Государственное автономное профессиональное образовательное учреждение Самарской области

«Новокуйбышевский нефтехимический техникум»

Куразеева Снежана Валерьевна

Рабочая тетрадь по написанию НТТ по дисциплине «Информатика»

Новокуйбышевск, 2024

**Аннотация**

Рабочая тетрадь по научно-техническому творчеству предназначена для студентов, стремящихся развить свои навыки в области информатики и инженерии. Тетрадь охватывает широкий спектр тем, включая основы программирования, алгоритмизацию, проектирование и конструирование технических устройств.

В процессе работы над заданиями студенты будут изучать современные технологии и методы, видеть, как теоретические знания применяются на практике. Каждое занятие включает практические задачи, экспериментальные проекты и творческие задания, которые способствуют развитию критического мышления, креативности и командной работы.

С помощью этой тетради студенты смогут не только закрепить полученные знания, но и научиться применять их в реальных проектах, что является важным опытом для будущей профессиональной деятельности в сфере науки и техники. Тетрадь будет полезна как для самостоятельной работы, так и для занятий в группе. Полезные ссылки, дополнительные материалы и примеры успешных проектов сделают процесс обучения увлекательным и продуктивным.

Надеемся, что эта рабочая тетрадь станет верным помощником на пути к открытию новых горизонтов в научно-техническом творчестве!

Сведения об авторе: Куразеева Снежана Валерьевна, преподаватель ГАПОУ СО «ННХТ»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

В условиях стремительного развития информационных технологий и их внедрения в различные сферы деятельности, формирование у студентов средних профессиональных образовательных учреждений (СПО) навыков научно-технического творчества становится особенно актуальным. Современные специалисты должны уметь не только использовать уже существующие технологии, но и создавать новые решения, проводить исследовательскую работу и презентовать свои идеи. Разработка рабочей тетради по написанию научно-технического творчества даст возможность студентам осваивать эти важные компетенции, наращивая потенциал для успешной профессиональной деятельности в будущем.

Цель данной рабочей тетради – предоставление студентам структурированного и доступного пособия, которое облегчит процесс изучения и применения методов научно-технического творчества. Мы стремимся помочь студентам разобраться в основных этапах разработки научно-технических проектов, от идеи до реализации, а также дать практические рекомендации по написанию проектов и защите их результатов.

Ожидаемым результатом использования данной рабочей тетради является формирование навыков, необходимых для успешного выполнения научно-технических проектов. Студенты научатся:

- Определять актуальную тему и формулировать проблему для исследования.

- Проводить анализ существующих решений и обосновывать выбор собственного подхода.

- Разрабатывать проект, включая этапы планирования, разработки и тестирования.

- Оформлять результаты своей работы в виде научного отчета или презентации.

Кроме этого, студенты смогут повысить уровень своей критической оценки работы как своей, так и других, что является важным аспектом научно-технического творчества.

Предлагаемая рабочая тетрадь выделяется среди существующих методических материалов своей практической ориентированностью и интеграцией современных технологий. В отличие от традиционных пособий, которые часто фокусируются на теоретических аспектах, наша тетрадь включает в себя разнообразные практические задания, которые вдохновляют студентов на творчество и инновации.

Кроме того, в рабочей тетради предусмотрены уникальные разделы, посвященные актуальным трендам в ИТ-индустрии, таким как искусственный интеллект, машинное обучение и интернет вещей, что делает ее содержание современным и востребованным. Студенты смогут работать над проектами, отражающими реальные потребности и задачи, стоящие перед обществом, что повысит их мотивацию и заинтересованность.

Таким образом, данная рабочая тетрадь не только способствует развитию навыков научно-технического творчества у студентов СПО, но и представляет собой актуальный инструмент для подготовки компетентных специалистов, способных успешно адаптироваться к вызовам современного мира.

**СОДЕРЖАНИЕ**

Введение………………………………………………………………………….7

Глава 1: Основы программирования……………………………………………8

- 1.1. Введение в языки программирования…………………………………..8

- 1.2. Алгоритмы и структуры данных……………………………………….15

- 1.3. Практические задачи по программированию………………………….26

Глава 2: Проектирование и разработка…………………………………………30

- 2.1. Процесс разработки программного обеспечения………………………30

- 2.2. Работа с системами управления версиями……………………………...37

- 2.3. Командные проекты……………………………………………………..47

Глава 3: Основы компьютерной системы и сети………………………………49

- 3.1. Структура компьютера………………………………………………….49

- 3.2. Основы сетевых технологий……………………………………………57

- 3.3. Безопасность информации………………………………………………64

Глава 4: Инновации и современные технологии………………………………70

- 4.1. Искусственный интеллект и машинное обучение……………………..70

- 4.2. Интернет вещей (IoT)……………………………………………………79

- 4.3. Будущее технологий……………………………………………………..85

Глава 5: Научно-исследовательская работа…………………………………….91

- 5.1. Основы научного метода………………………………………………..91

- 5.2. Методология проведения исследования……………………………….93

- 5.3. Обработка данных и представление результатов……………………..96

Глава 6: Творческие проекты…………………………………………………..100

- 6.1. Генерация идей и их реализация………………………………………101

- 6.2. Презентация проектов………………………………………………….111

Заключение……………………………………………………………………...117

Рефлексия……………………………………………………………………….119

Приложения…………………………………………………………………….122

**ВВЕДЕНИЕ**

Добро пожаловать в рабочую тетрадь по научно-техническому творчеству, посвященную дисциплине информатики! Эта тетрадь разработана с целью помочь вам погрузиться в мир информации и технологий, раскрыть ваш творческий потенциал и научить вас применению научных знаний на практике.

Основная цель данной тетради — развитие у студентов навыков критического мышления, креативности и командной работы через выполнение практических заданий. На протяжении курса вы будете работать с разнообразными темами, такими как:

- Основы программирования: изучение языков программирования, алгоритмов и структур данных.

- Проектирование и разработка: освоение этапов создания программного обеспечения и управления проектами.

- Основы компьютерной системы: понимание архитектуры компьютера и принципов функционирования сетевых технологий.

- Современные технологии: погружение в темы искусственного интеллекта, интернета вещей и новых трендов в информатике.

- Научно-исследовательская работа: применение научных методов для решения задач и представление своих результатов.

- Творческие проекты: реализация собственных идей и проектов в области научно-технического творчества.

Работа в тетради будет организована по главам, каждая из которых представит определенную тему. Каждая глава включает в себя:

- Теоретический материал, объясняющий основные концепции и принципы.

- Практические задания, которые помогут вам закрепить полученные знания и применить их в действии.

- Примеры и рекомендации, которые дадут вам дополнительные идеи для ваших собственных проектов.

Задания варьируются от индивидуальных до групповых, что позволит вам как разрабатывать идеи в одиночку, так и работать в команде, обсуждая и реализуя творческие решения.

Для успешного выполнения заданий следуйте следующим рекомендациям:

1. Планируйте свое время. Убедитесь, что у вас достаточно времени для выполнения заданий, особенно для более сложных проектов.

2. Ссылки на ресурсы. Используйте предложенные в тетради ресурсы и ссылки для углубленного изучения тем. Они помогут вам лучше понять материал и расширить свои знания.

3. Работа в команде. Делитесь своими идеями и обсуждайте их с однокурсниками. Командная работа приносит новые идеи и подходы к решению задач.

4. Записывайте свои мысли. Используйте раздел для комментариев и записей, чтобы фиксировать свои идеи и размышления по выполненным заданиям.

Этот курс не только предоставит вам знания и навыки, но также вдохновит на создание новых идей и проектов в области научно-технического творчества. Мы верим, что ваша работа в этой тетради станет важным шагом на пути к освоению информатики и смежных дисциплин.

Приятной работы и удачи в ваших научных и технических начинаниях!

**Глава 1: Основы программирования**

**1.1. Обзор популярных языков программирования, их специфики и применения**

Программирование стало основой многих современных технологий. Разные языки программирования обладают уникальными характеристиками и применяются в различных областях. В этой теме мы рассмотрим несколько наиболее популярных языков программирования, их спецификации и области применения.

1. Python

- Специфика: Python является высокоуровневым языком программирования с простой и читаемой синтаксисом, что делает его идеальным для начинающих. Он поддерживает множество парадигм программирования, таких как объектно-ориентированное, функциональное и процедурное программирование.

- Применение:

- Разработка веб-приложений (Django, Flask)

- Научные вычисления и анализ данных (Pandas, NumPy, SciPy)

- Искусственный интеллект и машинное обучение (TensorFlow, Keras)

- Автоматизация и сценарное программирование

2. Java

- Специфика: Java — это объектно-ориентированный язык программирования, который работает на платформе Java Virtual Machine (JVM), что делает его кроссплатформенным. Он предлагает строгую типизацию данных и мощные библиотеки.

- Применение:

- Разработка корпоративных приложений

- Мобильные приложения для Android

- Веб-приложения (Spring Framework)

- Разработка игр и программного обеспечения для встраиваемых систем

3. C++

- Специфика: C++ — это расширение языка C, которое добавляет поддержку объектно-ориентированного программирования. Он предоставляет высокую производительность и контроль над ресурсами системы.

- Применение:

- Системное программирование (операционные системы, драйверы)

- Игровая разработка (наиболее известные игровые движки, такие как Unreal Engine)

- Научные и численные вычисления

- Встраиваемые системы и IoT

4. JavaScript

- Специфика: JavaScript — это язык сценариев, который изначально был разработан для взаимодействия с веб-страницами. Он является основным языком для фронтенд-разработки и поддерживает асинхронное программирование.

- Применение:

- Веб-разработка (разработка пользовательских интерфейсов и динамических страниц)

- Серверная разработка с использованием Node.js

- Разработка мобильных приложений (с помощью фреймворков таких как React Native)

- Приложения для Интернета вещей

5. C#

- Специфика: C# — это язык, разработанный Microsoft, который используется в основном для разработки приложений в экосистеме Windows. Он поддерживает объектно-ориентированное программирование и предоставляет современную синтаксис.

- Применение:

- Разработка программного обеспечения на платформе .NET

- Разработка игр с использованием Unity

- Веб-приложения (ASP.NET)

- Приложения для Windows и мобильных устройств

6. Ruby

- Специфика: Ruby — это высокоуровневый, интерпретируемый язык, который фокусируется на производительности и гибкости. Одна из его сильных сторон – это простота и выразительность кода.

- Применение:

- Веб-разработка (Ruby on Rails)

- Программирование для прототипирования

- Разработка приложений в стартапах

7. Go (Golang)

- Специфика: Go — это язык программирования, разработанный Google, который сочетает в себе производительность C и простоту Python. Он имеет встроенные возможности для параллельного программирования.

- Применение:

- Разработка высокопроизводительных серверных приложений

- Разработка облачных сервисов и контейнеризации (Docker)

- Системные инструменты и утилиты

8. Swift

- Специфика: Swift — язык, созданный Apple для разработки программного обеспечения для iOS и macOS. Он заменил Objective-C и предлагает более простой и безопасный синтаксис.

- Применение:

- Разработка мобильных приложений для iPhone и iPad

- Разработка настольных приложений для macOS

- Программирование для устройств Apple (Apple Watch, Apple TV)

Каждый язык программирования имеет свои сильные и слабые стороны, а также предоставляет различные возможности для разработки. Выбор языка зависит от конкретных задач, требований проекта и личных предпочтений разработчика. Понимание специфики и применения разных языков программирования поможет вам сделать осознанный выбор в какой области вы хотели бы работать и каким технологиям уделять внимание в своем обучении и карьере.

***Контрольные вопросы***

На общие вопросы отвечают все, по основным языкам программирования по выбору.

1. Общие вопросы:

1. Какие основные преимущества использования языков программирования в современном мире?

2. Каковы критерии выбора языка программирования для конкретного проекта?

3. Какие факторы влияют на популярность языков программирования?

**2. Python:**

1. Почему Python считается идеальным языком для начинающих программистов?

2. Назовите три области, в которых Python широко используется. Приведите примеры библиотек или фреймворков, связанных с этими областями.

3. Каковы плюсы и минусы Python по сравнению с другими языками программирования?

**3. Java:**

1. В чем заключается кроссплатформенность Java и как она достигается?

2. Для каких типов приложений чаще всего используется Java? Приведите примеры.

3. Объясните, как Java управляет памятью и как это влияет на производительность приложений.

**4. C++:**

1. Как C++ сочетает в себе функциональность языков C и объектно-ориентированного программирования?

2. В каких областях, по вашему мнению, C++ наиболее эффективен и почему?

3. Какие особенности языка C++ могут представлять сложности для новых программистов?

**5. JavaScript:**

1. Какова основная роль JavaScript в веб-разработке?

2. Объясните, как JavaScript используется на стороне сервера и как это связано с Node.js.

3. Какие популярные библиотеки или фреймворки на базе JavaScript вы знаете и для чего они используются?

**6. C#:**

1. В чем специфичность C# как языка программирования, разработанного Microsoft?

2. Какие приложения обычно создаются с использованием C# и фреймворка .NET?

3. Как C# поддерживает объектно-ориентированное программирование и какие преимущества это дает?

**7. Ruby:**

1. Почему Ruby получил популярность среди веб-разработчиков? Какой фреймворк ассоциируется с этим языком?

2. В чем заключается философия Ruby, и как это влияет на его синтаксис?

3. Как Ruby подходит для стартапов и быстрой разработки прототипов?

**8. Go (Golang):**

1. Какие особенности языка Go делают его подходящим для разработки облачных приложений?

2. Как Go управляет параллельными процессами и какие преимущества это предоставляет разработчикам?

3. Почему язык Go становится все более популярным в таких компаниях, как Google?

**9. Swift:**

1. Какие изменения в разработке iOS-приложений привнес язык Swift по сравнению с Objective-C?

2. В каких областях, кроме мобильной разработки, может использоваться Swift?

3. Какие ключевые характеристики делают Swift безопасным языком для разработки?

*Список источников для самостоятельного ознакомления*

*Онлайн-ресурсы и документация:*

1. [Python Official Documentation](https://docs.python.org/3/) — официальная документация Python.

2. [Java Documentation](https://docs.oracle.com/javase/8/docs/) — официальная документация по Java.

3. [C++ Reference](http://www.cplusplus.com/) — справочник по C++.

4. [MDN Web Docs on JavaScript](https://developer.mozilla.org/en-US/docs/Web/JavaScript) — руководство и документация по JavaScript от Mozilla.

5. [Microsoft Docs for C#](https://docs.microsoft.com/en-us/dotnet/csharp/) — официальная документация по C#.

6. [Ruby Documention](https://www.ruby-lang.org/en/documentation/) — документация по языку Ruby.

7. [Go Programming Language Documentation](https://golang.org/doc/) — официальная документация по Go.

8. [Swift Documentation](https://developer.apple.com/documentation/swift) — официальный ресурс по языку Swift от Apple.

*Онлайн-курсы и видео-лекции:*

1. [Coursera](https://www.coursera.org/) — платформа с множеством курсов по программированию на различных языках.

2. [edX](https://www.edx.org/) — предлагает курсы от университетов, включая курсы по Python, Java, C#, и другим языкам.

3. [Udemy](https://www.udemy.com/) — платформа с курсами по программированию, включая специализированные курсы по Ruby, Go, Swift и другим языкам.

4. [YouTube - TheNetNinja](https://www.youtube.com/user/TheNetNinja) — множество обучающих видео по JavaScript, Python, и другим языкам программирования.

*Статьи и блоги:*

1. [Real Python](https://realpython.com/) — блог о программировании на Python с множеством статей и руководств.

2. [Java Code Geeks](https://www.javacodegeeks.com/) — сайт с множеством статей, туториалов и примеров кода для Java.

3. [C++ Institute](https://cppinstitute.org/) — ресурсы и обучение по C++.

4. [JavaScript.info](https://javascript.info/) — учебник по JavaScript, который охватывает основные и продвинутые темы.

5. [C# Corner](https://www.c-sharpcorner.com/) — сообщество разработчиков, где можно найти статьи и ресурсы по C#.

**1.2. Алгоритмы и структуры данных: Основные понятия**

Алгоритмы и структуры данных - это базовые концепции в программировании и компьютерных науках. Они представляют собой инструменты и методы, используемые для обработки, хранения и управления данными. Понимание алгоритмов и структур данных позволяет программистам писать эффективные и оптимизированные программы.

1. Что такое алгоритмы?

Алгоритм — это последовательность шагов или правил, используемых для решения определенной задачи. Алгоритмы могут быть выражены в различных формах, включая естественный язык, псевдокод или строгие языки программирования. Важные характеристики алгоритмов:

- Конечность: Алгоритм должен завершаться после конечного числа шагов.

- Определенность: Каждый шаг должен быть четко определен.

- Входные данные: Алгоритм может принимать ноль или более входных данных.

- Выходные данные: Алгоритм должен выдавать один или несколько результатов (выходных данных).

- Эффективность: Алгоритм должен быть рабочим и использовать разумные ресурсы (время и память).

2. Алгоритмы сортировки

Сортировка — это процесс упорядочивания элементов массива или списка по определенному критерию (например, по возрастанию или убыванию). Существует множество алгоритмов сортировки, и каждый из них имеет свои особенности и применим в различных ситуациях. Вот несколько популярных алгоритмов сортировки:

- Сортировка пузырьком (Bubble Sort): простой алгоритм, который многократно проходит по списку, сравнивая соседние элементы и меняя их местами, если они находятся в неправильном порядке. Имеет сложность O(n²).

- Сортировка вставками (Insertion Sort): строит отсортированный массив по одному элементу, вставляя каждый элемент на его правильное место. Эффективен для небольших массивов, сложность O(n²).

- Сортировка выбором (Selection Sort): разделяет массив на отсортированную и неотсортированную части и каждый раз находит минимальный элемент из неотсортированной части и перемещает его в конец отсортированной части. Имеет сложность O(n²).

- Сортировка слиянием (Merge Sort): использует метод "разделяй и властвуй", разбивая массив на подмассивы, сортируя их и затем сливая обратно. Сложность O(n log n).

- Быстрая сортировка (Quick Sort): также основана на методе "разделяй и властвуй". Она выбирает опорный элемент и распределяет массив на элементы меньше и больше опорного, затем рекурсивно сортирует подмассивы. В среднем, имеет сложность O(n log n), но в худшем случае — O(n²).

3. Структуры данных

Структуры данных — это способы организации и хранения данных, оптимизированные для доступа и модификации. Существуют различные типы структур данных, среди которых наиболее популярными являются:

- Массивы: линейные структуры данных, где элементы хранятся в последовательных ячейках памяти, что обеспечивает быстрый доступ по индексу. Возможны как одно-, так и многомерные массивы.

- Списки: динамические структуры данных, состоящие из узлов, где каждый узел содержит данные и указатель на следующий узел. Списки могут быть односвязными и двусвязными.

- Стек (Stack): структура данных, работающая по принципу "последний пришел - первый вышел" (LIFO). Основные операции включают push (добавление элемента) и pop (удаление верхнего элемента). Применяется для отслеживания вызовов функций и реализации отмены действий в приложениях.

- Очередь (Queue): структура данных, работающая по принципу "первый пришел - первый вышел" (FIFO). Операции включают enqueue (добавление элемента в конец очереди) и dequeue (удаление элемента с начала очереди). Используется в ситуациях, требующих очередности, например, в системах управления задачами.

Суть задач, связанных с алгоритмами сортировки, стеком и очередью, заключается в решении конкретных проблем, связанных с обработкой, хранением и управлением данными. Давайте разберем каждую категорию задач подробнее:

1. Задачи алгоритмов сортировки

Алгоритмы сортировки используются, когда необходимо организовать набор данных в определенном порядке, что делает дальнейшую работу с ними более эффективной. Примеры задач:

- Упорядочивание списка: Необходимо отсортировать массив чисел, имен или объектов по определенному критерию (возрастание/убывание, алфавиту и т.д.). Это позволяет легко находить элементы и упрощает представление данных.

- Поиск: Многие алгоритмы поиска, такие как бинарный поиск, требуют, чтобы данные были отсортированы. Это значительно ускоряет процесс поиска по сравнению с неотсортированными данными.

- Статистика: Для вычисления медианы, квартилей и других статистических показателей часто требуется сортировка данных.

- Анализ данных: Профессионалы анализируют и визуализируют данные в упорядоченном формате для более глубокого понимания информации.

2. Задачи стека

Стек используется в тех случаях, когда необходима работа по принципу "последний пришел — первый вышел" (LIFO). Примеры задач:

- Управление вызовами функций: Когда вызывается функция, ее данные помещаются в стек. После завершения функции данные удаляются. Это важно для отслеживания состояния выполнения программы.

- Реализация отмены действий: Например, в текстовых редакторах реализации команд "отменить" и "повторить" часто используют стек, чтобы хранить выполненные действия.

- Парные символы: Проверка правильно открытых и закрытых скобок ( (), {}, [] ) в выражениях и коде. Стек позволяет отслеживать открытые скобки и проверять, что они закрываются в правильном порядке.

3. Задачи очереди

Очередь используется, когда необходимо хранить данные по принципу "первый пришел — первый вышел" (FIFO). Примеры задач:

- Управление задачами: В системах управления задачами или в операционных системах (например, планировщики процессов) используются очереди для обработки процессов, где задачи обрабатываются в порядке поступления.

- Обработка данных: Когда данные поступают в систему, например, от пользователей или систем, требуется обработка их в порядке поступления, что можно реализовать с помощью очереди.

- Сетевые пакеты и буферизация: В сетевых приложениях, таких как серверы, пакеты данных могут поступать в порядке, который необходимо обрабатывать последовательно.

Суть задач, связанных с алгоритмами сортировки, стеком и очередью, заключается в оптимизации обработки данных, управлении состоянием и обеспечении правильного порядка выполнения операций. Эти структуры данных и алгоритмы играют важную роль в разработке эффективных программ, позволяющих решать широкий круг задач в компьютерных науках и программировании. Понимание их принципов и применения поможет разработчикам создавать более эффективные и удобные приложения.

