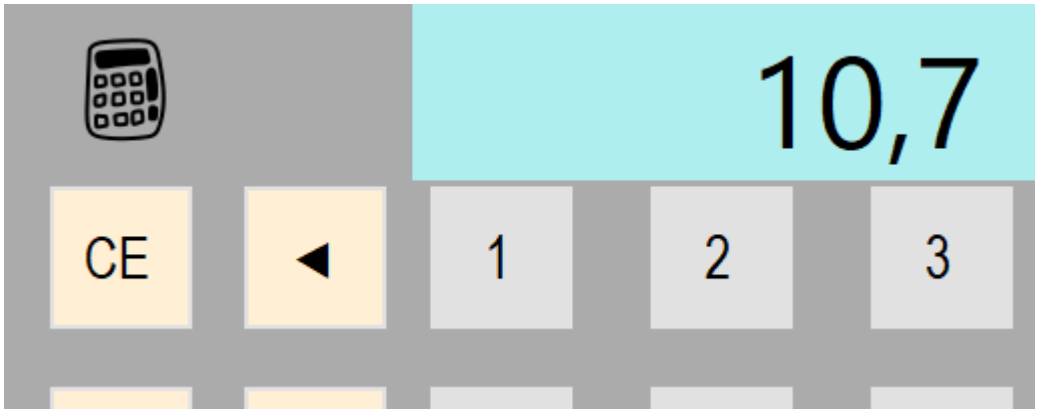
Тестовые данные		Операция	Ожидаемый результат. Что должно появиться	Выводимый результат. Что получилось
Значение A	Значение B			
10	3	+	13	13
5	7	-	-2	-2
8	0	/	Ошибка	∞
5.6	5.1	+	10.7	10.7
2	5+2	*	14	14
4	2	*	8	8
0	0	/	Недопустимая операция/ошибка программы	
4	5	/	0.8	0.8

Тесты

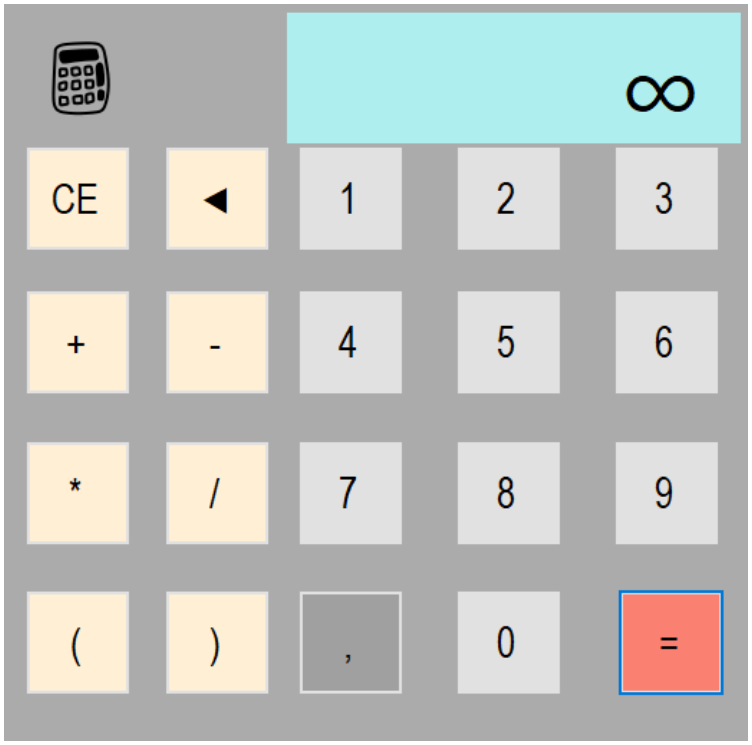
1. Тест 1 2 4 5 6

При вводе корректных данных программа работает правильно и выдаёт ожидаемые результаты

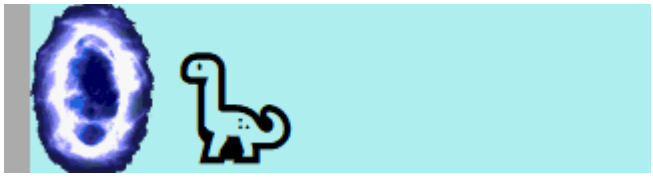




2. Тест 3



3. Тест 8



Сопровождение ПО (ИС)

Сопровождение ПО (информационной системы) - комплекс мероприятий, направленных на обеспечение работоспособности, исправление ошибок и внесение улучшений в программное обеспечение после его передачи в эксплуатацию. Сопровождение является финальным, но самым длительным этапом жизненного цикла ПО.

Варианты сопровождения:

1. Корректирующее сопровождение - деятельность по выявлению и исправлению ошибок, недостатков и дефектов, обнаруженных в процессе эксплуатации системы после ее передачи в производство.
2. Адаптивное сопровождение - модификация программного обеспечения для адаптации к изменениям во внешней среде (новые версии операционных систем, изменения в аппаратном обеспечении, изменения законодательства, интеграция с новыми системами).
3. Совершенствующее сопровождение - улучшение функциональных характеристик системы, добавление новых функций и возможностей, оптимизация производительности, улучшение пользовательского интерфейса в соответствии с меняющимися потребностями пользователей.
4. Профилактическое сопровождение - действия, направленные на повышение надежности, производительности и поддерживаемости системы без изменения ее функциональности (рефакторинг кода, обновление документации, оптимизация структуры базы данных).

7 практическая

Выполнение адаптации программного продукта к условиям функционированият Коллективная разработка программного обеспечения Администрирование программного обеспечения

Определения терминов

1. **Организационная структура**

- Система взаимосвязанных элементов (подразделений, должностей, специалистов), которые образуют определенную иерархию и распределение ответственности для достижения целей организации. Это «скелет» организации, определяющий официальные отношения между ее частями.

2. **Структура управления**

- Совокупность взаимосвязанных элементов системы управления (органов, подразделений, должностей) и их взаимодействие в процессе управления организацией или проектом. Включает распределение функций, прав и ответственности между участниками процесса управления.

3. **Элемент организационной структуры**

- Отдельное подразделение, должность или специалист, выполняющий определенные функции в рамках организационной структуры. Элементы могут быть как самостоятельными единицами (отдел, сотрудник), так и составными частями более крупных элементов.

4. **Уровни** (ступени управления)

- Иерархические ступени в организационной структуре, на которых осуществляются управленческие решения. Обычно выделяют три уровня: высший (стратегический), средний (тактический) и низший (оперативный).

5. **Регламентирование**

- Процесс установления правил, процедур и стандартов деятельности организации или проекта, закрепленных в официальных документах. Регламентирование обеспечивает упорядоченность, предсказуемость и контролируемость процессов.

6. **Нормирование**

- Установление количественных и качественных показателей (норм) деятельности организации, подразделений или сотрудников. Нормирование включает определение сроков выполнения задач,

ресурсных затрат, показателей качества и других измеримых параметров.

7. **Инструктирование**

- Процесс разработки, утверждения и доведения до исполнителей инструкций, регламентирующих порядок выполнения конкретных операций или задач. Инструктирование направлено на обеспечение единообразия и качества выполнения работ.

8. **Делегирование**

- Передача части полномочий и ответственности от руководителя подчиненному для выполнения определенных задач. Эффективное делегирование включает четкое определение задач, предоставление необходимых ресурсов и сохранение общей ответственности за результат.

9. **Полномочия**

- Права, предоставленные должностному лицу или подразделению для принятия решений и распоряжений в рамках их компетенции. Полномочия определяют границы возможного воздействия сотрудника на процессы и результаты деятельности.

10. **Ответственность**

- Обязанность лица или подразделения отвечать за результаты своей деятельности, выполнение возложенных функций и соблюдение установленных требований. Ответственность включает необходимость отчитываться о результатах и нести последствия за невыполнение или ненадлежащее выполнение обязанностей.