Конкретные примеры задач с использованием алгоритмов сортировки, стеков и очередей могут помочь лучше понять, как эти концепции применяются на практике. Давайте рассмотрим несколько каждых из них.

1. Задачи с алгоритмами сортировки

Пример 1: Сортировка списка оценок студентов

Задача: У нас есть список оценок студентов, и мы хотим отсортировать их по возрастанию.

Решение (Сортировка вставками):

def insertion\_sort(grades):

for i in range(1, len(grades)):

key = grades[i]

j = i - 1

while j >= 0 and key < grades[j]:

grades[j + 1] = grades[j]

j -= 1

grades[j + 1] = key

return grades

# Пример использования

grades = [75, 90, 85, 60, 95, 80]

sorted\_grades = insertion\_sort(grades)

print("Сортированные оценки:", sorted\_grades) # [60, 75, 80, 85, 90, 95]

Пример 2: Упорядочивание списка имен

Задача: У нас есть список имен, который необходимо отсортировать в алфавитном порядке.

Решение (Сортировка слиянием):

def merge\_sort(names):

if len(names) > 1:

mid = len(names) // 2

L = names[:mid]

R = names[mid:]

merge\_sort(L)

merge\_sort(R)

i = j = k = 0

while i < len(L) and j < len(R):

if L[i] < R[j]:

names[k] = L[i]

i += 1

else:

names[k] = R[j]

j += 1

k += 1

while i < len(L):

names[k] = L[i]

i += 1

k += 1

while j < len(R):

names[k] = R[j]

j += 1

k += 1

return names

# Пример использования

names = ["Anna", "John", "Zoe", "Mike", "Bob"]

sorted\_names = merge\_sort(names)

print("Сортированные имена:", sorted\_names) # ['Anna', 'Bob', 'John', 'Mike', 'Zoe']

2. Задачи со стеком

Пример 1: Проверка сбалансированных скобок

Задача: Проверить, являются ли скобки в строке сбалансированными.

Решение:

def is\_balanced(s):

stack = []

bracket\_map = {')': '(', '}': '{', ']': '['}

for char in s:

if char in bracket\_map.values():

stack.append(char)

elif char in bracket\_map.keys():

if stack == [] or bracket\_map[char] != stack.pop():

return False

return stack == []

# Пример использования

expression = "{[()]}[]"

print("Сбалансировано:", is\_balanced(expression)) # True

expression = "{[(])}"

print("Сбалансировано:", is\_balanced(expression)) # False

Пример 2: Реализация отмены действий в текстовом редакторе

Задача: Реализовать функциональность "отмены" в текстовом редакторе, используя стек.

Решение:

class TextEditor:

def \_\_init\_\_(self):

self.text = ""

self.history = []

def type(self, new\_text):

self.history.append(self.text) # Сохраняем текущее состояние

self.text += new\_text

def undo(self):

if self.history:

self.text = self.history.pop() # Возвращаемся к предыдущему состоянию

# Пример использования

editor = TextEditor()

editor.type("Hello")

editor.type(", world!")

print("Текущий текст:", editor.text) # "Hello, world!"

editor.undo()

print("После отмены:", editor.text) # "Hello"

3. Задачи с очередью

Пример 1: Обработка задач в очереди

Задача: Реализовать систему очереди для обработки задач.

Решение:

class TaskQueue:

def \_\_init\_\_(self):

self.queue = []

def add\_task(self, task):

self.queue.append(task)

def process\_task(self):

if self.queue:

return self.queue.pop(0) # Обработка первого элемента очереди

else:

return "Очередь пуста"

# Пример использования

task\_queue = TaskQueue()

task\_queue.add\_task("Задача 1")

task\_queue.add\_task("Задача 2")

print("Обработка задачи:", task\_queue.process\_task()) # "Задача 1"

print("Обработка задачи:", task\_queue.process\_task()) # "Задача 2"

Пример 2: Моделирование обслуживания клиентов в кафе

Задача: Сымитировать процесс обслуживания клиентов в кафе.

Решение:

class CafeQueue:

def \_\_init\_\_(self):

self.queue = []

def customer\_arrives(self, customer\_name):

self.queue.append(customer\_name)

def serve\_customer(self):

if self.queue:

return self.queue.pop(0) # Обслуживание первого клиента в очереди

else:

return "Очередь пуста"

# Пример использования

cafe\_queue = CafeQueue()

cafe\_queue.customer\_arrives("Клиент 1")

cafe\_queue.customer\_arrives("Клиент 2")

print("Обслуживаем клиента:", cafe\_queue.serve\_customer()) # "Клиент 1"

print("Обслуживаем клиента:", cafe\_queue.serve\_customer()) # "Клиент 2"

Эти примеры иллюстрируют, как можно решать конкретные задачи с использованием алгоритмов сортировки, стеков и очередей. Освоение этих концепций позволит вам эффективно обрабатывать данные и решать различные задачи в программировании.

***Контрольные вопросы***

*1. Алгоритмы сортировки*

1. Что такое алгоритм сортировки? Почему он важен в программировании?

2. Опишите основные различия между сортировкой пузырьком и сортировкой вставками. Когда целесообразно использовать каждую из них?

3. Что такое временная сложность алгоритма сортировки? Какова временная сложность сортировки слиянием в лучшем, среднем и худшем случаях?

4. Какие алгоритмы сортировки работают на принципе "разделяй и властвуй"? Приведите примеры.

5. Чем отличается стабильная сортировка от нестабильной? Приведите примеры стабильных и нестабильных алгоритмов.

*2. Стек*

1. Что такое стек и какие основные операции с ним выполняются? Опишите их.

2. Какова структура данных стека и в чем ее преимущества? Приведите примеры, где применяется стек.

3. Решите задачу: Как проверить, является ли строка из круглых скобок сбалансированной, используя стек?

4. Как можно реализовать стек с помощью массива или связного списка? Опишите, как будут выглядеть основные операции.

*3. Очередь*

1. Что такое очередь и чем она отличается от стека? Объясните принцип FIFO.

2. Приведите примеры задач, которые могут быть решены с использованием очереди.

3. Опишите методы и операции, используемые в реализации очереди. Как можно реализовать очередь с помощью массива?

4. Напишите код для обработки клиентов в очереди в кафе, используя структуру данных "очередь".

***Практические задания***

1. Алгоритмы сортировки

1. Сортировка массива: Реализуйте алгоритм Quick Sort и протестируйте его на различных наборах данных.

2. Сравнение сортировок: Сравните время выполнения сортировки пузырьком и сортировки слиянием на одном и том же наборе случайных чисел. Подготовьте отчет о результатах.

3. Сортировка по нескольким критериям: Напишите программу, которая сортирует список объектов (например, студентов) по нескольким критериям: сначала по имени, затем по возрасту.

2. Стек

1. Проверка сбалансированных скобок: Напишите программу, которая принимает строку и определяет, сбалансированы ли скобки. Используйте стек для решения этой задачи.

2. Реализация Undo: Создайте текстовый редактор, в котором можно вводить текст и делать его отмену с помощью стека. Реализуйте операции добавления текста и отмены.

3. Очередь

1. Реализация очереди: Реализуйте очередь с помощью массивов и связанных списков. Реализуйте операции добавления, удаления и просмотра элементов очереди.

2. Моделирование обработки клиентов: Сымитируйте очередь клиентов, которые получают обслуживание на кассе. Реализуйте возможность добавления клиентов в очередь и их обслуживания.

***Онлайн-ресурсы***

1. GeeksforGeeks (https://www.geeksforgeeks.org/)

- Обширный ресурс, содержащий статьи и руководства по структурам данных и алгоритмам, включая теоретические объяснения и практические примеры.

2. Khan Academy (https://www.khanacademy.org/computing/computer-science/algorithms)

- Бесплатные курсы по основам алгоритмов и структур данных с видеолекциями и интерактивными упражнениями.

3. Coursera (https://www.coursera.org/)

- Платформа, предлагающая курсы по алгоритмам и структурам данных, в том числе от таких университетов, как Stanford и Princeton.

4. Сайты с кодами и задачами

- Такие платформы, как LeetCode (https://leetcode.com/), HackerRank (https://www.hackerrank.com/), CodeWars (https://www.codewars.com/) позволяют практиковаться в решении задач, связанных с алгоритмами и структурами данных.

**1.3. Практические задачи по программированию**

В этом разделе предложены задания, которые требуют разработки простых приложений, используя базовые конструкции языка программирования. Эти задачи помогут студентам лучше понять основные концепции программирования и разработать логическое мышление.

Задание 1: Калькулятор

Описание: Разработайте консольное приложение "Калькулятор", которое выполняет базовые арифметические операции: сложение, вычитание, умножение и деление. Программа должна запрашивать у пользователя два числа и операцию, а затем выводить результат.

Дополнительные требования:

- Обработайте случаи деления на ноль.

- Добавьте возможность выполнять несколько операций подряд, пока пользователь не решит выйти.

Пример работы:

Введите первое число: 5

Введите оператор (+, -, \*, /): /

Введите второе число: 0

Ошибка: Деление на ноль!

Задание 2: Угадай число

Описание: Создайте игру "Угадай число", в которой компьютер случайным образом выбирает число от 1 до 100, а пользователь пытается его угадать. Программа должна предоставлять подсказки "меньше" или "больше" после каждой попытки.

Дополнительные требования:

- Укажите количество попыток, доступных пользователю.

- Поздравьте пользователя с победой или сообщите, если он исчерпал все попытки.

Пример работы:

Компьютер загадал число от 1 до 100. У вас есть 10 попыток.

Ваше предположение: 50

Меньше!

Ваше предположение: 30

Больше!

...

Вы угадали за 8 попыток!

Задание 3: Список покупок

Описание: Разработайте приложение для управления списком покупок. Пользователь должен иметь возможность добавлять, удалять и отображать элементы списка.

Дополнительные требования:

- Сохраняйте список в памяти, чтобы он не обнулялся при повторном запуске программы.

- Позвольте пользователю очищать весь список и отображать количество оставшихся элементов.

Пример работы:

1. Добавить элемент

2. Удалить элемент

3. Показать список

4. Очистить список

0. Выход

Выберите действие: 1

Введите название продукта: Яблоко

Яблоко добавлено в список.

Задание 4: Конвертер валют

Описание: Создайте приложение "Конвертер валют", которое позволяет пользователю конвертировать одну валюту в другую. Используйте фиксированные коэффициенты для конвертации (например, 1 USD = 74.5 RUB).

Дополнительные требования:

- Добавьте поддержку нескольких валют (USD, EUR, RUB).

- Позвольте пользователю вводить сумму на конвертацию и выводите результат.

Пример работы:

Выберите валюту для конвертации:

1. USD в RUB

2. EUR в USD

Введите номер операции: 1

Введите сумму в USD: 10

10 USD = 745 RUB

Задание 5: Список студентов

Описание: Разработайте приложение для управления списком студентов. Пользователь может добавлять студентов с их именами и оценками, а также отображать список студентов с их оценками.

Дополнительные требования:

- Реализуйте возможность вычисления среднего балла студентов.

- Позвольте пользователю искать студента по имени.

Пример работы:

1. Добавить студента

2. Показать студентов

3. Найти студента по имени

0. Выход

Выберите действие: 1

Введите имя студента: Иван

Введите оценку: 5

Иван добавлен в список.

**Глава 2. Проектирование и разработка**

**2.1. Процесс разработки программного обеспечения**

Процесс разработки программного обеспечения (ППО) представляет собой систематизированный подход к созданию программных продуктов. Этот процесс включает в себя несколько ключевых этапов, начиная с первоначальной идеи и заканчивая окончательной реализацией и сопровождением программного обеспечения. В данной статье мы рассмотрим эти этапы подробно.

1. Идея и определение требований

Описание:

На этом этапе происходит генерация идеи для нового программного продукта или улучшения существующего. Важно четко определить, какую проблему будет решать программа и для какой аудитории она предназначена.

Действия:

- Сбор информации о потребностях пользователей.

- Анализ рыночных тенденций и конкурентов.

- Формулирование цели проекта.

- Создание предварительных требований к продукту.

Результат:

Документ с определением требований, описывающий основные функции и целевую аудиторию.

---

2. Проектирование (Дизайн)

Описание:

На этапе проектирования происходит создание архитектуры программного продукта. Это включает в себя как высокоуровневое, так и детальное проектирование.

Действия:

- Разработка архитектуры приложения, включая выбор технологий.

- Создание макетов (wireframes) пользовательского интерфейса.

- Определение базы данных и структуры хранения данных.

- Разработка алгоритмов обработки данных.

Результат:

Документы проектирования, включающие архитектурные схемы, макеты интерфейса и планы интеграции.

---

3. Разработка (Кодирование)

Описание:

Это этап, на котором непосредственно осуществляется написание кода. Команда разработчиков реализует проектные документы, используя выбранные технологии.

Действия:

- Написание кода согласно стандартам и практикам программирования.

- Реализация модулей и функций, описанных в требованиях.

- Проведение регулярных встреч для координации работы команды (например, Agile-методологии).

- Ведение документации кода.

Результат:

Рабочий код продукта, который соответствует проектной документации.

---

4. Тестирование

Описание:

На этом этапе осуществляется проверка программного продукта на наличие ошибок и несоответствий требованиям. Тестирование может быть как автоматизированным, так и ручным.

Действия:

- Разработка тестовых сценариев и планов тестирования.

- Проведение различных видов тестирования:

- Модульное тестирование (проверка отдельных компонентов).

- Интеграционное тестирование (проверка взаимодействия между модулями).

- Системное тестирование (проверка всей системы в целом).

- Приемочное тестирование (проверка готовности продукта к эксплуатации).

- Исправление обнаруженных ошибок и повторное тестирование.

Результат:

Подтвержденная работоспособность программного продукта и наличие документации по тестированию.

---

5. Внедрение

Описание:

После успешного тестирования программный продукт передается пользователям. Внедрение может осуществляться поэтапно или одновременно, в зависимости от стратегии.

Действия:

- Подготовка инфраструктуры для запуска (серверы, устройства и т.д.).

- Установка и конфигурация программного обеспечения.

- Проведение начального обучения пользователей.

- Подготовка и распространение пользовательской документации.

Результат:

Работающий программный продукт, доступный для пользователей.

---

6. Поддержка и обслуживание

Описание:

После внедрения программного обеспечения необходимо обеспечить его поддержку и обслуживание. Эта стадия критически важна для долгосрочного успеха проекта.

Действия:

- Обеспечение технической поддержки пользователей (горячая линия, онлайн-чат).

- Решение проблем и исправление ошибок, возникающих в процессе эксплуатации.

- Обновление программного обеспечения с добавлением новых функций или улучшениями.

- Проведение регулярного мониторинга системы для поддержания ее производительности.

Результат:

Стабильное функционирование программного продукта, удовлетворяющее потребности пользователей.

---

7. Завершение проекта

Описание:

На этом этапе происходит формальное закрытие проекта. Важно проанализировать, чего удалось достичь, и какие уроки можно извлечь для будущих проектов.

Действия:

- Подготовка отчета о результатах проекта.

- Оценка выполненных задач и целей.

- Сбор отзывов от участников команды и пользователей.

- Проведение встречи для обсуждения успешных практик и улучшений.

Результат:

Документированный анализ проекта, который послужит основой для будущих разработок.

---

Процесс разработки программного обеспечения является многогранным и требует четкой организации на каждом этапе. Следование описанным этапам помогает обеспечить качество и устойчивость готового продукта, а также удовлетворить потребности пользователей. Каждая фаза предполагает активное взаимодействие между участниками проекта, что способствует успешному завершению и внедрению программного обеспечения.

***Контрольные вопросы***

1. Идея и определение требований:

- Каковы основные цели на этапе генерации идеи для нового программного продукта?

- Какие методы могут быть использованы для сбора требований от пользователей?

- Почему важно проводить анализ конкурентов и рыночных тенденций на этом этапе?

2. Проектирование (Дизайн):

- Что включает в себя архитектура программного обеспечения?

- Какова роль прототипирования и создания макетов на этапе проектирования?

- Почему важно подробно описывать структуру данных и алгоритмы обработки данных?

3. Разработка (Кодирование):

- Какие технологии и языки программирования могут быть выбраны для реализации проекта?

- Каковы основные задачи разработчиков на этапе кодирования?

- Зачем необходима документация кода, и какие ее основные элементы?

4. Тестирование:

- Какие виды тестирования существуют, и в чем их отличие?

- Как важно тестирование для качества программного продукта?

- Каковы основные цели приемочного тестирования?

5. Внедрение:

- Какие действия необходимо предпринять перед запуском продукта?

- Почему важно проводить обучение пользователей в процессе внедрения?

- Каковы основные задачи команды на этапе внедрения?

6. Поддержка и обслуживание:

- Какие виды поддержки могут быть предоставлены пользователям после внедрения?

- Почему важно регулярно обновлять программное обеспечение?

- Как осуществляется мониторинг производительности системы?

7. Завершение проекта:

- Что включает в себя процесс формального завершения проекта?

- Какую информацию следует собрать для анализа завершения проекта?

- Почему важно обсуждать успешные практики и улучшения после завершения проекта?

8. Общие вопросы:

- Как взаимосвязаны различные этапы процесса разработки программного обеспечения?

- Как применение Agile и других методологий может повлиять на процесс разработки?

- Какие риски могут возникнуть на каждом из этапов, и как их можно минимизировать?

***Список источников для самостоятельного ознакомления***

1. "Проектирование и разработка программного обеспечения" – И. В. Андреев

- Книга охватывает основные аспекты проектирования и разработки программного обеспечения, а также методы и технологии, используемые в этой области.

2. "Чистый код. Создание, анализ и рефакторинг" – Роберт Мартин

- Перевод классической работы о написании чистого и поддерживаемого кода, включая примеры и практические рекомендации.

3. "Разработка программного обеспечения. Методологии и технологии" – В. В. Фролов

- Учебное пособие, охватывающее ключевые методологии разработки программного обеспечения, включая Agile, Waterfall и другие.

4. "Современные методологии управления проектами" – А. В. Ковалев

- Книга рассматривает различные методологии и подходы к управлению проектами, включая принципы Agile и Scrum.

5. "Инженерия программного обеспечения" – И. Л. Снедкова

- Учебник, в котором подробно рассматриваются основные этапы разработки программного обеспечения и принципы проектирования.

6. "Agile-проектирование программного обеспечения" – М. Куинн

- Книга об Agile-методах и их применении в разработке, включая Scrum, Kanban и Extreme Programming.

7. "Тестирование программного обеспечения" – С. С. Кузнецов

- Книга описывает основные методы и практики тестирования программного обеспечения, включая автоматизацию тестирования.

8. "Управление проектами разработки программного обеспечения" – А. А. Петров

- Описание методов и инструментов управления проектами в области разработки программного обеспечения, а также обсуждение практических кейсов.

9. "Искусство программирования" – Д. Кнут

- Классическая работа по программированию, которая охватывает различные аспекты создания эффективных алгоритмов и структур данных.

10. "Методы и средства разработки программного обеспечения" – Н. В. Сапунов

- Учебник, подробно рассматривающий методы и средства, используемые в процессе разработки программного обеспечения.

*Онлайновые ресурсы*

- Хабр (Habr) — онлайн-платформа для обсуждения технологий и разработки, где можно найти множество статей и материалов по темам разработки программного обеспечения: [Habr](https://habr.com/ru/)

- Сайт "Программирование на русском" — ресурсы и статьи на русском языке по программированию, включая процесс разработки ПО: [Программирование на русском](http://www.programming-ru.com/)

**2.2. Работа с системами управления версиями**

1. Введение в системы управления версиями

1.1. Что такое система управления версиями?

- Система управления версиями (СУВ) — это инструмент, который помогает разработчикам отслеживать изменения в коде и управлять различными версиями программного обеспечения.

- СУВ позволяет возвращаться к предыдущим версиям кода, выполнять параллельную разработку и там, где это необходимо, сливать изменения назад в основную ветку.

1.2. Почему Git?

- Git является распределенной системой управления версиями, которая обеспечивает независимость разработчиков и позволяет им работать локально.

- Поддерживает параллельную разработку с помощью веток и облегчает слияние изменений.

2. Основные концепции Git

2.1. Репозиторий

- Репозиторий представляет собой хранилище кода, в котором находятся все изменения и версия проекта.

- Каждый пользователь может клонировать репозиторий и работать с его локальной копией.

2.2. Коммиты

- Коммит — это снимок состояния проекта в определенный момент времени. Каждый коммит имеет уникальный идентификатор (SHA-1) и включает информацию о внесенных изменениях.

2.3. Ветки

- Ветки позволяют разработчикам работать над различными задачами параллельно.

- Основная ветка (обычно main или master) представляет собой стабильную версию проекта, в которую вливаются изменения из других веток.

2.4. Слияние и конфликты

- Слияние (merge) — это процесс объединения изменений из одной ветки в другую.

- Конфликты могут возникнуть, если изменения в двух ветках касаются одних и тех же строк кода и не могут быть автоматически объединены.

3. Установка и начальная настройка Git

3.1. Установка Git

- Git можно установить из официального сайта [git-scm.com](https://git-scm.com/).

- Установка доступна для различных операционных систем, таких как Windows, macOS и Linux.

3.2. Настройка пользователя

После установки Git необходимо настроить имя пользователя и адрес электронной почты:

git config --global user.name "Ваше Имя"

git config --global user.email "ваш@email.com"

### 4. Основные команды Git

4.1. Создание репозитория

- Для создания нового репозитория используйте команду:

git init <имя\_папки>

- Если вы хотите клонировать существующий репозиторий:

git clone <URL\_репозитория>

4.2. Проверка статуса

- Для проверки состояния файлов в репозитории используйте команду:

git status

4.3. Добавление изменений

- Для добавления изменений в индекс (staging area) используется команда:

git add <имя\_файла>

- Чтобы добавить все изменения:

git add.

4.4. Создание коммита

- Для сохранения изменений в репозитории выполните:

git commit -m "Сообщение к коммиту"

4.5. Просмотр истории коммитов

- Для просмотра истории коммитов используйте команду:

git log

4.6. Работа с ветками

- Для создания новой ветки:

git branch <название\_ветки>

- Для переключения между ветками:

git checkout <название\_ветки>

- Для объединения изменений из одной ветки в другую:

git merge <название\_ветки>

5. Использование Git в проектах

5.1. Создание нового проекта

- Начните с создания нового репозитория Git и добавления исходного кода.

- Старайтесь делать небольшие и частые коммиты для лучшего отслеживания изменений.

5.2. Параллельная разработка

- Используйте ветки для работы над новыми функциональными возможностями или исправлениями.

- Периодически сливайте изменения с основной веткой, чтобы избежать больших конфликтов.

5.3. Ведение совместной работы

- Если несколько разработчиков работают над одним проектом, используйте пуши и пуллы для синхронизации изменений в удаленном репозитории.

- Убедитесь, что вы часто объединяете изменения, чтобы отслеживать работу других участников команды.

5.4. Использование удаленных репозиториев

- GitHub, GitLab и Bitbucket — популярные платформы, позволяющие размещать удаленные репозитории и упрощающие совместную работу.

- Используйте команды git push и git pull для отправки изменений на удаленные репозитории и получения изменений от них.

6. Заключение

Работа с версиями с использованием Git является важной частью современного процесса разработки программного обеспечения. Понимание основных команд, концепций и принципов работы с репозиториями поможет создать качественное, надежное и управляемое программное обеспечение. Git не только облегчает управление версиями, но и значительно ускоряет разработку за счёт параллельного процесса разработки и интеграции изменений. Владение Git — необходимый навык для любого разработчика, который стремится работать эффективно и организованно в команде.

***Контрольные вопросы***

1. Основные понятия:

- Что такое система управления версиями и какие основные задачи она решает?

- Чем Git отличается от централизованных систем управления версиями?

- Что такое репозиторий в контексте Git?

2. Команды Git:

- Какова основная функция команды git init?

- Что делает команда git add и как она используется?

- В чем разница между git commit и git push?

3. Работа с ветками:

- Как создать новую ветку в Git?

- Как переключиться на существующую ветку?

- Что происходит при выполнении команды git merge?

4. Коммиты и история:

- Что такое коммит и какую роль он играет в Git?

- Как можно просмотреть историю коммитов?

- Как можно откатить изменения в Git?

5. Управление конфликтами:

- Как возникают конфликты при слиянии веток и как их решать?

- Что делать, если изменения в разных ветках касаются одних и тех же строк кода?

6. Удаленные репозитории:

- Что такое удаленный репозиторий в Git?

- Как можно склонировать репозиторий и почему это полезно?

- Какую команду использовать для синхронизации локальных и удаленных изменений?

*Практические задания*

1. Создание нового репозитория:

- Создайте новый локальный репозиторий Git на своем компьютере.

- Добавьте текстовый файл с описанием проекта и выполните коммит.

2. Работа с ветками:

- Создайте новую ветку для новой функциональности (например, feature/login).

- Переключитесь на эту ветку, добавьте новый файл (например, login.py) и сделайте коммит.

- Вернитесь на основную ветку и слейте изменения из созданной ветки.

3. Работа с удаленным репозиторием:

- Создайте новый репозиторий на платформе GitHub или GitLab.

- Склонируйте этот репозиторий на локальный компьютер.

- Добавьте commit и push (отправьте) изменения в удаленный репозиторий.

4. Решение конфликтов:

- Создайте две ветки и внесите различные изменения в один и тот же файл в каждой ветке.

- Попробуйте выполнить слияние веток и исправить возникший конфликт.

5. Обзор истории:

- Используйте команду git log для просмотра истории коммитов вашего репозитория.

- Найдите последний коммит и вернитесь к нему, используя команду git checkout.

6. Работа с идеями и задачами:

- Напишите в одной ветке реализацию какой-либо новой функции, а в другой — исправление ошибки.

- После завершения работы, объедините обе ветки в основную ветку и убедитесь, что все работает корректно.

***Список источников для самостоятельного ознакомления***

*1. Книги:*

- Чамаров, А. (2015). Git для профессионалов: Продуктивная работа в системах контроля версий. М.: БХВ-Петербург.

- Зисевич, С. (2017). Git. Ускоренный курс. Санкт-Петербург: Питер.

- Базуев, Р. (2018). Git. Простое руководство для начинающих. Москва: Бомбора.

- Шнуриков, Н. (2016). Git. Книга для разработчика. Москва: НТ Пресс.

*2. Онлайн-ресурсы и статьи:*

- Блог "Хабр". (2020). Введение в Git: Основные команды и функции. [Ссылка на статью](https://habr.com/ru/company/ruvox/blog/496290/)

- Молотов, А. (2021). 10 полезных команд Git для разработчиков. [Ссылка на статью](https://habr.com/ru/post/532680/)

- Логвинов, И. (2019). Уроки Git для начинающих на GeekBrains. [Ссылка на курс](https://gb.ru/courses/1013)

- Курс на Stepik: Основы системы контроля версий Git. [Ссылка на курс](https://stepik.org/course/69799)

*3. Видео-лекции и курсы:*

- Студия "Онлайн-Университет". Курс по Git: Уроки и видеоинструкции. [Ссылка на курс](https://www.online-university.ru/online-git)

- YouTube-канал "Учеба и Работа". Git и GitHub для начинающих. [Ссылка на видео](https://www.youtube.com/watch?v=HxVyI1YneyI)

**2.3. Командные проекты**

Командные проекты представляют собой совместную работу группы разработчиков над одним или несколькими проектами. Эффективная работа в команде требует координации действий, использования инструментов для управления проектами и совместной работы, а также соблюдения четких процессов разработки. В этой теме мы рассмотрим основные аспекты командных проектов, включая роли участников, использование систем управления версиями (на примере Git), методы организации работы и практики, способствующие успешному завершению проектов.

1. Роли в командном проекте

Каждый участник команды может выполнять разные роли в зависимости от потребностей проекта и его целей. Примеры ролей:

- Разработчик (Developer) – отвечает за написание и тестирование кода. Разработчики могут иметь разные специализации, например, фронтенд или бэкенд разработка.

- Тестировщик (QA Engineer) – отвечает за тестирование приложения для выявления ошибок и обеспечения качества.

- Менеджер проекта (Project Manager) – планирует и контролирует процесс разработки, расставляет приоритеты задач, обеспечивает коммуникацию между участниками команды.

- Дизайнер (Designer) – разрабатывает пользовательский интерфейс и опыт, создавая визуальную идею проекта.

- Системный администратор (DevOps Engineer) – отвечает за развертывание и поддержку производственной среды, написание скриптов для автоматизации.

2. Инструменты для командной работы

Для успешного управления командным проектом важно использовать подходящие инструменты. Некоторые из них:

- Системы управления версиями (например, Git) – позволяют отслеживать изменения в коде, сотрудничать при кодировании и разрешать конфликты.

- Системы отслеживания задач (например, Jira, Trello, Asana) – помогают отслеживать выполнение задач, планировать спринты и организовывать работу.

- Инструменты для совместной разработки (например, GitHub, GitLab) – обеспечивают платформу для кодирования, совместного редактирования и обсуждения изменений.

- Чаты и мессенджеры (например, Slack, Microsoft Teams) – для быстрой и эффективной коммуникации внутри команды.

3. Процесс работы в командном проекте

Ключевые этапы работы над командным проектом:

1. Планирование:

- Определяются цели проекта.

- Составляется график работ.

- Определяются роли участников и задачи, которые они должны выполнить.

2. Разработка:

- Использование системы управления версиями для разработки кода.

- Регулярные коммиты и пуши изменений в удаленный репозиторий.

- Проведение код-ревью (peer review) для обеспечения качества кода.

3. Тестирование:

- Разработка тестов (автоматизированных и ручных) для проверки функциональности.

- Обеспечение документирования тестов и результатов.

4. Развертывание:

- Внедрение кода в производственную среду.

- Настройка автоматизированных процессов развертывания (CI/CD – Continuous Integration/Continuous Deployment).

5. Обратная связь и итерации:

- Сбор отзывов пользователей и тестировщиков.

- Планирование итераций для улучшения и доработки функционала.

4. Практика командной работы

Некоторые лучшие практики для успешной работы в командных проектах:

- Регулярные встречи команды (например, Daily Standup) для обсуждения текущего статуса, выявления проблем и совместного решения вопросов.

- Создание документации, описывающей процесс разработки, стандарты кодирования и другие важные аспекты для новых участников команды.

- Использование гит-веток для работы над новыми функциональностями и исправлениями ошибок (например, эти ветки могут называться feature/имя\_функции и bugfix/имя\_ошибки).

- Открытое обсуждение кода, предлагаемые улучшения и корректировки на этапе код-ревью.

Командные проекты представляют собой важный аспект разработки программного обеспечения. Эффективная работа в команде требует четкого определение ролей, использования подходящих инструментов и следования установленным процессам. Принятие лучших практик и постоянное улучшение методов работы позволит командам достигать высоких результатов и успешно завершать проекты.

***Практические задания:***

Задание 1: Определение ролей в команде

Цель: Понять различные роли в команде и их важность.

Инструкция:

1. Сформируйте группу из 4-5 человек.

2. Обсудите и назначьте роли (разработчик, тестировщик, менеджер проекта, дизайнер и системный администратор) каждому участнику команды.

3. На основе выбранной роли каждый участник должен написать короткий отчет (150-200 слов) о своих обязанностях и том, как они взаимодействуют с другими членами команды.

Задание 2: Выбор инструментов для командной работы

Цель: Изучить различные инструменты для командной работы и выбрать подходящие для вашего проекта.

Инструкция:

1. Исследуйте 3-4 инструмента для управления проектами (например, Trello, Jira, Asana, Microsoft Teams).

2. Создайте сравнительную таблицу, в которой укажите плюсы и минусы каждого инструмента.

3. На основании проведенного анализа выберите один инструмент, который лучше всего подойдет вашей команде, и обоснуйте свой выбор (150-200 слов).

Задание 3: Проектирование процесса разработки

Цель: Разработать процесс разработки для гипотетического проекта.

Инструкция:

1. Сформируйте небольшую группу (3-4 человека).

2. Придумайте гипотетический проект (например, создание веб-приложения или мобильного приложения).

3. Определите основные этапы разработки:

- Идея и планирование

- Проектирование

- Разработка

- Тестирование

- Развертывание

4. Напишите описание процессов (200-300 слов) для каждого этапа, включая способы мониторинга прогресса и управления задачами.

Задание 4: Проведение код-ревью

Цель: Научиться проводить код-ревью и давать конструктивную обратную связь.

Инструкция:

1. Находите либо используйте заранее созданный простой код (например, на GitHub), который можно улучшить.

2. Проведите код-ревью вместе с вашей командой, используя инструменты для совместной разработки (например, GitHub или GitLab).

3. Обсудите, что можно улучшить в коде, и составьте список рекомендаций по улучшению.

4. Напишите отчет (200-300 слов) о том, что вы узнали о процессе код-ревью и о том, как давать конструктивную обратную связь.

Задание 5: Симуляция командного проекта

Цель: На практике применить методы работы в команде.

Инструкция:

1. Сформируйте небольшую группу (4-6 человек).

2. Выберите проект (например, создание сайта, мобильного приложения или игры).

3. Рассмотрите возможность использования Git для контроля версий в вашем проекте.

4. Запланируйте спринт: определите задачи, роли и временные рамки.

5. Реализуйте спринт, следуя своему плану, и в конце обсудите результаты:

- Что удалось сделать?

- С какими трудностями столкнулись?

- Как можно улучшить процесс в будущем?

6. Напишите отчет (300-500 слов) о проведенной симуляции, включая выводы и рекомендации.

**Глава 3. Основы компьютерной системы и сети**

**3.1. Структура компьютера**

Компьютер в современном мире стал неотъемлемой частью жизни человека. Он выполняет множество задач и функций, обеспечивая выполнение вычислений, хранение данных и взаимодействие с пользователем. Понимание структуры компьютера требует изучения как аппаратной, так и программной частей. В этом курсе мы рассмотрим основные компоненты компьютера, их назначение и взаимодействие, а также программы, которые управляют аппаратным обеспечением.

1. Аппаратная часть компьютера

Аппаратная часть компьютера, или "железо", состоит из физических компонентов, которые выполняют различные операции. Основные компоненты включают:

1.1. Центральный процессор (ЦП)

Центральный процессор (ЦП) — это "мозг" компьютера, который выполняет все арифметические и логические операции. Его основные характеристики включают:

- Тактовая частота: Измеряется в герцах (ГГц) и определяет скорость выполнения операций.

- Количество ядер: Современные процессоры могут иметь несколько ядер, что позволяет им выполнять многозадачные операции более эффективно.

- Кэш-память: Встроенная память, используемая для ускорения доступа к часто используемым данным.

1.2. Оперативная память (ОП)

Оперативная память (ОП) — это временное хранилище, где хранятся данные и инструкции, необходимые для выполнения текущих задач. Основные характеристики:

- Объем: Измеряется в гигабайтах (ГБ). Чем больше объем памяти, тем больше задач может обрабатывать компьютер одновременно.

- Скорость: Указывает на быстродействие доступа к данным.

1.3. Хранилище данных

Системы хранения данных обеспечивают долговременное хранение информации. Основные виды:

- Жесткий диск (HDD): Механическое устройство с вращающимися пластинами для хранения данных.

- Твердотельный накопитель (SSD): Электронное устройство, представляющее быструю замену HDD, без движущихся частей.

1.4. Материнская плата

Материнская плата связывает все компоненты компьютера. Основные элементы включают:

- Сокет для процессора: Место, куда устанавливается ЦП.

- Слоты для оперативной памяти: Контейнеры для установки модулей ОП.

- Разъемы для хранения: Порты для подключения HDD и SSD.

1.5. Видеокарта

Видеокарта (графический процессор) отвечает за обработку графики. Основные характеристики:

- Объем видеопамяти: Определяет, сколько данных видеокарта может удерживать.

- Частота и количество ядер: Влияет на производительность графических операций.

1.6. Периферийные устройства

Периферийные устройства обеспечивают ввод и вывод данных. Примеры:

- Клавиатура и мышь: Устройства ввода.

- Монитор: Устройство вывода, отображающее информацию.

- Принтер: Устройство для печати документов.

2. Программная часть компьютера

Программная часть компьютера включает в себя набор инструкций и программ, которые управляют аппаратными компонентами. Она делится на несколько категорий:

2.1. Операционная система (ОС)

Операционная система управляет ресурсами компьютера и обеспечивает взаимодействие между аппаратным обеспечением и пользователем. Примеры ОС:

- Windows

- macOS

- Linux

Основные функции ОС:

- Управление файлами и папками.

- Оперативное распределение памяти.

- Управление процессами.

2.2. Прикладные программы

Прикладные программы выполняют конкретные задачи для пользователей. Примеры:

- Текстовые редакторы (например, Microsoft Word)

- Графические редакторы (например, Adobe Photoshop)

- Веб-браузеры (например, Google Chrome)

2.3. Утилиты

Утилиты — это программы, которые помогают настраивать, обслуживать и управлять компьютерной системой. Примеры:

- Антивирусное ПО

- Системы резервного копирования

- Операционные утилиты (например, Disk Cleanup)

2.4. Языки программирования

Языки программирования используются для написания программ, выполняющих определенные задачи. Примеры:

- Python

- Java

- C++

Изучение языков позволяет пользователям создавать свои приложения и системы.

3. Взаимодействие аппаратной и программной частей

Аппаратное обеспечение и программное обеспечение взаимодействуют друг с другом для выполнения вычислений и управления жизненным циклом данных:

- Низкоуровневые операции: ОС взаимодействует с аппаратным обеспечением через драйверы, которые являются программным обеспечением, позволяющим ОС "общаться" с конкретными устройствами.

- Использование ресурсов: Прикладные программы запрашивают ресурсы (память, процессорное время) у ОС для выполнения своих задач. ОС управляет распределением ресурсов.

Понимание структуры компьютера, включая аппаратную и программную части, является основой для работы с вычислительными системами. Знание основных компонентов и их функций помогает пользователям эффективно использовать компьютер, а также выявлять и устранять возможные проблемы. Совершенствование навыков в этой области открывает возможности для дальнейшего изучения более сложных тем, таких как сборка компьютерных систем, программирование и администрирование.

***Контрольные вопросы:***

1. Что такое центральный процессор (ЦП) и какую роль он играет в компьютере?

2. Каковы основные различия между оперативной памятью (ОП) и хранилищем данных (HDD/SSD)?

3. Опишите, как материнская плата взаимодействует с другими компонентами компьютера.

4. Почему видеокарта важна для компьютеров, ориентированных на графику и игры?

5. Что такое операционная система и какие ее основные функции? Приведите примеры популярных ОС.

6. Назовите и объясните три основных типа программного обеспечения.

7. Как осуществляется взаимодействие между аппаратной частью компьютера и программным обеспечением?

8. Что такое драйвер, и для чего он нужен в системе?

9. Объясните, как языки программирования связаны с работой компьютеров.

10. Что такое периферийные устройства и как они влияют на использование компьютера?

***Задания по теме***

Задание 1: Исследование компонентов компьютера

Инструкция:

1. Найдите информацию о модернизации вашего компьютера или компьютера, который вам известен.

2. Создайте список всех компонентов компьютера (ЦП, ОП, видеокарта, HDD и т. д.) и укажите их характеристики (модель, объем, тактовая частота и т.д.).

3. Напишите краткое описание (параграф) о том, как эти компоненты работают вместе для выполнения вычислений и выполнения задач.

Задание 2: Сравнение жесткого диска и твердотельного накопителя

Инструкция:

1. Исследуйте и соберите информацию о преимуществах и недостатках жесткого диска (HDD) и твердотельного накопителя (SSD).

2. Составьте сравнительную таблицу, в которой перечислены ключевые характеристики (скорость, надежность, цена, производительность и т.д.).

3. На основании своего анализа напишите краткий отчет (150-200 слов) о том, какой тип накопителя вы выбрали бы для домашнего компьютера и почему.

Задание 3: Рабочая станция

Инструкция:

1. Определите, какие компоненты понадобятся для сборки рабочего компьютера, ориентированного на определенные задачи (например, графический дизайн, программирование, игры).

2. Создайте список необходимых компонентов и их характеристик.

3. Напишите краткое обоснование выбора каждого компонента (например, почему вы выбрали определенный процессор или видеокарту).

Задание 4: Роль операционной системы

Инструкция:

1. Выберите одну из популярных операционных систем (например, Windows, Linux или macOS).

2. Найдите информацию о ее функциональности и особенностях.

3. Напишите краткий обзор (200-300 слов) о том, как эта ОС управляет аппаратными ресурсами и взаимодействует с программным обеспечением.

Задание 5: Создание схемы

Инструкция:

1. Создайте схему, изображающую взаимодействие между аппаратным и программным обеспечением.

2. Убедитесь, что включены все основные компоненты (ЦП, ОП, материнская плата, ОС, прикладные программы, драйверы и т.д.).

3. Добавьте пояснительные надписи к каждому элементу схемы, описывающие его функции и взаимодействие с другими компонентами.

Задание 6: Программирование и аппаратное обеспечение

Инструкция:

1. Напишите короткий эссе (200-300 слов) о том, как языки программирования влияют на работу компьютеров и как программное обеспечение взаимодействует с аппаратным обеспечением.

2. Приведите примеры языков программирования, которые широко используются для разработки программного обеспечения, работающего с различными аппаратными компонентами.

Задание 7: Периферийные устройства

Инструкция:

1. Исследуйте несколько примеров периферийных устройств (например, принтеры, сканеры, веб-камеры, внешние жесткие диски).

2. Напишите краткое описание каждого устройства, объяснив его функцию и принцип работы.

3. Укажите, как каждое из этих устройств может улучшить функциональность компьютера для пользователя.

***Список источников для самостоятельного ознакомления***

1. Уткин, А. В. "Основы компьютерной архитектуры". - М.: Горячая линия – Телеком, 2018.

Книга охватывает ключевые концепции компьютерной архитектуры, включая аппаратные компоненты и их взаимодействие.

2. Буховцев, О. Н. "Компьютеры: устройство, работа, программирование". - М.: ДМК Пресс, 2017.

Учебник, посвященный устройству компьютеров и основам программирования, который подает материал доступным языком.

3. Лисовский, С. Н. "Компьютерная грамота в вопросах и ответах". - М.: Время, 2015.

Краткое руководство по основам компьютеров, его устройству и использованию программного обеспечения.

4. Селезнев, В. И. "Основы информационных технологий". - М.: ИДДК, 2016.

Книга охватывает широкий спектр вопросов, включая апаратную и программную части компьютеров, а также их применение в различных областях.

5. Константинов, С. В. "Архитектура компьютерных систем". - М.: Издательство Бином, 2019.

Обширное руководство по архитектуре компьютерных систем, охватывающее как аппаратные, так и системные аспекты.

6. Голубев, А. В. "Современные операционные системы". - М.: Бином. Лаборатория знаний, 2020.

Книга подробно рассматривает современные операционные системы и их взаимодействие с аппаратным обеспечением.

7. Шевченко, Ю. И. "Программирование для начинающих". - М.: Питер, 2018.

Учебное пособие для начинающих, в котором рассматриваются основы программирования и его связь с компьютером.

8. Ченкин, А. В. "Компьютерные сети и технологии". - М.: Высшая школа, 2020.

Книга посвящена сетевым технологиям и их взаимодействию с аппаратной частью компьютеров.

9. Семенов, А. В. "Основы теории и практики системного администрирования". - М.: Инфра-М, 2019.