8 прктическая

1. Введение

Назначение документа

Руководство содержит инструкции по использованию маркетплейса AliExpress (веб-версия и мобильное приложение), включая регистрацию, поиск товаров, оформление заказов, взаимодействие с поддержкой.

Целевая аудитория

Конечные пользователи, не имеющие опыта работы с AliExpress.

2. Регистрация и вход в аккаунт

2.1. Создание аккаунта

Перейдите на сайт aliexpress.com или откройте мобильное приложение.

Нажмите «Зарегистрироваться».

Укажите email или номер телефона, придумайте пароль.

Подтвердите данные через SMS или email.

2.2. Вход в существующий аккаунт

Нажмите «Войти» в правом верхнем углу.

Введите email/телефон и пароль.

3. Основные функции платформы

3.1. Поиск товаров

В строке поиска введите название товара (например, «беспроводные наушники»).

Используйте фильтры:

Цена («от **1000 до 3000** руб.»).

Рейтинг продавца (4.8+).

Доставка («Бесплатная»).

Пример: Для выбора качественного товара отсортируйте результаты по «Лучшие продажи».

3.2. Оформление заказа

- Выберите товар и нажмите «Купить сейчас».
- Укажите адрес доставки.

- Выберите способ доставки (стандартная/экспресс).
- Добавьте комментарии для продавца (по желанию).

3.3. Оплата

Поддерживаемые способы:

- Банковские карты (Visa, Mastercard).
- Электронные кошельки (PayPal, WebMoney).
- AliPay.
- Важно: Никогда не передавайте данные карты третьим лицам.

3.4. Отслеживание заказа

- Перейдите в раздел «Мои заказы».
- Нажмите «Отследить» напротив заказа.
- Просмотрите статус: «Обработка», «Отправлен», «Доставлен».

4. Безопасность и защита данных

- Двухфакторная аутентификация: Активируйте в настройках аккаунта.
- Подозрительные предложения: Избегайте сделок вне платформы AliExpress.
- Конфиденциальность: Не сохраняйте пароли в браузере.

5. Возврат товара и поддержка

5.1. Возврат

- Откройте заказ в разделе «Мои заказы».
- Нажмите «Вернуть товар» в течение 15 дней после получения.
- Укажите причину и загрузите фото товара.

5.2. Контакты поддержки

- Онлайн-чат: Доступен в приложении (значок «Поддержка»).
- Email: support@aliexpress.com .
- База знаний: help.aliexpress.com .

6. Мобильное приложение

Особенности:

- Push-уведомления о скидках.
- Сканер штрих-кодов для поиска товаров.
- Упрощенная оплата через Face ID/Touch ID.
- Скачать:

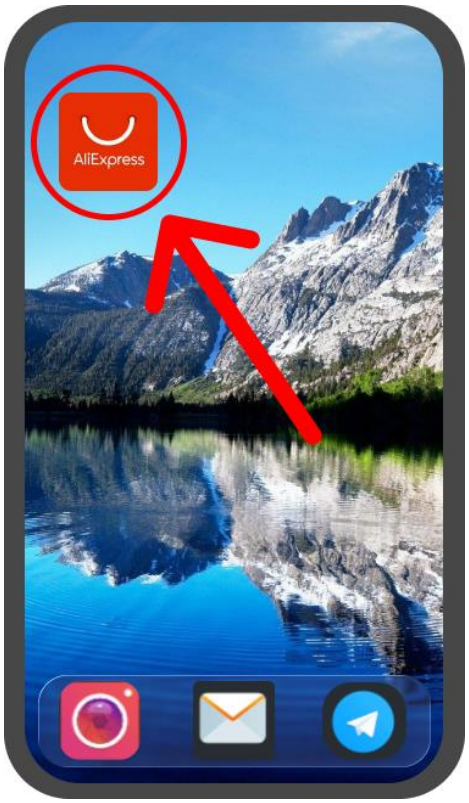
- App Store
- Google Play

7. Часто задаваемые вопросы (FAQ)

- Вопрос: Как отменить заказ?
    - Ответ: Откройте заказ в «Мои заказы» → Нажмите «Отменить», если статус «Ожидает отправки».
- Вопрос: Почему заказ задерживается?
    - Ответ: Проверьте статус в разделе отслеживания. При задержке более 7 дней обратитесь в поддержку.