Учебник для системных администраторов, охватывающий программное обеспечение и работу с аппаратной частью компьютера.

10. Храмов, С. В. "Компьютерная безопасность и защищенные системы". - М.: Экотон-ЛТД, 2019.

Книга о безопасности компьютерных систем, включая аппаратные и программные аспекты.

**3.2. Основы сетевых технологий**

Сетевые технологии играют ключевую роль в современных информационных системах, обеспечивая связь и обмен данными между устройствами. В данной теме мы рассмотрим основы сетевых технологий, включая различные сетевые протоколы, их функции и архитектуру сетевых систем. Это знание поможет понять, как устроены современные коммуникационные сети, и как они работают.

1. Основы сетевых технологий

1.1. Определение сети

Сеть — это группа взаимосвязанных устройств, которые могут обмениваться данными. Сети могут быть локальными (LAN), широкими (WAN) или даже глобальными (Internet). Они обеспечивают разные типы связи, включая проводные и беспроводные.

1.2. Компоненты сетей

Основные компоненты компьютерных сетей включают:

- Устройства: компьютеры, серверы, маршрутизаторы, коммутаторы и точки доступа.

- Программное обеспечение: операционные системы, сетевые протоколы и приложения.

- Средства передачи данных: кабели, оптоволокно, радиоволны и т.д.

2. Архитектура сетей

2.1. Модель OSI

Модель OSI (Open Systems Interconnection) — это концептуальная модель, описывающая слои взаимодействия в сетях. Она состоит из семи слоев:

1. Физический уровень (Physical layer): передача данных по физическим носителям (кабели, радиосигналы и т.д.).

2. Канальный уровень (Data Link layer): управление доступом к среде передачи и формирование пакетов данных.

3. Сетевой уровень (Network layer): определение маршрутов и адресация данных.

4. Транспортный уровень (Transport layer): управление передачей данных между узлами (например, TCP, UDP).

5. Сеансовый уровень (Session layer): управление сессиями связи между приложениями.

6. Представительский уровень (Presentation layer): преобразование данных для обеспечения совместимости между различными форматами.

7. Прикладной уровень (Application layer): взаимодействие с пользовательскими приложениями.

2.2. Модель TCP/IP

Модель TCP/IP — это упрощенная модель, состоящая из четырех уровней:

1. Уровень доступа (Network Interface layer): совокупность технологий для доступа к физической среде.

2. Интернет-уровень (Internet layer): управление адресацией и маршрутизацией (например, IP).

3. Транспортный уровень (Transport layer): управление передачей данных (например, TCP, UDP).

4. Прикладной уровень (Application layer): сетевые приложения и услуги (например, HTTP, FTP, SMTP).

3. Сетевые протоколы

Протоколы — это набор правил и стандартов, регламентирующих процесс передачи данных в сети. Рассмотрим наиболее важные сетевые протоколы:

3.1. Протоколы доступа к среде передачи:

- Ethernet: стандарт для локальных сетей, определяющий способы передачи данных по кабелям.

- Wi-Fi: стандарт для беспроводных сетей, позволяющий устройствам подключаться без проводов.

3.2. Протоколы сетевого уровня:

- IP (Internet Protocol): основной протокол, обеспечивающий адресацию и маршрутизацию пакетов данных. Подразделяется на IPv4 и IPv6.

- ICMP (Internet Control Message Protocol): протокол для отправки управляющих сигналов и диагностики сетевых проблем.

3.3. Протоколы транспортного уровня:

- TCP (Transmission Control Protocol): протокол, обеспечивающий надежную передачу данных с контролем за их доставкой и порядком.

- UDP (User Datagram Protocol): протокол без установления соединения, используемый для передачи данных, где критична скорость (например, потоковое видео).

3.4. Протоколы прикладного уровня:

- HTTP (Hypertext Transfer Protocol): протокол для передачи гипертекстовых документов в Интернете.

- FTP (File Transfer Protocol): протокол для передачи файлов между клиентом и сервером.

- SMTP (Simple Mail Transfer Protocol): протокол для отправки электронной почты.

4. Проблемы и перспективы развития сетевых технологий

4.1. Проблемы

С развитием сетевых технологий возникают новые проблемы, такие как:

- Угрозы безопасности (вирусы, хакерские атаки).

- Проблемы совместимости между различными протоколами.

- Увеличение объема передаваемых данных и необходимость их обработки.

4.2. Перспективы

- Развитие технологий 5G и беспроводных сетей.

- Увеличение скорости и надежности передачи данных.

- Внедрение новейших стандартов безопасности и защиты данных.

Сетевые технологии составляют важную основу современного общества, обеспечивая связь и обмен информацией. Знание сетевых протоколов и архитектуры

***Контрольные вопросы***

1. Что такое компьютерная сеть и какие бывают её типы?

2. Опишите основные компоненты компьютерной сети.

3. В чем заключается основная функция модели OSI? Назовите ее слои.

4. Каковы основные отличия между моделью OSI и моделью TCP/IP?

5. Что такое сетевой протокол? Почему они важны для работы компьютерных сетей?

6. Что такое Ethernet и где он используется?

7. Какой протокол отвечает за адресацию и маршрутизацию данных в компьютерной сети? Опишите его виды.

8. Что такое TCP и UDP, и в чем их различия? Приведите примеры сценариев использования каждого из них.

9. Каковы основные функции протокола HTTP? Чем он отличается от FTP?

10. Какие проблемы безопасности могут возникнуть в сети и какие меры можно принять для их предотвращения?

***Практические задания***

Задание 1: Сравнение моделей OSI и TCP/IP

Инструкция:

1. Составьте таблицу, в которой сравните слои модели OSI и модели TCP/IP, указав соответствие между уровнями двух моделей и их основные функции.

2. Напишите краткое обоснование (150-200 слов) о том, какая из моделей вам кажется более удобной для понимания сетевых технологий и почему.

Задание 2: Исследование протоколов

Инструкция:

1. Выберите три сетевых протокола (например, HTTP, FTP, SMTP).

2. Для каждого протокола найдите следующую информацию:

- Назначение и область применения.

- Основные функции.

- Примеры программного обеспечения или устройств, использующих этот протокол.

3. Подготовьте краткий отчет (200-300 слов), обобщающий ваш анализ.

Задание 3: Анализ сетевой проблемы

Инструкция:

1. Представьте, что у вас возникла проблема с подключением к Интернету. Опишите шаги, которые вы предпримете для диагностики и решения проблемы.

2. Напишите список возможных причин проблемы и решения к ним.

Задание 4: Применение протоколов на практике

Инструкция:

1. Проведите исследование и найдите информацию о том, как работает и используется протокол DHCP (Dynamic Host Configuration Protocol).

2. Напишите краткое эссе (150-200 слов) о том, как этот протокол помогает управлять IP-адресами в сетях.

Задание 5: Составление схемы

Инструкция:

1. Создайте схему, изображающую взаимодействие различных уровней модели OSI с сетевыми устройствами и приложениями.

2. Каждый уровень должен содержать примеры протоколов и их основных функций, а также способы взаимодействия с другими уровнями и устройствами.

Задание 6: Защита сети

Инструкция:

1. Придумайте сценарий для защиты небольшой локальной сети (например, домашней или офисной).

2. Напишите план из пяти пунктов о том, какие меры безопасности вы можете предпринять для защиты этой сети от внешних угроз. Обоснуйте свой выбор.

Задание 7: Изучение технологий Wi-Fi

Инструкция:

1. Исследуйте стандарты Wi-Fi (например, 802.11a/b/g/n/ac/ax).

2. Составьте таблицу, в которой будут детализированы максимальная скорость передачи данных, диапазон частот и особенности каждого стандарта.

3. Напишите заключение о том, какие стандарты наиболее современные и для каких нужд они могут быть рекомендованы.

***Список источников для самостоятельного ознакомления***

1. Григорьев, С. В. Основы компьютерных сетей. — Москва: Издательство "БИИК", 2018.

Книга охватывает основные принципы построения компьютерных сетей, включая их архитектуру и протоколы.

2. Мещеряков, А. Р. Сетевые технологии: Учебник. — Санкт-Петербург: Питер, 2020.

В учебнике изложены основы сетевых технологий, рассмотрены различные протоколы и модели, а также принципы работы современных сетей.

3. Левин, А. А. Компьютерные сети: от теории к практике. — Москва: КНОРУС, 2019.

Книга содержит как теоретические основы, так и практические примеры применения сетевых технологий и протоколов.

4. Кузнецов, В. А. Информационные технологии: компьютерные сети и системы. — Москва: "Инфра-М", 2021.

Работа рассматривает различные аспекты компьютерных сетей, включая архитектуру и стандарты протоколов.

5. Барабанов, С. В. Основы сетевого администрирования. — Москва: "Эхо Москвы", 2020.

Книга предлагает практические рекомендации по настройке и управлению сетями, также рассматриваются сетевые протоколы.

6. Плужников, И. В. Исчерпывающий курс по компьютерным сетям. — Москва: ДМК Пресс, 2020.

Учебник охватывает основы работы компьютерных сетей, включая детальное изучение различных протоколов и технологий.

7. Санков, А. С. Сетевые технологии и протоколы. — Ростов-на-Дону: Феникс, 2021.

Издание посвящено сетевым технологиям и различным стандартам, используемым в современных сетях.

8. Шалевич, А. И. Сетевое программирование: Процессы, потоки, протоколы. — Москва: БХВ-Петербург, 2019.

Книга исследует сетевое программирование и протоколы, необходимые для создания сетевых приложений.

9. Ушаков, В. П. Компьютерные сети: Учебное пособие. — Нижний Новгород: ННГУ, 2018.

Учебное пособие, которое охватывает архитектуру и технологии современных компьютерных сетей.

10. Степанов, К. В. Элементы теории и практики сетевой безопасности. — Москва: "Деян", 2022.

Работа рассматривает основы безопасности сетей, методы защиты и управления доступом к данным.

**3.3. Безопасность информации**

Защита информации является важной частью современного общества, инициируемой развитием технологий и увеличением объемов хранимых и передаваемых данных. Угрозы безопасности могут проявляться в различной форме и затрагивать как личные, так и корпоративные данные. Понимание этих угроз и знание методов защиты информации позволят минимизировать риски и сохранить целостность информации.

1. Угрозы безопасности информации

1.1. Внешние угрозы

- Вирусы и вредоносное ПО: Программы, предназначенные для повреждения, уничтожения или кражи данных. Примеры: вирусы, трояны, шпионские программы, ransomware.

- Атаки через сеть: Хакеры могут использовать различные методы для доступа к защищенным системам. Популярные примеры — DDoS-атаки, MITM-атаки (атаки «человек посередине»).

- Фишинг: Метод социальной инженерии, при котором злоумышленники пытаются обманом получить конфиденциальные данные пользователей (например, пароли или номера кредитных карт).

1.2. Внутренние угрозы

- Неправомерный доступ: Сотрудники или пользователи, имеющие доступ к системе, могут злоупотреблять своими полномочиями.

- Ошибка пользователя: Неправильные действия или недосмотр со стороны пользователей могут привести к утечке информации или ее повреждению.

- Утрата данных: Потеря информации в результате технических неполадок или сбоя системы.

1.3. Физические угрозы

- Кража устройств: Устройство, содержащие важные данные, может быть украдено или потеряно.

- Природные катастрофы: Пожары, наводнения и другие стихийные бедствия могут привести к утрате данных.

2. Методы защиты информации

2.1. Технические методы

- Антивирусное ПО: Установка и регулярное обновление антивирусного программного обеспечения для защиты от вредоносных файлов.

- Межсетевые экраны (фаерволы): Использование фаерволов для контроля входящего и исходящего трафика и предотвращения несанкционированного доступа.

- Шифрование: Применение алгоритмов шифрования для защиты данных при передаче и хранении. Это включает симметричное и ассиметричное шифрование.

- Системы обнаружения и предотвращения вторжений (IDS/IPS): Мониторинг сети и систем на предмет подозрительной активности и автоматическая блокировка атак.

2.2. Организационные методы

- Политики безопасности: Разработка и внедрение документированных политик по управлению безопасностью информации.

- Обучение сотрудников: Проведение регулярных тренингов по вопросам безопасности для повышения осведомленности сотрудников о возможных угрозах.

- Журналирование и мониторинг: Ведущие журналы безопасности и мониторинг доступа к данным для быстрого выявления инцидентов.

2.3. Физические методы

- Контроль доступа: Использование систем доступа (ключи, карты, биометрические системы) для защиты физического доступа к устройствам и данным.

- Резервное копирование: Регулярное создание резервных копий данных для предотвращения их утраты в случае сбоев или атак.

- Безопасное хранение: Использование сейфов и защищенных серверных комнат для хранения критически важных данных.

3. Рекомендуемые практики

- Регулярные аудиты безопасности: Проведение периодических проверок систем на наличие уязвимостей.

- Обновление ПО: Регулярное обновление операционных систем и приложений для устранения известных уязвимостей.

- Создание многоуровневой защиты: Реализация комплексного подхода к безопасности с использованием нескольких методов защиты для повышения общей надежности.

Защита информации является жизненно важной задачей для организации и индивидов, особенно в условиях быстро меняющейся технологии и возрастающих угроз. Эффективное понимание угроз безопасности и применение соответствующих методов защиты поможет минимизировать риски и сохранить конфиденциальность, целостность и доступность данных. Регулярное обновление знаний в области безопасности информации и применение лучших практик станет залогом надежной защиты данных в будущем.

***Контрольные вопросы***

1. Основы безопасности информации

1. Что такое защита информации и почему она важна в современном обществе?

2. Каковы основные цели защиты информации?

3. Перечислите три ключевых аспекта защиты информации.

2. Угрозы безопасности

4. Какие существуют внешние угрозы для безопасности информации? Приведите примеры.

5. Что такое фишинг и каким образом он может угрожать безопасности данных?

6. Каковы основные внутренние угрозы для защиты информации?

7. Каковы потенциальные физические угрозы, связанные с хранением информации?

3. Методы защиты информации

8. Какие технические меры могут быть приняты для защиты информации? Приведите примеры.

9. Какую роль играет антивирусное ПО в защите информации?

10. Объясните, что такое шифрование и в каких случаях оно используется.

11. Какие организационные меры можно предпринять для повышения безопасности информации на предприятии?

12. Как осуществляется контроль доступа к информации и как он влияет на безопасность?

4. Применение методов защиты

13. Что такое полиция безопасности и как она помогает в защите информации?

14. Какие виды резервного копирования существуют и зачем они нужны?

15. Каковы преимущества использования многоуровневой защиты информации?

5. Практические аспекты безопасности

16. Какие практические шаги вы бы рекомендовали для повышения информационной безопасности в организации?

17. Как часто необходимо проводить аудит безопасности в компании и что он включает?

18. Почему важно держать программное обеспечение в актуальном состоянии?

6. Актуальные события и тренды

19. Какие современные угрозы для безопасности информации становятся наиболее актуальными в связи с развитием технологий?

20. Как новые технологии (например, облачные вычисления, IoT) влияют на подход к защите информации?

***Список источников для самостоятельного ознакомления***

1. Александров, И. Б. Основы защиты информации: Учебник. — Санкт-Петербург: Питер, 2021.

Учебник охватывает основные аспекты защиты информации и методов противодействия угрозам безопасности.

2. Барабанов, С. В. Информационная безопасность: от угроз до защиты. — Москва: "Издательство БХВ-Петербург", 2020.

Книга состоит из описания основных угроз безопасности информации и методов их предотвращения.

3. Конев, А. В. Защита информации в компьютерных системах. — Москва: "Гардарика", 2018.

Практическое руководство по защите данных и управления рисками в информационных системах.

4. Лебедев, А. А. Основы безопасности информации: Учебное пособие. — Москва: "КноРус", 2019.

Пособие, в котором рассматриваются основные принципы и методы защиты информации.

5. Николаев, П. Г. Информационная безопасность: Учебник для вузов. — Москва: "КноРус", 2021.

Систематизированный учебник по информационной безопасности, включая теорию и практику.

6. Овчинников, М. П. Информационная безопасность: угрозы, защита, управление. — Санкт-Петербург: БХВ-Петербург, 2020.

Книга рассматривает актуальные угрозы информации и методы их предотвращения с практическими примерами.

7. Панкратов, Е. Н. Безопасность информации: Эффективные методы и средства. — Москва: Республика, 2022.

Издание, охватывающее различные аспекты защиты информации с фокусом на современные методы.

8. Семёнкин, В. В. Технологии защиты информации. — Казань: Казанский университет, 2018.

Обзор современных технологий защиты информации, включая схемы шифрования и аутентификации.

9. Тимофеев, Г. Д. Социальная инженерия и защита информации. — Москва: "Эксмо", 2021.

Книга исследует методы социальной инженерии и способы защиты от них.

10. Шумилов, В. М. Угрозы безопасности информации и их минимизация. — Москва: "ТК Велби", 2020.

Рассматриваются актуальные угрозы безопасности информации и методы их минимизации.

**Глава 4. Инновации и современные технологии**

**4.1. Искусственный интеллект и машинное обучение**

Искусственный интеллект (ИИ) и машинное обучение (МО) стали одними из самых актуальных и обсуждаемых тем в области технологий за последние годы. Эти концепции трансформируют множество секторов, от медицины до финансов и образования. Понимание основных принципов ИИ и МО позволяет оценить их потенциал и влияние на будущее.

1. Искусственный интеллект: определения и основные концепции

1.1. Что такое искусственный интеллект?

Искусственный интеллект — это область компьютерных наук, посвященная созданию систем и технологий, способных выполнять задачи, которые обычно требуют человеческого интеллекта. Эти задачи могут включать:

- Обработку естественного языка

- Распознавание образов

- Принятие решений

- Игры и стратегии

1.2. Типы искусственного интеллекта

- Узкий (или слабый) ИИ: Специализированные системы, предназначенные для выполнения конкретных задач. Примеры включают чат-боты и рекомендательные системы.

- Общий (или сильный) ИИ: Гипотетическая форма ИИ, способная выполнять любые интеллектуальные задачи на уровне человека. На данный момент не существует.

- Суперинтеллект: ИИ, который превышает возможности человека в практически всех сферах. Это пока лишь теоретическая концепция.

2. Машинное обучение: основы и разновидности

2.1. Что такое машинное обучение?

Машинное обучение — это подсекция ИИ, которая фокусируется на разработке алгоритмов и моделей, позволяющих системам обучаться на основе данных и улучшать свою производительность с течением времени без явного программирования.

2.2. Основные виды машинного обучения

- Обучение с учителем: Модели обучаются на предварительно размеченных данных, где известен ввод и ожидаемый результат (например, классификация изображений).

- Обучение без учителя: Модели обнаруживают скрытые закономерности в неразмеченных данных без заданных выходов (например, кластеризация).

- Обучение с подкреплением: Модели обучаются через взаимодействие с окружающей средой и получение награды или наказания в зависимости от успешности выполненных действий (например, игры).

3. Применение искусственного интеллекта и машинного обучения

3.1. В медицине

- Диагностика заболеваний: Использование ИИ для анализа медицинских изображений и диагностики заболеваний (например, распознавание рака на рентгене).

- Персонализированная медицина: МО позволяет предсказывать, какие методы лечения наиболее эффективны для конкретного пациента, основываясь на его истории болезни и генетических данных.

3.2. В финансах

- Алгоритмическая торговля: ИИ и МО используются для прогнозирования рыночных трендов и автоматизации торговых операций.

- Обнаружение мошенничества: Системы ИИ фильтруют транзакции, выявляя аномалии и подозрительное поведение, что помогает предотвратить мошенничество.

3.3. В образовании

- Интеллектуальные обучающие системы: ИИ настраивает образовательный процесс в зависимости от потребностей и стиля обучения каждого студента.

- Анализ успеваемости: МО оценивает результаты работы учеников и предлагает индивидуальные рекомендации.

3.4. В производстве

- Предсказательная аналитика: Использование ИИ для прогнозирования поломок оборудования и оптимизации производственных процессов.

- Автоматизация и робототехника: Чат-боты и роботы помогают в автоматизации задач, что повышает производительность и снижает затраты.

3.5. В повседневной жизни

- Умные помощники: Использование виртуальных помощников (например, Siri, Alexa) для упрощения ежедневных задач.

- Рекомендательные системы: Платформы (например, Netflix, Amazon) используют ИИ для предлагать пользователям контент на основе их интересов и предпочтений.

Искусственный интеллект и машинное обучение представляют собой важные инструменты для преобразования множества секторов и улучшения мирового опыта. Их применение открывает перед человечеством новые перспективы, но также ставит перед ним и вызовы, такие как этические вопросы и необходимость обучения кадров в быстро меняющемся технологическом ландшафте. Учитывая их быстрое развитие, знание основных концепций ИИ и МО становится все более важным как для специалистов, так и для широкой аудитории.

Создание небольшой модели искусственного интеллекта на основе открытых данных — это практическая задача, которая может быть выполнена с помощью Python и его библиотек. В качестве простого примера мы можем создать модель для классификации цветов ириса с использованием набора данных Iris, который часто используется для образовательных целей.

Шаг 1: Установка необходимых библиотек

Для этого примера нам понадобятся следующие библиотеки:

- pandas: для работы с данными,

- scikit-learn: для создания модели машинного обучения,

- matplotlib и seaborn: для визуализации данных.

Убедитесь, что эти библиотеки установлены в вашей среде Python. Вы можете установить их, выполнив команду:

pip install pandas scikit-learn matplotlib seaborn

Шаг 2: Импорт библиотек и загрузка данных

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn.datasets import load\_iris

# Загрузка датасета Ирисов

iris = load\_iris()

iris\_df = pd.DataFrame(data=iris.data, columns=iris.feature\_names)

iris\_df['species'] = iris.target

iris\_df['species'] = iris\_df['species'].map({0: 'setosa', 1: 'versicolor', 2: 'virginica'})

Шаг 3: Анализ данных

Для начала вы можете посмотреть на данные, визуализировать их и понять, как устроены различные классы ирисов.

# Визуализация данных

sns.pairplot(iris\_df, hue='species')

plt.show()

Шаг 4: Подготовка данных для обучения модели

# Разделение данных на признаки и целевую переменную

X = iris.df.drop('species', axis=1)

y = iris\_df['species']

# Деление данных на обучающую и тестовую выборки

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Шаг 5: Создание и обучение модели

Создание модели

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

Обучение модели

model.fit(X\_train, y\_train)

Шаг 6: Оценка модели

# Прогнозирование на тестовой выборке

y\_pred = model.predict(X\_test)

# Оценка точности

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Accuracy: {accuracy:.2f}')

Отчет о классификации

print(classification\_report(y\_test, y\_pred))

Шаг 7: Использование модели

Теперь, когда у вас есть модель, можно использовать ее для прогнозирования:

# Пример нового образца

new\_sample = [[5.1, 3.5, 1.4, 0.2]] # Пример: длина и ширина чашелистика и лепестка

predicted\_species = model.predict(new\_sample)

print(f'Predicted species: {predicted\_species[0]}')

Это базовая модель машинного обучения, которая использует открытые данные о цветах ириса для создания классификатора. Вы можете экспериментировать с различными алгоритмами, настройками модели и исследовать данные более глубоко, чтобы улучшить качество прогноза.

Создадим другую модель, на этот раз для классификации данных о рукописных цифрах. Мы будем использовать набор данных MNIST, который содержит изображения рукописных цифр. Он широко используется для обучения и тестирования алгоритмов машинного обучения.

Шаг 1: Установка необходимых библиотек

Для работы с наборами данных и создания модели нам понадобятся следующие библиотеки:

- numpy: для работы с массивами,

- scikit-learn: для создания модели машинного обучения,

- matplotlib: для визуализации.

Убедитесь, что эти библиотеки установлены в вашей среде Python. Вы можете установить их, выполнив команду:

pip install numpy scikit-learn matplotlib

Шаг 2: Импорт библиотек и загрузка данных

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import fetch\_openml

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

# Загрузка набора данных MNIST

mnist = fetch\_openml('mnist\_784', version=1)

X, y = mnist['data'], mnist['target'].astype(np.uint8)

### Шаг 3: Анализ данных

Вы можете визуализировать несколько примеров, чтобы понять данные лучше.

# Визуализация первых 10 цифр

def plot\_digits(X, y):

plt.figure(figsize=(10, 5))

for index in range(10):

plt.subplot(2, 5, index + 1)

plt.imshow(X.iloc[index].values.reshape(28, 28), cmap='binary')

plt.title(y[index])

plt.axis('off')

plt.show()

plot\_digits(X, y)

Шаг 4: Подготовка данных для обучения модели

# Деление данных на обучающую и тестовую выборки

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Шаг 5: Создание и обучение модели

# Создание модели

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

# Обучение модели

model.fit(X\_train, y\_train)

### Шаг 6: Оценка модели

# Прогнозирование на тестовой выборке

y\_pred = model.predict(X\_test)

# Оценка точности

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Accuracy: {accuracy:.2f}')

# Отчет о классификации

print(classification\_report(y\_test, y\_pred))

### Шаг 7: Использование модели

Теперь, когда у вас есть модель, вы можете использовать ее для прогнозирования.

# Пример нового образца (получение цифры из тестового набора)

sample\_image = X\_test.iloc[0].values.reshape(1, -1) # Получаем первую цифру из тестового набора

predicted\_digit = model.predict(sample\_image)

plt.imshow(sample\_image.reshape(28, 28), cmap='binary')

plt.title(f'Predicted digit: {predicted\_digit[0]}')

plt.axis('off')

plt.show()

Этот пример демонстрирует, как создать модель для классификации рукописных цифр с использованием открытого набора данных MNIST. Вы можете улучшить модель, используя другие алгоритмы, изменяя параметры или работая с другими подходами к предобработке данных.

***Контрольные вопросы***

1. Основы искусственного интеллекта

1. Что такое искусственный интеллект и какие его основные цели?

2. В чем разница между узким (слабым) ИИ и общим (сильным) ИИ?

3. Приведите три примера применения узкого ИИ в повседневной жизни.

2. Машинное обучение

4. Определите, что такое машинное обучение и как оно связано с искусственным интеллектом.

5. Назовите и кратко объясните три основных типа машинного обучения.

6. Чем отличается обучение с учителем от обучения без учителя?

3. Применение ИИ и МО

7. Как ИИ и МО используются в медицине? Приведите два примера.

8. Какие приложения ИИ и МО вы можете назвать в области финансов, и как они помогают в борьбе с мошенничеством?

9. Как интеллектуальные обучающие системы могут изменить подход к образованию?

4. Этические и социальные аспекты

10. Какие этические вопросы могут возникать в связи с использованием искусственного интеллекта?

11. Какое влияние ИИ может оказать на рынок труда и занятость?

***Практические задания***

Задание 1: Исследование применения

Выберите одну из сфер применения ИИ и машинного обучения (например, медицина, финансы, образование) и подготовьте краткий отчет (300-500 слов) о текущих трендах и возможностях, которые открываются благодаря этим технологиям.

Задание 2: Анализ примеров

Найдите и проанализируйте два примера успешного применения ИИ в различных отраслях. Укажите, какие технологии были использованы, какие результаты были достигнуты и какую ценность это принесло.

Задание 3: Кейс-стадии

Представьте, что вы работаете в компании, которая планирует внедрить систему ИИ в одном из своих бизнес-процессов. Определите следующий:

- Какую проблему эта система должна решить?

- Какие данные будут необходимы для обучения модели?

- Как вы планируете оценить ее эффективность?

Задание 4: Ситуационный анализ

Выберите известную компанию, активно использующую ИИ или машинное обучение, и проанализируйте, как это влияет на бизнес-процессы и взаимодействие с клиентами. Подготовьте презентацию на эту тему.

Задание 5: Рефлексия

Напишите краткое эссе (300-400 слов) на тему "Будущее искусственного интеллекта: угрозы и возможности". Обсудите как положительные, так и отрицательные аспекты, которые могут возникнуть с развитием ИИ.

**4.2. Интернет вещей (IoT): Принципы работы IoT и примеры успешных проектов**

Интернет вещей (IoT) представляет собой концепцию, в которой физические объекты ("вещи") подключены к интернету и могут обмениваться данными друг с другом и с центральными системами. IoT охватывает широкий спектр технологий и приложений, включая дома, города, транспорт, здравоохранение и промышленность. В этом разделе мы рассмотрим основные принципы работы IoT и приведем примеры успешных проектов.

Принципы работы Интернет вещей (IoT)

1. Сенсоры и устройства:

- В центре IoT являются сенсоры и устройства, которые собирают данные из окружающей среды. Это могут быть датчики температуры, влажности, движения, изображения и многие другие.

- Устройства могут быть как простыми (например, термостаты), так и сложными (например, автономные роботы).

2. Подключение и протоколы:

- Устройства IoT подключены к интернету через различные сети — Wi-Fi, Bluetooth, Zigbee, LoRaWAN и другие.

- Используются специальные протоколы передачи данных, такие как MQTT, CoAP и HTTP, которые обеспечивают эффективную и надежную передачу данных.

3. Обработка и анализ данных:

- Собранные данные передаются на облачные платформы или локальные серверы, где они обрабатываются и анализируются.

- Для анализа часто применяются технологии больших данных и машинного обучения, которые помогают извлечь полезные инсайты из огромных объемов информации.

4. Хранение данных:

- Данные могут храниться в облаке или в локальных системах хранения, и доступ к ним может быть организован через API.

- Эффективное хранение и управление данными — важный аспект активов IoT, который влияет на производительность и стоимость.

5. Интерфейсы и взаимодействие:

- Пользователи могут взаимодействовать с устройствами IoT через мобильные приложения, веб-интерфейсы или голосовых помощников.

- Автоматизация и удаленное управление являются важными компонентами IoT-экосистемы.

6. Безопасность и конфиденциальность:

- Безопасность является критическим аспектом IoT, так как устройства могут подвергаться атакам или взломам.

- Механизмы аутентификации, шифрования и регулярные обновления программного обеспечения играют ключевую роль в обеспечении безопасности IoT-устройств.

Примеры успешных проектов IoT

1. Умные города:

- В некоторых городах внедрены IoT-решения для управления трафиком, освещением, вывозом мусора и другим муниципальным обслуживанием.

- Пример: Сингапур использует IoT для управления дорожным движением и улучшения общественного транспорта, собирая данные о плотности трафика в реальном времени и оптимизируя маршруты.

2. Умный дом:

- Устройства умного дома, такие как термостаты, камеры безопасности, датчики движения и освещения, позволяют пользователям управлять своим домом из любой точки с помощью смартфонов или голосовых команд.

- Пример: Google Nest — это экосистема умных устройств, которые помогают управлять температурой, безопасностью и экономить электроэнергию.

3. Здравоохранение:

- IoT-технологии используются для мониторинга здоровья пациентов, что позволяет врачам собирать данные о состоянии здоровья в режиме реального времени.

- Пример: Устройства, такие как фитнес-браслеты или умные часы (например, Apple Watch), фиксируют жизненные показатели и могут отправлять алармы врачам при критических изменениях.

4. Сельское хозяйство:

- IoT-применения в агрономии включают использование датчиков для мониторинга состояния почвы, уровня влажности, температуры и других факторов, влияющих на урожай.

- Пример: Компания CropX предлагает решения для точного земледелия, позволяя фермерам оптимизировать полив и использование удобрений на основе данных из сенсоров.

5. Промышленный IoT (IIoT):

- В промышленности IoT используется для мониторинга оборудования, предсказательного обслуживания и оптимизации производственных процессов.

- Пример: GE использует IIoT для анализа данных с промышленных машин, что позволяет предсказывать возможные поломки и производить обслуживание до их появления.

Интернет вещей значительно меняет наш образ жизни и работу во многих отраслях, обеспечивая возможность эффективного мониторинга, анализа и управления данными. Успешные проекты IoT демонстрируют его огромный потенциал, но также подчеркивают важность безопасности и защиты данных. В будущее ожидается дальнейшее развитие технологий IoT и расширение их применения, что создаст новые возможности для бизнеса и общества в целом.

***Контрольные вопросы***

1. Определение понятий:

- Что такое Интернет вещей (IoT)?

- Перечислите основные компоненты системы IoT.

2. Принципы работы:

- Какую роль играют сенсоры в системе IoT?

- Какие технологии используются для передачи данных в IoT?

3. Обработка данных:

- Как обрабатываются и анализируются данные, собранные с помощью IoT-устройств?

- Какие методы и технологии могут использоваться для анализа данных в IoT?

4. Примеры применения:

- Назовите три успешных проекта, использующих IoT, и кратко опишите их.

- Как Интернет вещей может улучшить жизнь в умных городах?

5. Безопасность:

- Какие основные угрозы безопасности существуют для IoT-устройств?

- Какие методы могут быть применены для повышения безопасности IoT-устройств?

6. Будущее IoT:

- Как вы видите будущее Интернет вещей?

- Какие области, по вашему мнению, могут получить новое развитие благодаря IoT в следующем десятилетии?

***Практические задания***

1. Исследовательское задание:

- Найдите и проанализируйте один конкретный проект в области IoT (например, умный дом, агрономия, здравоохранение). Подготовьте презентацию, в которой будет описано:

- Цели проекта.

- Используемые технологии и устройства.

- Результаты или достигнутые преимущества.

2. Применение знаний:

- Предложите идею нового IoT-приложения, которое может быть полезно в повседневной жизни. Опишите:

- Как оно будет работать.

- Какие устройства и технологии будут использованы.

- Какое влияние оно окажет на пользователей.

3. Кейс-стадия:

- Рассмотрите ситуацию, когда IoT-устройству (например, умному термостату) не удалось подключиться к сети. Напишите краткий анализ возможных причин этой проблемы и предложите решения.

4. Групповая работа:

- В группе разработайте простое IoT-решение на бумаге или в одиночном проекте. Опишите:

- Проблему, которую оно решает.

- Устройства и технологии, которые будут использоваться.

- Возможные риски и меры по их минимизации.

5. Визуализация данных:

- Создайте график или диаграмму, иллюстрирующую, как данные движутся в системе IoT от сенсора до конечного пользователя. Включите все этапы: сбор данных, передача, обработка и визуализация.

***Список источников для самостоятельного ознакомления***

*Книги*

1. Смит, Р. (2017). Интернет вещей. Как технологии меняют наш мир. М.: АСТ.

2. Гнедов, В.Ю. (2018). Интернет вещей: новый взгляд на технологии и бизнес. М.: ЛКИ.

3. Корнеев, Д.Ю. (2019). Интернет вещей: от проекта до внедрения. М.: Наука.

4. Давыдов, В. (2020). Интернет вещей и умный дом: полное руководство. М.: Питер.

5. Шарков, А. (2021). Интернет вещей: практическое руководство по проектированию и внедрению. М.: БХВ-Петербург.

*Статьи и журналы*

1. Захаров, А. (2021). "Интернет вещей: технологии и применения". IT-Expert, 3(15), 56-63.

2. Соловьев, А.Н. (2020). "Безопасность технологий Интернет вещей". Журнал информационной безопасности, 1(2), 12-19.

3. Кузнецов, И. (2019). "Умные города: реальность и перспективы". Современные технологии управления, 2(4), 22-30.

4. Петров, С. (2022). "Разработка IoT-решений: путь от идеи к реализации". Научные записки НаУКМА по информатике, 4(1), 45-52.

*Онлайн-ресурсы*

1. "Интернет вещей: что это и как работает" — статья на сайте https://habr.com/ru/company/rostelecom/blog/505576/

2. Тема: "Интернет вещей (IoT)" — раздел на сайте https://www.innopolis.university/

3. "Интернет вещей и его применение в здравоохранении" — статья на сайте https://www.rscf.ru/

**4.3. Будущее технологий**

Современный мир быстро меняется благодаря новым технологиям, которые меняют не только сами технологии, но и подходы к бизнесу, организации жизни, взаимодействию между людьми и государствами. В этом контексте особое внимание вызывают такие инновации, как блокчейн и квантовые вычисления. Данная тема рассмотрит главные аспекты и потенциал этих технологий, их влияние на различные сферы жизни и бизнеса, а также вызовы и возможности, которые они приносят.

1. Блокчейн

1.1 Определение и принцип работы

- Блокчейн представляет собой распределенную базу данных, хранящую информацию в виде цепочки блоков. Каждый блок содержит данные, временную метку и криптографическую подпись предыдущего блока.

- Данная технология обеспечивает высокий уровень безопасности и прозрачности, делая подделку данных практически невозможной.

1.2 Применения об блокчейн

- Финансовый сектор: Криптовалюты, токены, децентрализованные финансовые (DeFi) протоколы.

- Логистика и цепочки поставок: Отслеживание товаров, управление запасами и прозрачность операций.

- Государственные услуги: Электронное голосование, управление идентификацией граждан, улучшение прозрачности в государственном секторе.

- Система управления правами собственности: Регистрация прав на недвижимость, интеллектуальную собственность и т.д.

1.3 Будущее и тренды

- Развитие смешанных и гибридных моделей управления с использованием блокчейн-технологий.

- Увеличение числа государств, рассматривающих внедрение цифровых валют на основе блокчейн.

- Рост интереса к смарт-контрактам и расширение их применения в различных отраслях.

2. Квантовые вычисления

2.1 Определение и принципы работы

- Квантовые вычисления опираются на принципы квантовой механики и используют квантовые биты (кубиты) для обработки информации.

- Квантовые компьютеры способны выполнять вычисления значительно быстрее, чем традиционные компьютеры, за счет одновременной обработки множества состояний.

2.2 Применения квантовых вычислений

- Научные исследования: Моделирование сложных молекул для химических соединений и лекарств.

- Оптимизация: Решение сложных задач о маршрутизации и оптимизации ресурсов в логистике.

- Шифрование и безопасность: Разработка новых методов шифрования, защищающих данные от потенциальных атак.

2.3 Будущее и тренды

- Увеличение производительности квантовых компьютеров и их достижения в коммерческом использовании.

- Потенциальное изменение парадигмы разработки алгоритмов и программного обеспечения.

- Разработка гибридных систем, сочетающих квантовые и классические вычисления для решения задач.

3. Вызовы и возможности

- Этические и правовые аспекты: questioned о конфиденциальности, безопасности данных и защите прав личности в условиях использования новых технологий.

- Экономические факторы: Адаптация бизнес-моделей и стратегий к требованиям новой технологической реальности.

- Образование и подготовка кадров: Необходимость пересмотра образовательных программ для подготовки специалистов в области блокчейн-технологий и квантовых вычислений.

Блокчейн и квантовые вычисления представляют собой значимые технологические тренды, которые могут оказать долгое и глубокое влияние на многие сферы жизни. Важно помнить, что, помимо возможностей, данные технологии также приносят и новые вызовы, требующие тщательного изучения и осмысленного подхода. Для успешного внедрения и использования требуется не только технологическая экспертиза, но и честный диалог между бизнесом, государством и обществом, чтобы адаптировать технологические достижения к нуждам людей и создавать безопасное и справедливое будущее.

***Контрольные вопросы***

1. Общие понятия:

- Определите, что такое блокчейн и как он работает.

- Объясните, что такое кубиты и как они отличаются от битов в традиционных вычислениях.

2. Применение блокчейна:

- Назовите три сферы, в которых технология блокчейн может быть применена, и опишите, как она улучшает процессы в каждой из них.

- Какие преимущества предоставляет использование смарт-контрактов в бизнесе?

3. Квантовые вычисления:

- В чем заключается ключевое отличие между классическими и квантовыми компьютерами?

- Приведите примеры потенциальных применений квантовых вычислений в различных сферах.

4. Вызовы и возможности:

- Какие основные вызовы стоят перед внедрением технологий блокчейн и квантовых вычислений в современное общество?

- Как эти технологии могут повлиять на экономику и рынок труда в будущем?

5. Этика и безопасность:

- Обсудите возможные этические и правовые проблемы, связанные с использованием блокчейна и квантовых вычислений.

- Каковы риски безопасности, связанные с квантовыми вычислениями, и как можно их минимизировать?

***Практические задания***

1. Исследовательская работа:

- Напишите краткий отчет (от 2 до 3 страниц) о потенциальных рисках и преимуществах использования квантовых вычислений в сфере кибербезопасности.

2. Анализ кейсов:

- Найдите два примера компаний или проектов, которые используют блокчейн в своей деятельности. Проанализируйте их успешность и возможные направления для дальнейшего улучшения.

3. Групповая дискуссия:

- Проведите дискуссию в группе о будущем технологий: как вы считаете, какие из них будут иметь наибольшее влияние на общество в следующие 10 лет? Приведите аргументы.

4. Разработка концепции:

- Разработайте концепцию проекта, который интегрирует технологию блокчейн и квантовые вычисления для решения конкретной проблемы. Опишите целевую аудиторию, цель проекта и предполагаемые результаты.

5. Презентация:

- Подготовьте 10-минутную презентацию на тему "Как блокчейн и квантовые вычисления могут изменить бизнес-процессы в различных отраслях". Используйте графики и примеры для иллюстрации своих аргументов.

***Список источников для самостоятельного ознакомления***

*Книги*

1. Косарев, Ю. А. (2018). Блокчейн и криптовалюты: от теории к практике. М.: Альпина Паблишер.

2. Ковалев, А. (2020). Квантовые вычисления: введение в теорию и практику. М.: БХВ-Петербург.

3. Савельев, И. (2021). Блокчейн: Как эта технология меняет бизнес и мир. М.: Питер.

4. Левин, А. А. (2019). Основы теории квантовых вычислений. М.: Физматлит.

5. Пронин, Д. (2022). Криптовалюты и блокчейн: от основ до применения в бизнесе. М.: Кучково Pole.

*Статьи и журналы*

1. Петров, С. (2020). "Блокчейн как драйвер цифровой трансформации". Журнал информационных технологий, 2(5), 34-41.

2. Иванова, Н. В. (2021). "Перспективы применения квантовых вычислений в финансах". Научные записки КФУ, 3(1), 57-63.

3. Смирнова, Е. (2022). "Этические аспекты внедрения блокчейн-технологий". Этика и право, 1(2), 78-85.

4. Захаров, А. (2020). "Квантовые технологии: от научных теорий к реальным применениям". Научный журнал международной ассоциации исследователей, 4(3), 45-52.

*Онлайн-ресурсы*

1. Статья: "Что такое блокчейн? Полное описание технологии". Доступно на сайте https://habr.com/ru/post/348200/

2. Тема: "Введение в квантовые вычисления". Онлайн-курс на платформе https://stepik.org/course/17034/info

3. "Блокчейн и его применение в бизнесе" — видео лекция на YouTube: https://www.youtube.com/watch?v=7g8NFFr\_SGg

4. "Технология блокчейн и ее влияние на мир" — статья на сайте https://www.rbc.ru/technology\_and\_media/23/07/2021/60fe12f89a7947cbb43c0c7d

**Глава 5. Научно-исследовательская работа**

**Тема 5.1. Основы научного метода**

Научный метод является систематическим способом исследования, используемым для получения и проверки знаний. Он основан на наблюдениях, экспериментах и логическом анализе. Понимание этапов научного метода критически важно для успешного проведения исследований в любых областях науки, так как это позволяет формулировать гипотезы, разрабатывать экспериментальные процедуры, анализировать данные и делать выводы. Рассмотрим основные этапы научного исследования.

1. Формулирование проблемы

- Определение проблемы: На первом этапе исследователь выделяет конкретную проблему или вопрос, который нуждается в исследовании. Это может быть идея, основанная на предшествующих знаниях или наблюдениях.

- Уточнение объекта исследования: Определение объектов и субъектов, которые будут изучаться в рамках исследования.