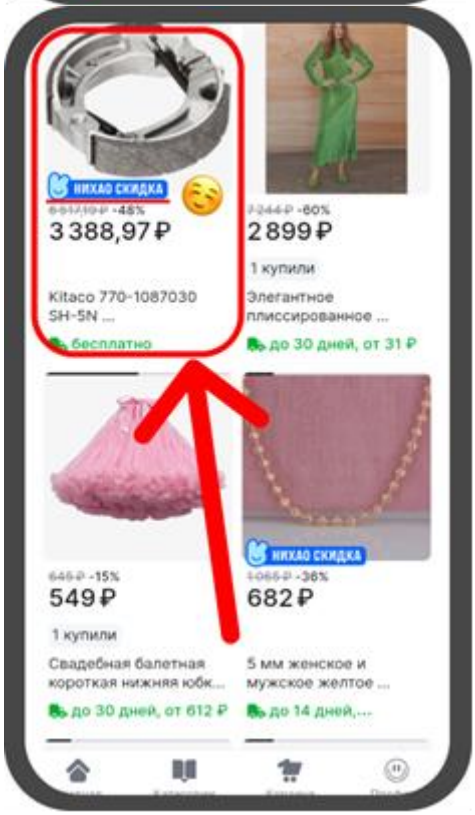
8. Глоссарий

- Продавец с рейтингом 98%: 98% покупателей оценили сотрудничество положительно.
- Выборочная доставка: Товары отправляются с разных складов.

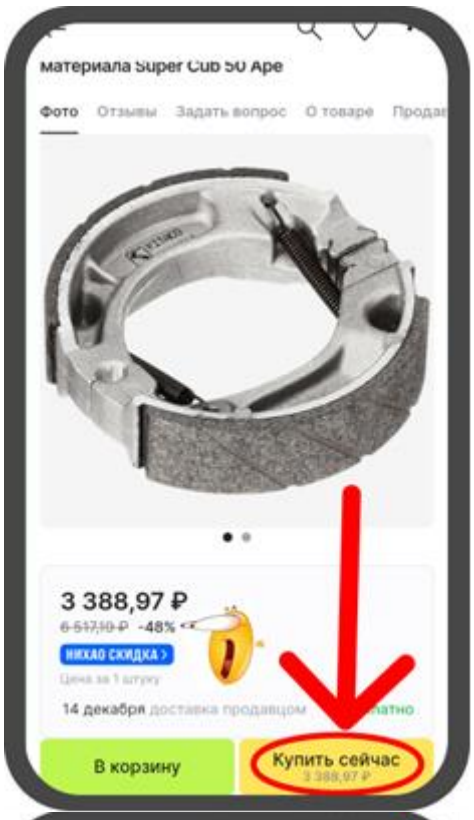


1.Открыть приложение

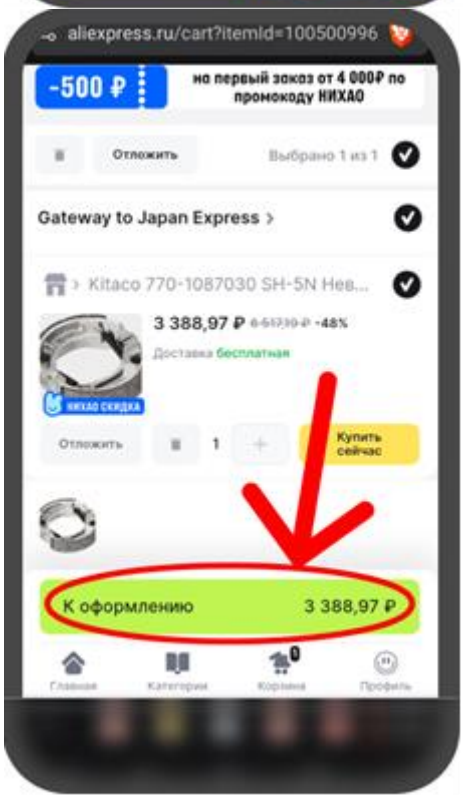
2.Найти нужный товар через поиск/или посмотреть ассортимент



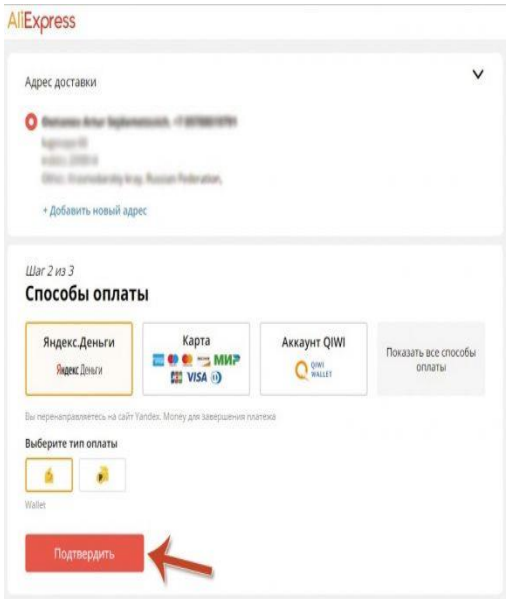
3.Выбрать подходящий товар



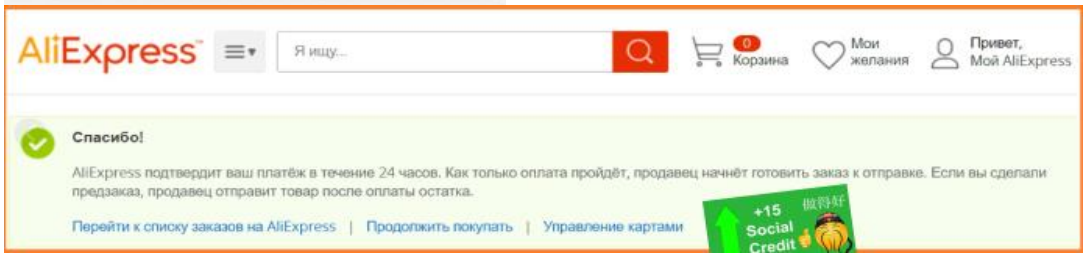
4.Добавить товар в корзину или приобрести сразу



5.Перейти к оплате товара



6.Выбрать способ оплаты и оплатить следуя инструкциям



7.Ждать заказ на ПВЗ

9 практическая

Сертификация и лицензирование программного продукта

1. Определения терминов

1.1. Сертификация услуг (работ)

Сертификация услуг (работ) процедура подтверждения соответствия услуг или работ установленным требованиям, проводимая независимой третьей стороной (сертификационным органом). Результатом является выдача сертификата, подтверждающего соответствие услуг определенным стандартам качества, безопасности или другим критериям. В контексте ИТ-услуг сертификация может касаться качества разработки ПО, безопасности информационных систем, соответствия стандартам ISO/IEC 27001, CMMI и другим.

1.2. Лицензия на ПО

Лицензия на ПО правовой документ, определяющий условия использования программного обеспечения, права и обязанности пользователя и правообладателя. Лицензия устанавливает:

- Количество установок программы
- Срок использования
- Права на модификацию и распространение
- Гарантии и ограничения ответственности
- Стоимость и порядок оплаты

Лицензия не передает права собственности на ПО, а лишь предоставляет право на его использование в оговоренных рамках.

1.3. Исключительные и Неисключительные права

- Исключительные права права, принадлежащие единственному правообладателю, который вправе использовать объект интеллектуальной собственности по своему усмотрению и разрешать или запрещать такое использование другим лицам (ст. 1229 ГК РФ). Правообладатель может передать исключительные права полностью или частично по лицензионному договору.
- Неисключительные права право использования объекта интеллектуальной собственности, которое сохраняется за

правообладателем и может быть предоставлено одновременно нескольким лицам. Пользователь не может передавать эти права третьим лицам без согласия правообладателя.