2. Изучение литературы

- Сбор информации: Исследователь анализирует существующие исследования, литературу и теории, связанные с проблемой. Это дает возможность выявить пробелы в знаниях и построить теоретическую основу для своего исследования.

- Составление библиографии: Запись необходимых источников и их краткое описание для дальнейшего обращения.

3. Постановка гипотезы

- Формулирование гипотезы: На основе собранной информации исследователь разрабатывает гипотезу — предположение о возможном результате исследования, которое можно проверить.

- Определение переменных: Установление зависимых и независимых переменных, которые будут исследоваться в эксперименте.

4. Разработка методологии

- Выбор методов и дизайна исследования: Определение, какие методы исследования будут использоваться (количественные, качественные или смешанные). Разработка дизайна исследования, который обеспечит получение необходимых данных.

- Определение выборки: Установление, какие объекты или участники будут включены в исследование и каким образом они будут выбраны.

5. Сбор данных

- Проведение экспериментов или опросов: Реализация разработанного метода сбора данных, что может включать эксперименты в лаборатории, опросы, наблюдения или анализ существующих данных.

- Регистрация данных: Систематизация собранной информации для дальнейшего анализа.

6. Анализ данных

- Обработка и интерпретация результатов: Использование статистических и аналитических методов для обработки собранных данных. Выявление закономерностей, тенденций и аномалий в данных.

- Визуализация данных: Презентация результатов в виде графиков, таблиц или диаграмм для упрощения их интерпретации.

7. Формулирование выводов

- Оценка гипотезы: На основе анализа данных исследователь решает, подтверждается ли гипотеза или отвергается.

- Обсуждение результатов: Обсуждение значимости полученных результатов, их вклада в существующие знания и возможных практических применений.

8. Документация и публикация результатов

- Подготовка отчета о исследовании: Составление научной работы, которая включает введение, методологию, результаты, обсуждение и выводы.

- Публикация исследования: Представление результатов в научных журналах, на конференциях или других платформах с целью донесения информации до научного сообщества.

Научный метод является важнейшим инструментом в процессе исследования. Каждый из вышеперечисленных этапов играет ключевую роль и требует тщательной работы. Осознание и соблюдение этих этапов позволяет получить объективные и воспроизводимые результаты, что, в свою очередь, способствует дальнейшему развитию науки. Понимание основ научного метода также является важной компетенцией для студентов и молодых исследователей, стремящихся внести свой вклад в свою область знания.

**Тема 5.2. Методология проведения исследования**

Методология исследования представляет собой совокупность методов и принципов, используемых для организации и проведения научного исследования. Она включает выбор подходящих методов сбора данных и методов их анализа. Правильный выбор методологии зависит от цели исследования, типа данных и контекста. В данной теме мы рассмотрим основные методы сбора данных и их анализа, а также их применение в различных областях.

1. Методы сбора данных

Методы сбора данных можно разделить на несколько категорий:

1.1. Количественные методы

Количественные методы направлены на получение численных данных, которые можно подвергать статистическому анализу.

- Опросы и анкетирование: Этот метод позволяет собирать данные у большой группы респондентов. Анкеты могут быть как бумажными, так и электронными, включать закрытые и открытые вопросы. Применяется для изучения мнений, предпочтений или характеристик группы.

- Эксперименты: Включают контрольные и экспериментальные группы. Экспериментатор манипулирует независимой переменной и фиксирует изменения в зависимой. Позволяет установить причинно-следственные связи.

- Анализ существующих данных (secondary data analysis): Использование уже существующих данных, собранных в предыдущих исследованиях или статистических базах, для анализа и интерпретации.

1.2. Качественные методы

Качественные методы сосредоточены на понимании глубины и контекста изучаемого явления.

- Интервью: Глубокие интервью позволяют получить детальную информацию о мнениях и опыте респондентов. Они могут быть структурированными, полуструктурированными или неструктурированными.

- Фокус-группы: Вовлечение небольшой группы людей для обсуждения определенной темы, что позволяет выявить различные перспективы и мнения.

- Наблюдение: Исследователь наблюдает за поведением людей в естественных условиях. Это может быть как скрытое, так и открытое наблюдение.

- Анализ текстов и документов: Изучение и интерпретация текстовых данных, таких как статьи, отчеты, сообщения в социальных сетях и т. д.

1.3. Смешанные методы

Смешанные методы подразумевают комбинирование количественных и качественных методов для более полного анализа.

- Комбинированные опросы: Использование как закрытых, так и открытых вопросов в одном исследовании.

- Последовательное использование методов: Сначала проводится количественное исследование для определения общих тенденций, а затем — качественное для глубокого изучения найденных аспектов.

2. Методы анализа данных

После сбора данных следующим этапом является их анализ, который может включать следующие методы:

2.1. Количественные методы анализа

- Статистический анализ: Использование статистических методов (описательной статистики, корреляционного анализа, регрессионного анализа) для работы с числовыми данными. Позволяет выявить тенденции, связи и закономерности.

- Проверка гипотез: Статистические тесты (например, t-тест, ANOVA) используются для определения значимости результатов.

- Визуализация данных: Графики, диаграммы и таблицы помогают наглядно представить результаты и сделать их более понятными.

2.2. Качественные методы анализа

- Контент-анализ: Систематическое изучение содержимого текстов и документов для выявления тем, паттернов и значений.

- Тематический анализ: Методика, позволяющая идентифицировать, анализировать и отчетливо описывать темы, возникающие в данных.

- Кодирование данных: Присвоение меток категориям данных для упрощения анализа и последующей интерпретации.

2.3. Смешанные методы анализа

Смешанные методы анализа предполагают использование как количественных, так и качественных данных для достижения более целостного понимания исследуемой проблемы.

- Интеграция результатов: Совмещение и комплексное изучение данных различных типов для получения более глубокой информации.

- Метапоединение: Объединение количественных и качественных выводов для комплексного обобщения результатов.

Методология исследования является критически важным компонентом научного изучения. Правильный выбор методов сбора и анализа данных помогает исследователю получить надежные и обоснованные результаты. Каждый метод имеет свои сильные и слабые стороны, и комбинация различных подходов может привести к более точным и полным выводам. Понимание этих методов и их применение в своей работе — ключевой аспект для успешного проведения исследований в любой научной дисциплине.

**Тема 5.3. Обработка данных и представление результатов**

Обработка данных и представление результатов являются ключевыми этапами в научном исследовании. На этом этапе исследователь анализирует собранные данные, использует статистические методы для обработки информации и представляет результаты в понятной и доступной форме. Эффективная структура отчетов и использование статистики позволяют донести результаты до аудитории, помочь в интерпретации данных и установить практическую значимость исследования. В данной теме мы рассмотрим основы статистики, методы обработки данных и принципы написания отчетов.

1. Обработка данных

Обработка данных включает в себя несколько важных этапов:

1.1. Подготовка данных

- Очистка данных: Удаление или исправление ошибок и недостающих значений в данных. Это важный шаг для обеспечения достоверности анализа. Методы включают проверку на выбросы, пропуски и аномалии.

- Кодирование данных: Присвоение числовых кодов категориальным переменным (например, “мужчина” = 1, “женщина” = 2) для упрощения анализа.

1.2. Описание данных

- Дескриптивная статистика: Использование основных статистических показателей для описания характеристик датасета:

- Среднее: Мера центральной тенденции, которая отображает среднее значение данных.

- Медиана: Значение, делящее набор данных пополам, когда они упорядочены.

- Мода: Наиболее часто встречающееся значение в наборе данных.

- Стандартное отклонение и дисперсия: Меры разброса данных, показывающие, насколько данные удалены от среднего значения.

1.3. Анализ данных

- Статистические тесты: Применение методов статистического анализа для проверки гипотез, таких как t-тест, ANOVA, хи-квадрат и др. Эти тесты помогают определить значимость полученных результатов.

- Корреляционный анализ: Оценка связи между переменными с использованием коэффициента корреляции, который показывает, насколько сильно и в каком направлении связаны две переменные.

2. Представление результатов

После обработки данных необходимо четко и структурированно представить результаты исследования.

2.1. Использование графиков и таблиц

- Графики: Визуализация данных с использованием столбчатых диаграмм, круговых диаграмм, линейных графиков и других видов графики помогает донести информацию до аудитории:

- Столбчатые диаграммы: Подходят для сравнения категорий.

- Линейные графики: Идеальны для отображения изменений со временем.

- Круговые диаграммы: Хороши для иллюстрации составных частей целого.

- Таблицы: Использование таблиц позволяет представить данные в упрощенном виде, делая акцент на числовых значениях и их сравнении.

2.2. Структура отчета

Отчет о проведенном исследовании должен включать следующие разделы:

1. Введение: Краткое описание темы, постановка проблемы и цели исследования.

2. Методы: Подробное изложение методов сбора данных, выборки и анализа.

3. Результаты: Основные находки, представленные в виде графиков, таблиц и текстовых описаний.

4. Обсуждение: Интерпретация результатов, анализ их значимости, сравнение с предыдущими исследованиями, обозначение ограничений работы.

5. Выводы: Краткое резюме ключевых результатов и рекомендаций для практики или будущих исследований.

6. Список литературы: Перечень источников, на которые опирался исследователь.

2.3. Язык и стиль

- Тематика и целевая аудитория: Учет уровня подготовки и интересов аудитории при написании отчета. Использование доступного языка и отсутствие избыточной терминологии.

- Четкость и краткость: Логичная структура и последовательное изложение помогают читателю лучше понять представленные материалы.

Обработка данных и представление результатов являются важными аспектами научного исследования. Основы статистики помогают исследователю правильно анализировать и интерпретировать данные, в то время как грамотное написание отчетов обеспечивает доступность и понятность результатов для научного сообщества и широкой общественности. Понимание и применение этих принципов играет ключевую роль в успешном проведении исследований и их последующем влиянии на соответствующую область науки.

***Список источников для самостоятельного ознакомления***

*Книги:*

- Бейли, К. Д. (1994). Методы социологического исследования. Москва: Издательство "Наука".

- Керли, Д. (2015). Основы научного исследования: Методология и практика. М.: Издательство "Юрайт".

- Хантер, Д. (2017). Научный метод: Пошаговое руководство для традиционных и современных подходов. Нью-Йорк: Springer.

- Кросс, Х. (2018). Анализ данных: Методология, технологии, статистические методы. С.-Петербург: Издательство "БХВ-Петербург".

- Глазырин, В. А., & Селяев, В. И. (2016). Статистика и исследование операций в экономике. Москва: КНОРУС.

*Статьи:*

- Дворянкин, Д. Н. (2019). Научный метод: Основные этапы и принципы. Российский журнал научных исследований, 7(2), 12-22.

- Виноградова, Т. И. (2020). Применение смешанных методов в социально-гуманитарных исследованиях. Социология образования, 5(3), 45-52.

- Петров, A. (2021). Статистические методы для анализа данных: современные подходы. Журнал количественных исследований, 8(1), 78-88.

*Учебные пособия:*

- Романов, И. А. (2022). Методология и методы научных исследований: Учебное пособие для студентов и аспирантов. Москва: Издательство "Финансовый университет".

- Филиппов, В. Г. (2021). Статистика и обработка данных: Учебное пособие для студентов экономических специальностей. С.-Петербург: Издательство "Питер".

*Онлайн-ресурсы:*

- Курс "Основы научного метода" на Coursera. Доступно по адресу: [Coursera.org](https://www.coursera.org)

- Вебинар "Статистика для исследователей" от исследовательского центра "Даталаб". Запись доступна на [Datalab.ru](https://www.datalab.ru/webinars)

**Глава 6. Творческие проекты**

**6.1. Генерация идей и их реализация**

Генерация идей является одним из ключевых этапов в процессе разработки новых продуктов, услуг или решений для решения возникших проблем. Эффективные методы генерации идей, такие как мозговой штурм и различные техники креативного мышления, помогают в стимулировании творческого потенциала участников, создании инновационных концепций и обеспечении оригинальности решений. В данной теме мы рассмотрим основные методы генерации идей, их принципы и применения в различных контекстах.

1. Мозговой штурм

Мозговой штурм — это групповая техника, предназначенная для генерации идей и решений. Метод был разработан Алеком Осборном в 1940-х годах и с тех пор стал одним из самых популярных средств креативной деятельности.

1.1. Принципы мозгового штурма

- Отсутствие критики: Во время мозгового штурма критика идей участников не допускается. Это создает атмосферу свободы, где каждый может предложить свою мысль без страха быть осуждённым.

- Стремление к количеству: Количество предложенных идей более важно, чем их качество на начальном этапе. Чем больше идей, тем выше вероятность появления действительно оригинальных и ценных концепций.

- Строительство на идеях других: Участники поощряются к тому, чтобы развивать и модифицировать идеи, предложенные другими. Это способствует созданию комбинированных решений и новых подходов.

1.2. Процесс мозгового штурма

1. Определение проблемы или темы: Ясно сформулируйте вопрос, на который нужно найти ответ.

2. Сбор группы: Соберите diverse (разнообразную) команду участников с разными навыками и опытом.

3. Обсуждение: Позвольте участникам высказывать свои идеи. Записывайте каждую идею на видном месте, чтобы все могли их видеть.

4. Завершение сессии: После окончания мозгового штурма отфильтруйте идеи и выберите наиболее перспективные для дальнейшего анализа и реализации.

1.3. Варианты мозгового штурма

- Мозговой штурм в тишине: Участники записывают свои идеи самостоятельно на бумаге или в цифровом формате, прежде чем делиться ими с группой. Это помогает предотвратить влияние группового давления.

- Электронный мозговой штурм: Идеи генерируются и обмениваются через онлайн-платформы, что может быть полезно для распределенных команд.

2. Методы креативного мышления

Креативное мышление охватывает различные подходы, которые помогают стимулировать новые идеи и решения. Вот несколько популярных методов:

2.1. Метод "6 хат"

Разработанный Эдвардом де Боно, этот метод предполагает использование шести "шляп", каждая из которых символизирует разные виды мышления:

- Белая шляпа: Факты и информации.

- Красная шляпа: Эмоции и интуиция.

- Черная шляпа: Критическое мышление и риски.

- Желтая шляпа: Оптимизм и преимущества.

- Зеленая шляпа: Творческих идей и альтернативы.

- Синяя шляпа: Управление процессом мышления.

2.2. Метафоры и аналогии

Использование метафор и аналогий для создания новых связей и идей. Например, сравнение проблемы с известным объектом или явлением может помочь найти новые решения.

2.3. Метод "Случайных слов"

Этот метод включает в себя выбор случайного слова из словаря или генератора случайных слов. Участники пытаются связать это слово с проблемой, что может привести к неожиданным и инновационным идеям.

2.4. "Система ограничения"

Метод требует установления ограничений или условий, которые помогают задать рамки для генерации идей. Например, можно установить, что все идеи должны быть реализованы с использованием определенного бюджета или ресурсов.

3. Реализация идей

Генерация идей — это лишь первый шаг. Не менее важным этапом является их реализация.

3.1. Оценка идей

После генерации идей важно выбрать наиболее перспективные. Оцените идеи по критериям, таким как:

- Эффективность в решении проблемы

- Реалистичность и осуществимость

- Потенциальная стоимость и выгода

3.2. Прототипирование

Создание прототипов и тестирование идей на ранних стадиях позволяет выявить недостатки и улучшить концепции перед полной реализацией. Это может быть как физический образец продукта, так и концептуальная модель.

3.3. Обратная связь

Получение обратной связи от пользователей, клиентов или экспертов на этапе тестирования помогает корректировать и улучшать идеи.

3.4. Платформы для реализации

Использование специализированных платформ и сообществ для реализации идей может повысить шансы на успех. Такие платформы могут включать краудфандинг, акселераторы стартапов и бизнес-инкубаторы.

Генерация идей и их реализация — это важные процессы в любой творческой или научной деятельности. Методы мозгового штурма и креативного мышления позволяют сформировать новые и инновационные идеи, в то время как систематический подход к их оценке и реализации способствует успешному внедрению в практику. Эффективное использование этих методов может значительно повысить креативный потенциал команды и привести к достижениям в различных сферах.

**Примеры тем для научно-технического творчества**

1. Разработка экологически чистых источников энергии

- Исследование потенциальных источников чистой энергии, таких как солнечные панели, ветряные турбины и биомасса.

2. Интернет вещей (IoT) в умных городах

- Применение IoT-технологий для улучшения городской инфраструктуры: умные светофоры, системы управления трафиком и умные парковки.

3. Искусственный интеллект и машинное обучение

- Применение AI в медицинской диагностике: как глубокое обучение может помочь в выявлении заболеваний на ранних стадиях.

4. Нанотехнологии в медицине

- Исследование применения наночастиц для целевой доставки лекарственных средств и лечения рака.

5. Биометрические технологии для безопасности

- Разработка и применение систем распознавания лиц, отпечатков пальцев и радужной оболочки глаза в различных сферах.

6. Кибербезопасность в эпоху цифровизации

- Методы защиты данных и предотвращения кибератак в онлайн-среде.

7. Возобновляемые источники воды

- Технологии опреснения воды и их эффективность в разных климатических условиях.

8. Применение виртуальной и дополненной реальности в образовании

- Разработка образовательных платформ с использованием VR и AR для улучшения процесса обучения.

9. Робототехника и автоматизация

- Создание роботов для использования в сельском хозяйстве: преимущества и целесообразность.

10. Устойчивое сельское хозяйство

- Исследование методов вертикального сельского хозяйства и гидропоники для повышения продуктивности.

11. Космические технологии и исследования

- Разработка технологий для колонизации Марса: проблемы и решения.

12. Экотехнологии в строительстве

- Использование экологически чистых материалов и технологий для создания устойчивых зданий.

13. Зеленые технологии в промышленности

- Исследование способов минимизации отходов и углеродного следа в производственных процессах.

14. Разработка приложений на основе больших данных

- Как анализ больших данных может помочь в принятии решений для бизнеса или государственного управления.

15. Криптовалюты и блокчейн

- Потенциал технологии блокчейн для обеспечения прозрачности и безопасности в различных отраслях.

16. Системы умного дома

- Разработка технологий автоматизации для повышения комфорта, безопасности и энергоэффективности в жилых помещениях.

17. Технологии 3D-печати

- Применение 3D-печати в медицине: создание протезов и имплантатов на заказ.

18. Мобильные технологии в современном обществе

- Влияние мобильных приложений на повседневную жизнь и социальное взаимодействие.

19. Транспорт будущего

- Исследование технологий автономного вождения и их влияние на городскую среду.

20. Психология и технологии

- Как различные технологические инновации влияют на психическое здоровье и социальные взаимодействия.

Структура научно-технического проекта важна для систематизации работы и ясного представления результатов. Она помогает организовать информацию и обеспечить последовательность изложения. Вот типичная структура научно-технического проекта:

Структура научно-технического проекта

1. Титульный лист

- Название проекта

- ФИО авторов и их учебное заведение/организация

- Дата и место выполнения проекта

2. Аннотация

- Краткий обзор проекта, включая основную цель, методы и результаты.

- Обычно объём — 150-300 слов.

3. Содержание

- Перечень разделов и подразделов с указанием страниц.

4. Введение

- Формулировка проблемы, актуальность проекта.

- Цели и задачи исследования.

- Обзор существующих решений или технологий, связанных с темой.

5. Обзор литературы

- Обзор современных исследований и разработок в данной области.

- Выявление недостатков или пробелов в существующих работах.

- Описание методов и подходов, используемых другими исследователями.

6. Методы исследования

- Описание используемых методов, технологий и инструментов.

- Пояснение выбора методов и их обоснование.

- Принципы работы, экспериментальные установки и протоколы.

7. Результаты

- Предоставление полученных данных, результатов экспериментов и исследований.

- Таблицы, графики и схемы для иллюстрации данных.

- Анализ и интерпретация результатов.

8. Обсуждение

- Обсуждение значимости полученных результатов.

- Сравнение с результатами других исследований.

- Оценка применимости и практической значимости полученных данных.

- Обсуждение возможных ограничений исследования.

9. Заключение

- Подведение итогов проекта.

- Обобщение основных выводов.

- Рекомендации для дальнейших исследований и возможные направления для применения.

10. Список литературы

- Перечень всех источников, использованных в процессе исследования.

- Оформление по стандартам (например, APA, MLA).

11. Приложения

- Дополнительные материалы, которые могут быть полезны для понимания проекта, но не входят в основной текст (например, детализированные данные, дополнительная информация, программный код, методические указания).

Структурирование научно-технического проекта обеспечивает четкость и логичность изложения, помогает авторам и читателям лучше понять цели, методы и результаты исследования. Следуя этому шаблону, вы сможете создать качественный и профессиональный проект, который будет понятен и полезен вашей аудитории.

Требования к оформлению научно-технического проекта

1. Общие требования

- Проект должен быть оформлен на стандартном листе формата A4 (210 x 297 мм).

- Все страницы должны быть пронумерованы (за исключением титульного листа).

- Используйте полуторный интервал для разделов и списка литературы.

- Шрифт: Times New Roman или Arial. Размер шрифта — 14 pt в зависимости от требований.

- Параметры полей: обычно 2-2.5 см с каждой стороны (верхнее, нижнее, правое, левое).

2. Структура документа

- Титульный лист должен содержать название проекта, фамилии и инициалы авторов, название учебного заведения, год выполнения и город. Титульный лист оформляется по отдельным требованиям (см. приложение 2).

- Аннотация помещается на отдельной странице и обычно содержит 150-300 слов, излагающих цели, методы и результаты проекта.

- Содержание должно быть размещено на отдельной странице и включать все разделы и подразделы с соответствующими номерами страниц (см. приложение 3).

Используйте структуру как основу для написания текста.

- *Титульный лист*: должен содержать название учебного заведения, название работы, фамилию и имя автора, факультет и кафедру, курс и дату защиты (см. приложение 2).

- *Оглавление*: следует составить после завершения написания работы и включить в него названия всех разделов и подразделов с указанием страниц (см. приложение 3).

*- Введение:*

Напишите введение, в котором кратко представьте тему, обоснуйте ее актуальность и сформулируйте исследовательский вопрос или вопросы. Количество страниц – 1-2 стр. См. приложение 4.

*- Основная часть*:

Подробно изложите результаты вашего исследования, структурируя информацию по подразделам. Всего в основной части две главы. Первая глава посвящена теоретическим вопросам. Вторая глава посвящена практическим вопросам. Количество страниц – 10-15 стр

I. Первая глава: Теория

1. Введение в тему:

- Определите и охарактеризуйте ключевые понятия, актуальные для вашей темы.

- Обоснуйте актуальность темы на основе существующих исследований.

2. Обзор литературы:

- Изучите и проанализируйте существующие теории и концепции, связанные с вашей темой.

- Обозначьте основные работы, которые оказали влияние на ваше исследование.

3. Теоретические основы исследования:

- Опишите теории и модели, которые вы будете использовать для понимания и анализа вашей темы.

- Обоснуйте выбор этих теорий и их связь с вашим исследованием.

4. Формулировка гипотезы:

- Сформулируйте исследовательскую гипотезу или основные вопросы, которые вы планируете изучить.

5. Методологические аспекты:

- Объясните, каким образом теоретические знания будут применены на практике в рамках вашего исследования.

II. Вторая глава: Практика

1. Методы и материалы:

- Опишите методы, которые вы будете использовать для сбора данных (опросы, интервью, эксперименты и т.д.).

- Укажите материалы (учебные пособия, статьи, интернет-ресурсы), которые вы будете использовать в своем исследовании.

2. Проведение исследования:

- Подробно опишите процесс проведения исследования, включая условия и контекст.

3. Анализ данных:

- Представьте собранные данные в структурированном виде (таблицы, графики и диаграммы).

- Объясните методы анализа данных, которые вы использовали.

4. Обсуждение результатов:

- Интерпретируйте полученные данные и проводите сравнение с предыдущими исследованиями.

- Обсудите, как результаты вашего исследования согласуются или расходятся с существующими теориями.

5. Заключение:

- Подведите итоги вашей практической части, выделите ключевые выводы на основе анализа данных. Количество страниц – 1-2 стр. См. приложение 6.

Поддерживайте свои аргументы примерами и ссылками на источники. Ссылки на источники должны быть оформлены по ГОСТ Р 7.0.100-2018 «Библиографическая запись. Библиографическое описание. Общие требования и правила составления» (см. приложение 5).

- *Список литературы*: включает все источники, на которые вы ссылались в тексте, должен быть оформлен по установленному стандарту ГОСТ Р 7.0.100-2018 «Библиографическая запись. Библиографическое описание. Общие требования и правила составления» (см. приложение 5).

- *Приложения*: если есть дополнительные материалы (графики, таблицы, анкеты и т.д.), они могут быть размещены в приложениях.

3. Разделы и текст

- Каждый раздел начинается с нового абзаца и выделяется заголовком, который может быть написан курсивом или полужирным шрифтом.

- Заголовки разделов и подразделов должны быть четкими и содержательными. Уровни заголовков следует выделять различными стилями (например, верхний уровень — крупный шрифт, второй уровень — полужирный, третий уровень — обычный).

- Текст должен быть логично структурирован и легко читаем. Используйте абзацы, подзаголовки, списки для удобства восприятия.

4. Графические материалы

- Все таблицы, графики и изображения должны иметь свои названия и быть пронумерованы (например, «Таблица 1. Название таблицы», «Рисунок 1. Название рисунка»).

- Графические элементы должны быть четкими и читабельными, с обозначением единиц измерения.

- Ссылки на графические материалы должны быть сделаны в основном тексте (например, «См. рисунок 2» или «По данным таблицы 3»).

5. Приложения

- Приложения (если имеются) должны иметь отдельную пронумерованную страницу и включать дополнительные материалы, которые важны для понимания проекта (например, детализированные данные или методики).

- Они могут быть обозначены как «Приложение A», «Приложение B» и т.д.

6. Корректура

- Проверьте текст на наличие грамматических, орфографических и стилевых ошибок.

- Рекомендуется, чтобы проект был отредактирован и вычитан другим лицом для повышения точности и улучшения качества.

**6.2. Презентация проектов**

Презентация проекта — это важный этап в процессе представления своих идей, исследований или продуктов аудитории. Умение эффективно подготовить и провести презентацию позволяет донести информацию до слушателей, вовлечь их в обсуждение и получить положительные результаты от представления вашего проекта. В данной теме мы рассмотрим основные компоненты подготовки и проведения успешной презентации.

1. Подготовка презентации

1.1. Определение цели презентации

Прежде чем приступить к созданию презентации, важно определить её цель. Задайте себе следующие вопросы:

- Какую информацию я хочу донести?

- Каковы ожидаемые результаты презентации? (например, получение финансирования, интерес инвесторов, согласование проекта и т.д.)

- Какой тип аудитории будет присутствовать?

1.2. Исследование аудитории

Понимание вашей аудитории имеет ключевое значение для успешной презентации. Учитывайте следующие аспекты:

- Уровень знаний аудитории по вашей теме.

- Интересы и ожидания слушателей.

- Потенциальные вопросы и возражения.

1.3. Структура презентации

Структура вашей презентации должна быть логичной и последовательной. Рассмотрите следующие разделы:

- Введение: Кратко представьте саму тему и обозначьте цель презентации.

- Основная часть: Изложите основные идеи, детали исследования или предложенные решения.

- Заключение: Обобщите ключевые моменты и выдвиньте рекомендации или призыв к действию.

1.4. Визуальные материалы

Используйте визуальные материалы для поддержания и иллюстрации вашей темы:

- Слайды: Примените PowerPoint или аналогичные программы для создания презентационных слайдов. Помните о простоте и чистоте дизайна.

- Графики и таблицы: Для представления данных и статистики используйте графики и диаграммы.

- Изображения и видеоматериалы: Визуальные элементы помогают удерживать внимание аудитории.

1.5. Репетиция презентации

Проведите несколько репетиций вашей презентации. Это поможет вам:

- Ознакомиться с материалом и улучшить навык презентации.

- Определить возможные проблемы и исправить их до самой презентации.

- Убедиться, что вы укладываетесь в отведенное время.

2. Проведение презентации

2.1. Установка контакта с аудиторией

Во время презентации важно установить контакт с аудиторией:

- Начните с приветствия и краткого введения.

- Установите зрительный контакт с участниками.

- Улыбайтесь и используйте открытые жесты для создания дружелюбной атмосферы.

2.2. Четкость изложения

- Говорите ясно и уверенно.

- Избегайте использования сложных терминов, если они не являются общепринятыми для вашей аудитории.

- Подчеркивайте ключевые моменты, чтобы помочь слушателям лучше усвоить материал.

2.3. Взаимодействие с аудиторией

Стремитесь к активному взаимодействию с аудиторией:

- Задавайте вопросы для вовлечения участников.

- Пригласите мнения и комментарии.

- Обсуждайте и реагируйте на вопросы по мере их поступления.

2.4. Управление временем

Следите за временем, чтобы уложиться в отведенный лимит. Если вы чувствуете, что время подходит к концу:

- Сократите менее важные разделы.

- Перейдите к заключению.

3. Завершение презентации

3.1. Заключение

В конце презентации подведите итоги:

- Повторите ключевые моменты ваших основных идей.

- Выразите благодарность за внимание и участие.

3.2. Вопросы и обсуждение

Убедитесь, что вы оставили время для вопросов:

- Поощряйте участников задавать вопросы и обменивайтесь мнениями.

- Подготовьте ответы на возможные вопросы заранее, чтобы выглядеть уверенно.

3.3. Обратная связь

- Поинтересуйтесь у аудитории о их мнении после презентации, чтобы понять, что было особенно эффективно, а что можно улучшить в будущем.

***Требования к оформлению презентации***

1. Структура презентации

- Титульный слайд: должен включать название презентации, фамилию и имя автора(ов), название учебного заведения или организации, дату.

- Содержание: (необязательно) краткий список разделов или ключевых тем, которые будут рассмотрены в презентации.

- Слайды с основным содержанием: организуйте информацию по логическим блокам (введение, основная часть, выводы).

- Заключительный слайд: должен содержать основные выводы, возможные рекомендации и благодарности.

2. Дизайн и оформление слайдов

- Шаблон: используйте единый шаблон оформления для всей презентации, чтобы обеспечить согласованность.

- Цветовая палитра: выбирайте контрастные цвета для фона и текста, чтобы улучшить читаемость. Избегайте слишком ярких и отвлекающих цветов.

- Шрифт: используйте один или два шрифта для всего текста. Рекомендуется использовать простой шрифт без засечек (например, Arial или Helvetica) и не менее 24 pt для основного текста и 36 pt для заголовков.

- Выравнивание: текст на слайде должен быть выровнен по левому краю или по центру. Избегайте выравнивания по правому краю, так как это может усложнить чтение.

3. Текст и содержание

- Краткость: используйте краткие фразы и буллеты вместо длинных предложений. Каждая строка не должна содержать более 6-8 слов.

- Количество текста: не перегружайте слайды текстом; один слайд — одна ключевая идея или тема.

- Нумерация слайдов: нумеруйте слайды для упрощения навигации.

4. Графики и изображения

- Иллюстрации: используйте качественные и актуальные изображения, схемы и графики, чтобы визуализировать данные и поддержать идеи.

- Подписи: все изображения и графики должны иметь подписи и быть описаны в контексте текста.

- Размер: убедитесь, что графики и изображения достаточно крупные, чтобы быть видимыми на экране.

5. Анимации и переходы

- Анимации: используйте анимацию с умеренной частотой и только для выделения ключевых моментов. Избегайте чрезмерных анимаций, которые могут отвлекать внимание.

- Переходы между слайдами: применяйте простые и ненавязчивые эффекты перехода слайдов.

6. Подготовка к выступлению

- Практика: заранее отрепетируйте свое выступление, чтобы уверенно представить материал и уложиться во время.

- Обратная связь: по возможности протестируйте презентацию на другой аудитории, чтобы получить мнение о содержании и оформлении.

7. Техники взаимодействия

- Взаимодействие с аудиторией: вовлекайте слушателей, задавая вопросы или предлагая обсудить темы на слайдах.

- Коммуникация: не читайте текст слайдов; вместо этого объясняйте материал своим словами и используйте слайды как вспомогательные средства.

Успешная презентация проектов требует тщательной подготовки и уверенного исполнения. Понимание своей аудитории, структурирование информации и умение взаимодействовать с ней — ключевые факторы, которые помогут донести идею или проект до слушателей. Последовательно следуя представленным рекомендациям, вы сможете создать профессиональную и привлекательную презентацию, которая произведет положительное впечатление на вашу аудиторию.

\

**ЗАКЛЮЧЕНИЕ**

В завершение работы с данной рабочей тетрадью по научно-техническому творчеству в области информатики, мы можем уверенно констатировать, что предложенные материалы и задания успешно способствуют развитию у студентов критического мышления, креативности и практических навыков, необходимых для реализации научно-технических проектов.

Студенты освоили основные этапы творческого процесса, включая выбор темы, формулирование исследовательских вопросов, анализ существующих решений и разработку собственных проектов. Использование таблиц, схем и практических упражнений дало возможность наглядно представить изучаемые концепции и уложить их в контекст реальных задач, с которыми сталкивается современная IT-индустрия.

Одной из ключевых задач данной тетради было стимулирование студентов к самостоятельной работе и исследовательскому подходу. В результате выполнения поставленных заданий, студенты получили возможность развить навыки работы с информацией, научились оформлять свои идеи в научные отчеты и усовершенствовали свои презентационные навыки, что существенно повысит их конкурентоспособность на рынке труда в будущем.

Мы уверены, что данный материал способен стать надежным помощником не только в учебном процессе, но и в дальнейшем профессиональном развитии студентов. Он открыт для адаптации и дополнения, что позволяет использовать тетрадь не только как готовый продукт, но и как основу для создания уникальных проектов, соответствующих интересам и потребностям каждого студента.

Таким образом, данная рабочая тетрадь представляет собой важный шаг в подготовке будущих специалистов в области информатики, обеспечивая их необходимыми знаниями и навыками для успешной реализации научно-технических идей в своей профессиональной деятельности. Надеемся, что проделанная работа вдохновит студентов на новые достижения и успешные проекты в будущем.

**Рефлексия**

1. Общий анализ опыта:

- Опишите, что нового вы узнали в процессе работы над исследованием. Как это изменило ваше понимание темы?

- Как вы оцениваете уровень своих исследований? Что вам удалось сделать особенно хорошо?

- Какие навыки, по вашему мнению, вы развили в ходе работы (например, исследовательские, аналитические, письменные)?

2. Методы и инструменты:

- Какие методы исследования вы использовали? Почему вы выбрали именно их?

- Были ли какие-либо методы, которые вы бы хотели изучить более глубоко? Почему?

- Оцените, насколько эффективно вы использовали инструменты и ресурсы (например, книги, статьи, интернет-ресурсы) во время исследования.

3. Проблемы и трудности:

- Какие трудности возникли в процессе работы над исследованием? Как вы их преодолели?

- Была ли какая-либо информация, которую вам было сложно найти? Как вы справились с этим?

- Есть ли что-то, что вы бы сделали иначе, если бы у вас была возможность вернуться в прошлое?

4. Результаты исследования:

- Каковы основные выводы вашего исследования? Оправдали ли они ваши ожидания?

- Чем вы гордитесь в своем исследовании? Какой аспект работы вы считаете наиболее успешным?

- В чем вы видите возможные ограничения вашего исследования? Как их можно было бы устранить?

5. Практическое значение:

- Как результаты вашего исследования могут быть применены на практике? Есть ли у вас идеи по дальнейшему использованию ваших находок?

- Какие рекомендации вы могли бы дать другим, кто будет изучать аналогичную тему?

6. Профессиональное развитие:

- Как итог исследования повлиял на ваши карьерные цели или образовательные планы?

- Какие области знаний вы бы хотели углубить в дальнейшем? Почему это важно для вас?

- Имеются ли у вас идеи относительно будущих исследовательских проектов? Каковы ваши планы на следующий шаг в вашем образовательном или профессиональном пути?

7. Оценка работы с руководителем:

- Как вы оцениваете взаимодействие с вашим научным руководителем или консультантом?

- Что вы узнали от них, и как их советы повлияли на вашу работу?

- Есть ли что-то, что вы бы хотели изменить в процессе сотрудничества?

8. Обратная связь:

- Как вы относитесь к отзывам, полученным во время защиты? Как они соответствуют вашему собственному восприятию вашей работы?

- Какую критику вы готовы принять, а какую вам было сложно воспринимать? Почему?

- Что вы считаем наиболее полезным в полученных вами отзывах? Как они помогут вам улучшить вашу будущую работу?

*Форматы для выполнения рефлексии*

- Письменный отчет: Напишите рефлексивный отчет, охватывающий все вышеперечисленные задания.

- Дневник исследователя: Ведите дневник в течение всего периода работы над исследованием, фиксируя свои мысли и чувства.

- Групповая дискуссия: Участвуйте в дискуссии с вашими коллегами, чтобы обсудить свои ощущения и получить обратную связь.

- Визуальная рефлексия: Создайте ментальную карту или диаграмму, чтобы наглядно представить свои идеи и взаимосвязи между различными аспектами вашей работы.

**ПРИЛОЖЕНИЕ 1**

**Ответы на решение задач по пункту 1.2**

1. Алгоритмы сортировки

Задание 1: Сортировка массива (Quick Sort)

def quick\_sort(arr):

if len(arr) <= 1:

return arr

else:

pivot = arr[len(arr) // 2]

left = [x for x in arr if x < pivot]

middle = [x for x in arr if x == pivot]

right = [x for x in arr if x > pivot]

return quick\_sort(left) + middle + quick\_sort(right)

# Пример использования

data = [10, 7, 8, 9, 1, 5]

sorted\_data = quick\_sort(data)

print("Отсортированный массив:", sorted\_data) # [1, 5, 7, 8, 9, 10]

#### Задание 2: Сравнение сортировок

import time

import random

def bubble\_sort(arr):

n = len(arr)

for i in range(n):

for j in range(0, n - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

return arr

def merge\_sort(arr):

if len(arr) > 1:

mid = len(arr) // 2

left\_half = arr[:mid]

right\_half = arr[mid:]

merge\_sort(left\_half)

merge\_sort(right\_half)

i = j = k = 0

while i < len(left\_half) and j < len(right\_half):

if left\_half[i] < right\_half[j]:

arr[k] = left\_half[i]

i += 1

else:

arr[k] = right\_half[j]

j += 1

k += 1

while i < len(left\_half):

arr[k] = left\_half[i]

i += 1

k += 1

while j < len(right\_half):

arr[k] = right\_half[j]

j += 1

k += 1

return arr

# Генерация случайного массива

random\_data = random.sample(range(1000), 100)

# Сортировка пузырьком

start\_time = time.time()

bubble\_sorted = bubble\_sort(random\_data.copy())

bubble\_sort\_time = time.time() - start\_time

# Сортировка слиянием

start\_time = time.time()

merge\_sorted = merge\_sort(random\_data.copy())

merge\_sort\_time = time.time() - start\_time

print(f"Время сортировки пузырьком: {bubble\_sort\_time:.5f} секунд")

print(f"Время сортировки слиянием: {merge\_sort\_time:.5f} секунд")

Задание 3: Сортировка по нескольким критериям

class Student:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def \_\_repr\_\_(self):

return f"{self.name} ({self.age})"

def multiple\_criteria\_sort(students):

return sorted(students, key=lambda x: (x.name, x.age))

# Пример использования

students = [Student("Anna", 18), Student("John", 20), Student("Anna", 22), Student("Mike", 18)]

sorted\_students = multiple\_criteria\_sort(students)

print("Отсортированные студенты:", sorted\_students)

2. Стек

Задание 1: Проверка сбалансированных скобок

def is\_balanced(s):

stack = []

bracket\_map = {')': '(', '}': '{', ']': '['}

for char in s:

if char in bracket\_map.values():

stack.append(char)

elif char in bracket\_map.keys():

if stack == [] or bracket\_map[char] != stack.pop():

return False

return stack == []

# Пример использования

expression = "{[()]}[]"

print("Сбалансировано:", is\_balanced(expression)) # True

expression = "{[(])}"

print("Сбалансировано:", is\_balanced(expression)) # False

Задание 2: Реализация Undo

class TextEditor:

def \_\_init\_\_(self):

self.text = ""

self.history = []

def type(self, new\_text):

self.history.append(self.text) # Сохраняем текущее состояние

self.text += new\_text

def undo(self):

if self.history:

self.text = self.history.pop() # Возвращаемся к предыдущему состоянию

# Пример использования

editor = TextEditor()

editor.type("Hello")

editor.type(", world!")

print("Текущий текст:", editor.text) # "Hello, world!"

editor.undo()

print("После отмены:", editor.text) # "Hello"

3. Очередь

Задание 1: Реализация очереди

class Queue:

def \_\_init\_\_(self):

self.queue = []

def add(self, item):

self.queue.append(item)

def remove(self):

if self.queue:

return self.queue.pop(0) # FIFO

else:

return "Очередь пуста"

def peek(self):

if self.queue:

return self.queue[0]

else:

return "Очередь пуста"

# Пример использования

q = Queue()

q.add("Задача 1")

q.add("Задача 2")

print("Обработка задачи:", q.remove()) # "Задача 1"

print("Следующая задача:", q.peek()) # "Задача 2"

#### Задание 2: Моделирование обработки клиентов

class CafeQueue:

def \_\_init\_\_(self):

self.queue = []

def customer\_arrives(self, customer\_name):

self.queue.append(customer\_name)

def serve\_customer(self):

if self.queue:

return self.queue.pop(0) # Обслуживание первого клиента в очереди

else:

return "Очередь пуста"

# Пример использования

cafe\_queue = CafeQueue()

cafe\_queue.customer\_arrives("Клиент 1")

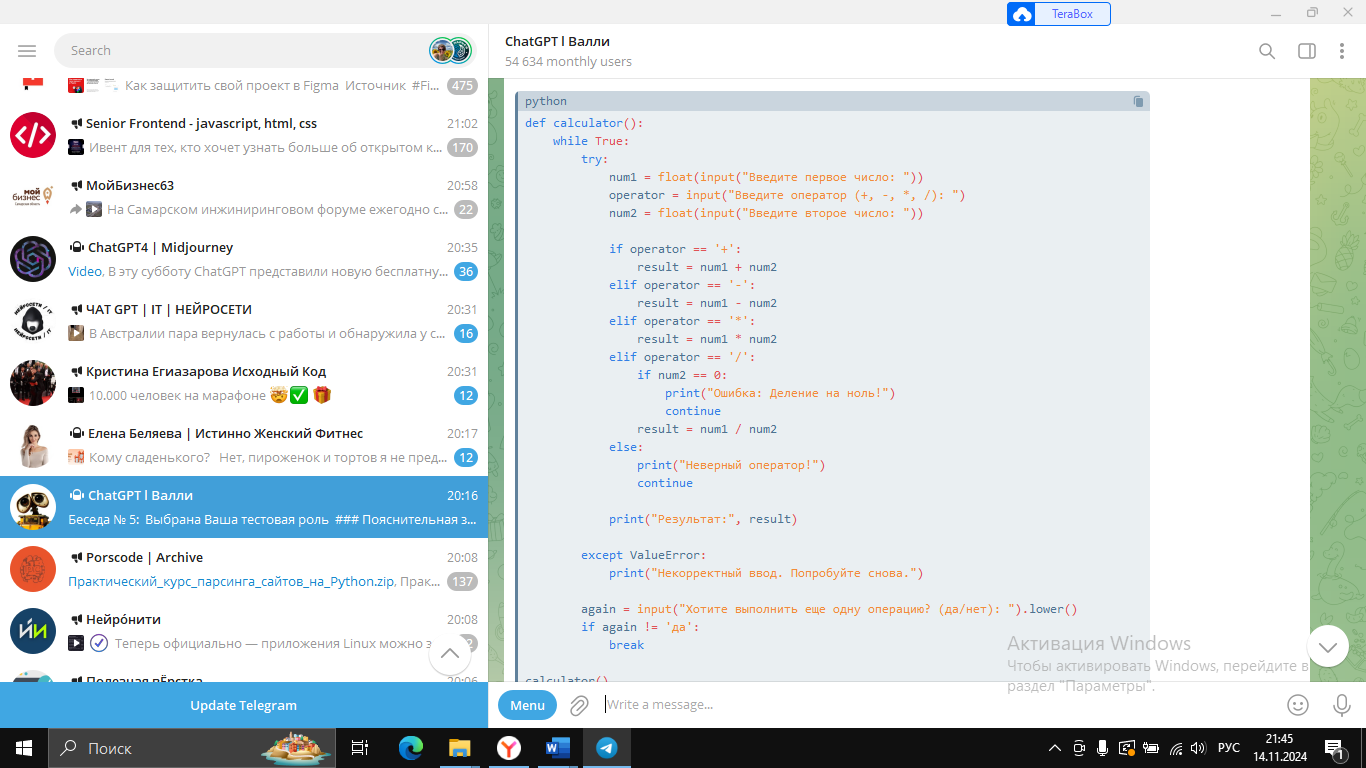
cafe\_queue.customer\_arrives("Клиент 2")

print("Обслуживаем клиента:", cafe\_queue.serve\_customer()) # "Клиент 1"