1.4. GNU GPL и FreeBSD

- GNU GPL (General Public License) свободная лицензия на программное обеспечение, разработанная Фондом свободного программного обеспечения (FSF). Основные принципы:
- Свобода запуска программы для любых целей
- Свобода изучения и модификации исходного кода
- Свобода распространения копий
- Свобода распространения модифицированных версий

Особенность GPL требование "вирусности": производные работы также должны распространяться под лицензией GPL.

- FreeBSD License пермиссивная открытая лицензия с минимальными ограничениями. Позволяет:
- Свободное использование, модификацию и распространение ПО
- Использование в коммерческих продуктах без обязательного открытия исходного кода
- Отсутствие требования "вирусности"

Основное требование сохранение авторских прав и лицензионного уведомления в исходном коде.

1.5. Классификация ПО по лицензированию

- Бесплатное ПО (Freeware) программы, предоставляемые бесплатно без предоставления исходного кода. Автор сохраняет авторские права и может ограничивать модификацию и распространение. Пример: Skype, Adobe Reader.
- Условно-бесплатное ПО (Shareware) программы, предоставляемые для бесплатного пробного использования с ограничениями (по времени, функциональности), после чего требуется оплата для получения полной версии. Пример: WinRAR (пробный период 40 дней).
- Коммерческое ПО программы, распространяемые на платной основе с закрытым исходным кодом. Требуется покупка лицензии для законного использования. Пример: Microsoft Office, Adobe Photoshop.
- OEM/BOX-версии:

- OEM (Original Equipment Manufacturer) версии ПО, поставляемые производителям компьютеров для предустановки на оборудование. Обычно привязаны к конкретному оборудованию и не могут быть перенесены на другое устройство.
- BOX-версия розничная версия ПО в коробке с лицензионным ключом, которая может быть установлена на любой компьютер в пределах количества лицензий.

1.6. Ответственность (Законодательство РФ)

Согласно законодательству РФ, ответственность за нарушение авторских прав на ПО регулируется:

- Гражданским кодексом РФ (часть IV, раздел VII) предусматривает возмещение убытков и компенсацию в размере от 10 тыс. до 5 млн руб. за незаконное использование ПО.
- Кодексом об административных правонарушениях (ст. 7.12) административная ответственность в виде штрафа от 1.5 тыс. до 40 тыс. руб. для граждан и до 200 тыс. руб. для юридических лиц.
- Уголовным кодексом РФ (ст. 146) уголовная ответственность за нарушение авторских прав в крупном размере (до 200 тыс. руб. или лишение свободы до 2 лет) и в особо крупном размере (до 500 тыс. руб. или лишение свободы до 6 лет).

2. Методы оценки затрат на создание ПО

2.1. Методы оценки затрат

На основе анализа предоставленных материалов можно выделить следующие методы оценки затрат на разработку ПО:

1. Экспертный метод (Delphi)

- Особенности расчета: Оценка основывается на мнении экспертов-разработчиков с последующей итеративной корректировкой с учетом обратной связи. Эксперты дают независимые оценки, которые затем агрегируются (обычно медиана или среднее значение).
- Преимущества: Учитывает специфику проекта и опыт экспертов.
- Недостатки: Субъективность, зависит от квалификации экспертов.
- Пример: Для оценки разработки мобильного приложения для интернет-магазина 3 эксперта дали оценки: 250, 300 и 350 человеко-часов. После обсуждения согласована оценка в 280 человеко-часов.

2. Параметрический метод (Function Point Analysis)

- Особенности расчета: Оценка основывается на измеримых параметрах функциональности ПО (число функциональных точек, строк кода, количество экранов, отчетов, интеграций). Для каждого параметра определяются весовые коэффициенты и сложность.
- Преимущества: Объективность, возможность автоматизации расчетов.
- Недостатки: Требует исторических данных для калибровки коэффициентов.
- Пример: Система учета имеет 15 входных экранов (вес 4 каждое), 10 отчетов (вес 7 каждое), 5 внешних интерфейсов (вес 10 каждое). Общее число функциональных точек = 15×4 + 10×7 + 5×10 = 180 FP. При стоимости 1 FP = 1000 руб., общая стоимость = 180,000 руб.

3. Сравнительный метод (аналогов)

- Особенности расчета: Оценка на основе сравнения с аналогичными проектами, выполненными ранее. Учитывается масштаб, сложность, технологический стек.
- Преимущества: Простота, наглядность.
- Недостатки: Требует базы данных по аналогичным проектам, сложно найти точные аналоги.
- Пример: Разработка CRM-системы оценивается как 120% от стоимости реализованного ранее проекта учета клиентов (500,000 руб.), с учетом дополнительной функциональности по аналитике продаж.

2.2. Основные виды затрат

1. Трудовые затраты основная статья расходов, включает:

- Зарплаты разработчиков, тестировщиков, аналитиков
- Налоги и страховые взносы
- Обучение и профессиональное развитие

2. Аппаратные затраты:

- Серверное оборудование
- Компьютеры для разработчиков
- Сетевое оборудование
- Устройства для тестирования (мобильные устройства)

3. Программные затраты:

- Лицензии на IDE и инструменты разработки

- Системное ПО и базы данных
- Инструменты тестирования и мониторинга
- Облачные сервисы и хостинг

4. Накладные расходы:

- Аренда офиса
- Коммунальные платежи
- Административный персонал
- Маркетинг и продвижение

5. Затраты на сопровождение:

- Техническая поддержка
- Обновления и доработки
- Резервное копирование и восстановление
- Безопасность и защита данных

3. Пояснительная записка (ГОСТ)

Согласно ГОСТ 19.404-79 "Пояснительная записка. Требования к содержанию и оформлению", пояснительная записка должна содержать следующие разделы:

1. Введение

- Основание для разработки (договор, приказ, задание)
- Назначение системы (область применения, цели создания)
- Краткое описание предметной области
- Источники разработки (использованные стандарты, нормативные документы)

2. Основная часть

2.1. Описание предметной области

- Бизнес-процессы, которые автоматизируются
- Участники процессов (роли пользователей)
- Входные и выходные документы
- Требования к безопасности и защите данных

2.2. Требования к системе

- Функциональные требования (что система должна делать)

- Нефункциональные требования (производительность, надежность, масштабируемость)
- Требования к пользовательскому интерфейсу
- Требования к интеграции с внешними системами

2.3. Архитектура системы

- Высокоуровневая архитектурная схема
- Описание выбранных технологий и платформ
- Структура базы данных (ER-диаграммы)
- Описание модулей системы и их взаимодействие

2.4. Реализация

- Описание ключевых алгоритмов
- Примененные паттерны проектирования
- Описание сложных технических решений
- Скриншоты ключевых интерфейсов системы

2.5. Тестирование

- Стратегия тестирования
- Описание тестовых сценариев
- Результаты тестирования (таблицы выявленных ошибок и их устранения)
- Методы обеспечения качества

3. Заключение

- Оценка соответствия разработанной системы требованиям
- Перечень решенных задач
- Рекомендации по внедрению
- Планы по дальнейшему развитию системы

4. Приложения

- Листинги программного кода (ключевые модули)
- Диаграммы (UML, DFD, ERD)
- Формы отчетов и документы
- Инструкции по установке и настройке
- Список использованных источников и литературы

5. Титульный лист

- Название организации

- Название проекта
- Авторы (исполнители)
- Руководители проекта
- Год выполнения

Требования к оформлению:

- Страницы должны иметь сквозную нумерацию
- Текст оформляется шрифтом 14 пт с полуторным интервалом
- Абзацный отступ 1.25 см
- Поля: левое 30 мм, правое 10 мм, верхнее и нижнее 20 мм
- Диаграммы и таблицы должны иметь сквозную нумерацию и подписи
- Ссылки на источники оформляются в квадратных скобках

Такая структура пояснительной записки обеспечивает полное и систематизированное описание программного продукта, его архитектуры, реализации и результатов тестирования, что соответствует требованиям ГОСТ и позволяет эффективно оценить качество выполненной работы.