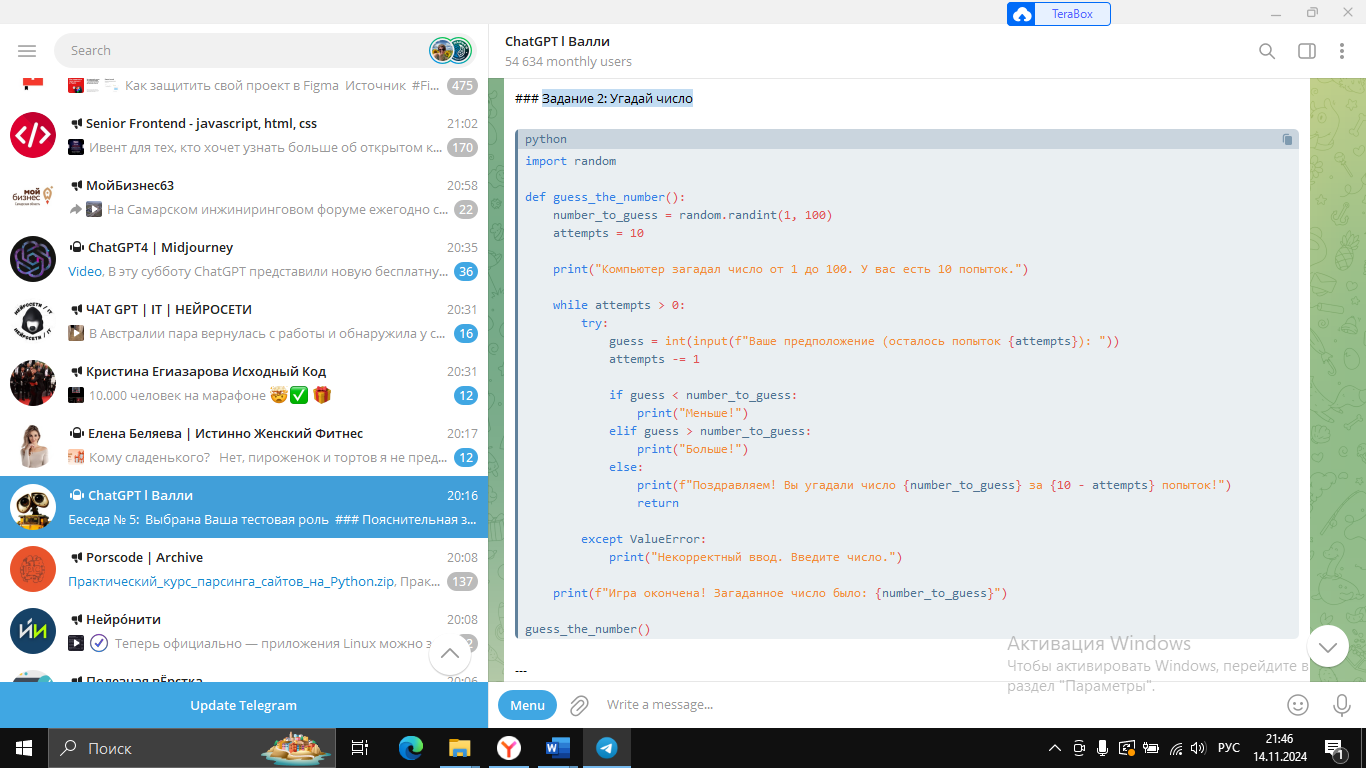
print("Обслуживаем клиента:", cafe\_queue.serve\_customer()) # "Клиент 2"

**Ответы на решение задач по пункту 1.3**

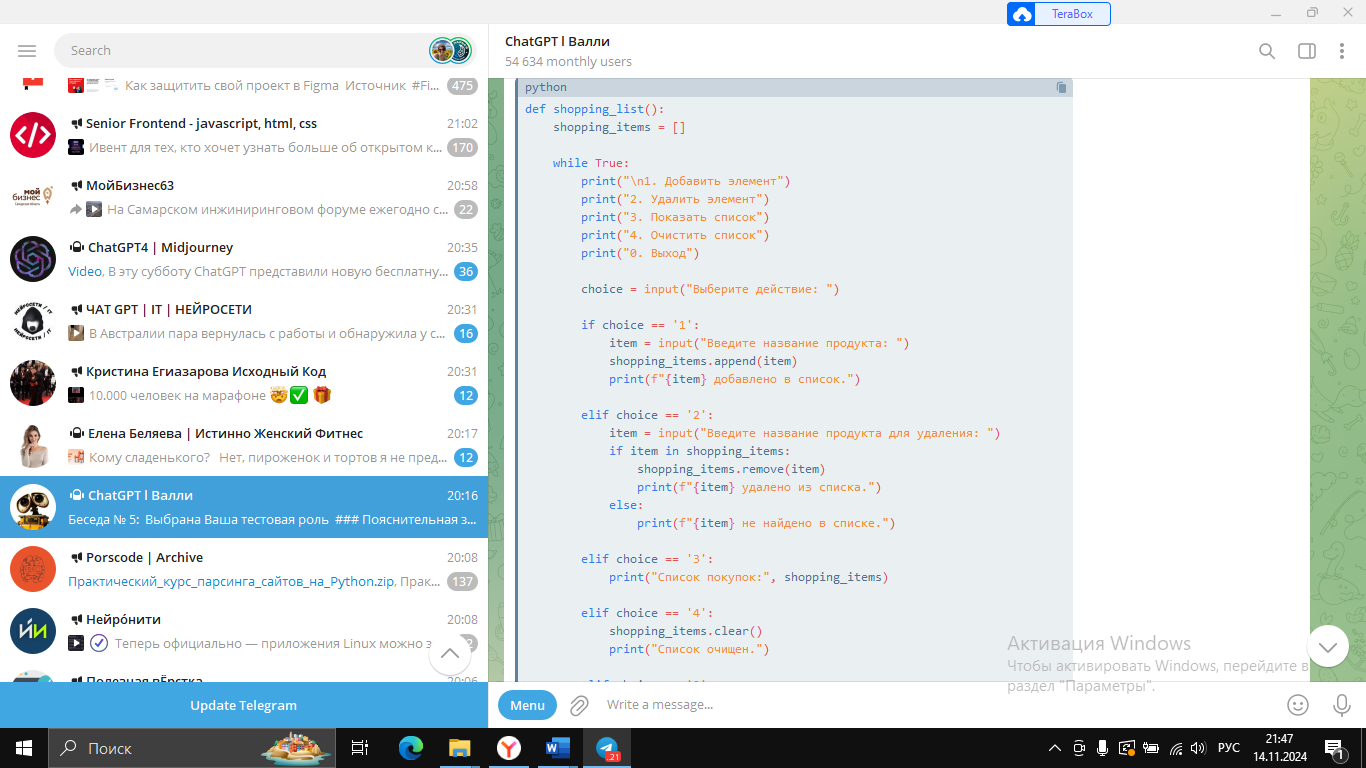
Задание 1: Калькулятор

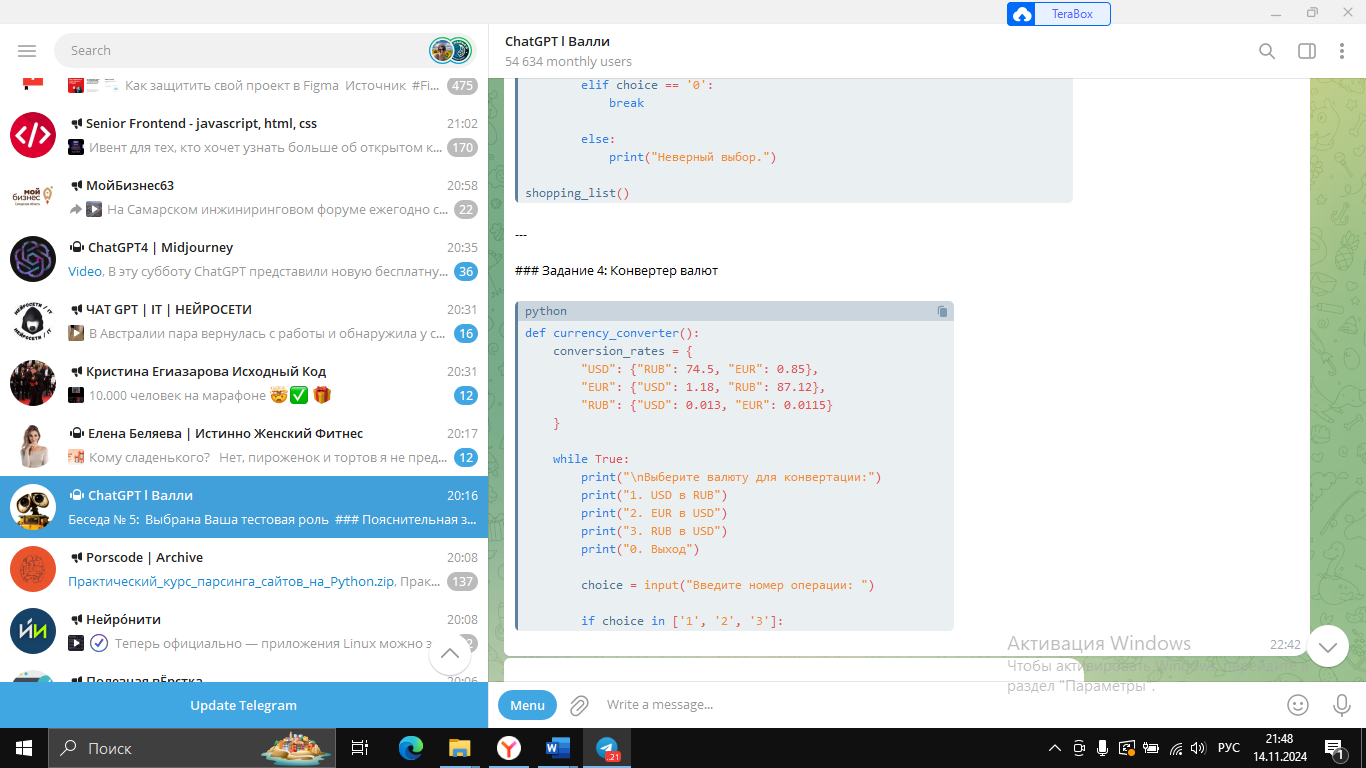


Задание 2: Угадай число

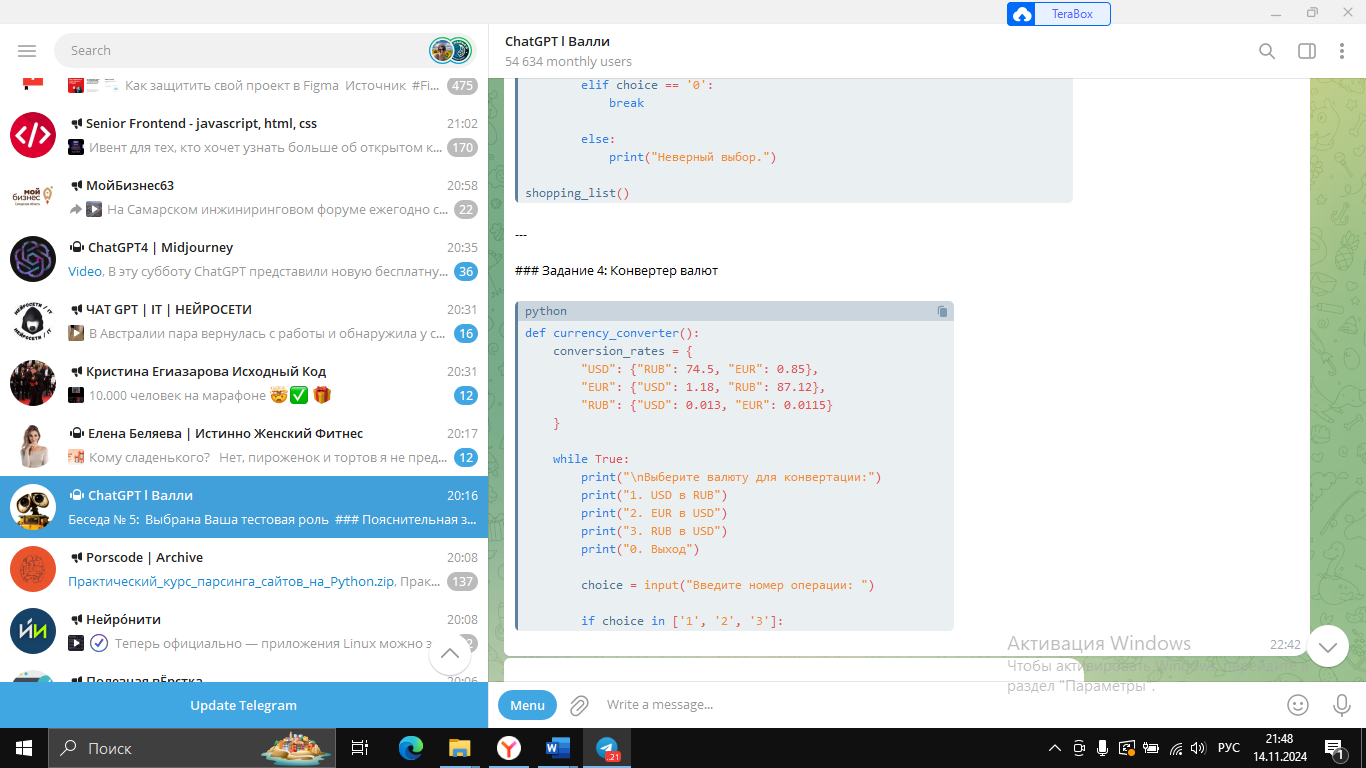


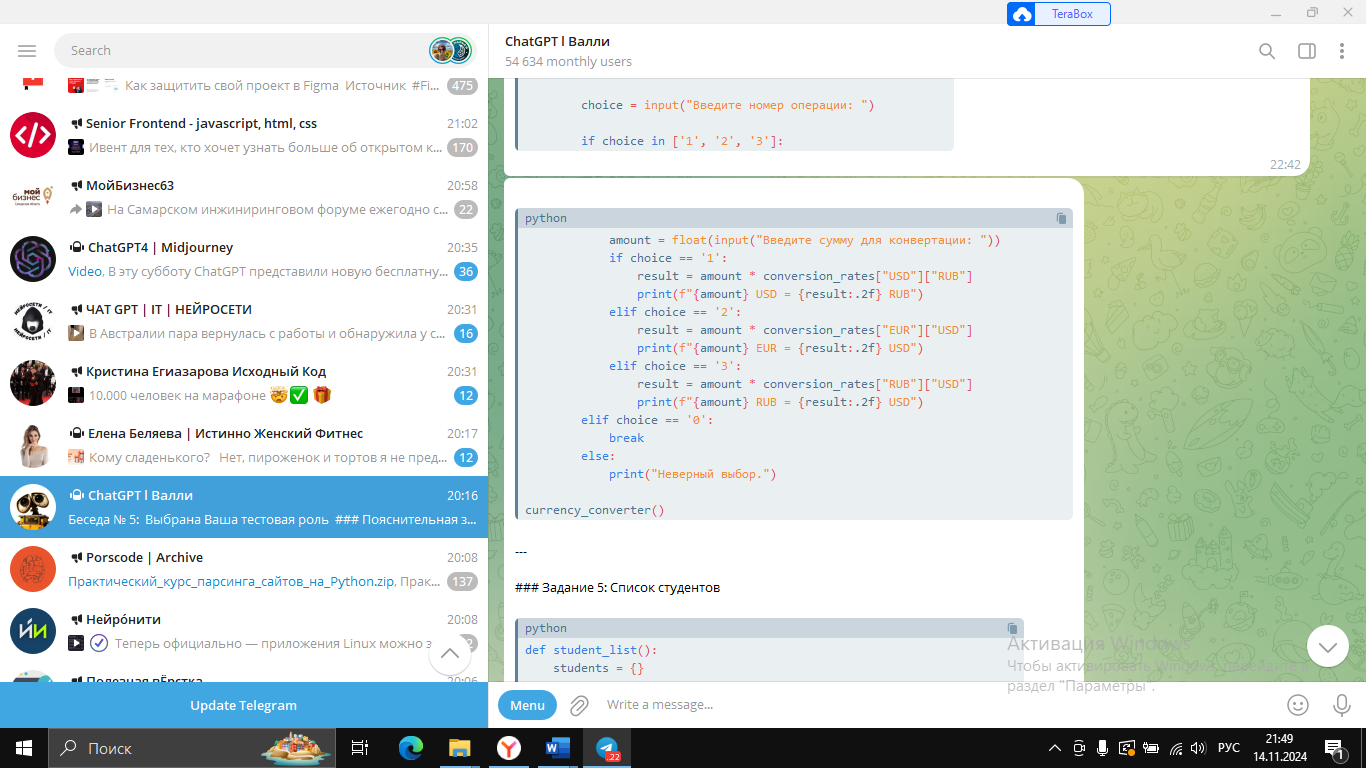
Задание 3: Список покупок



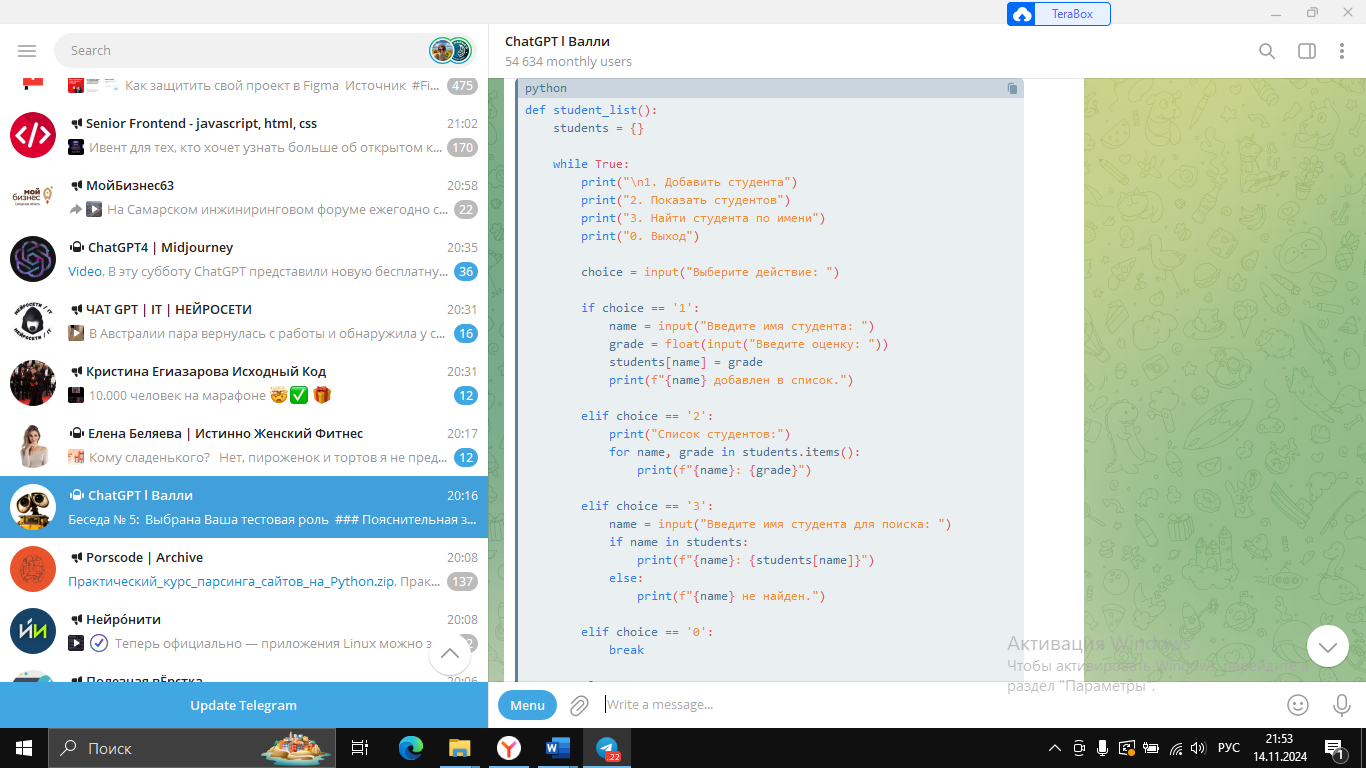


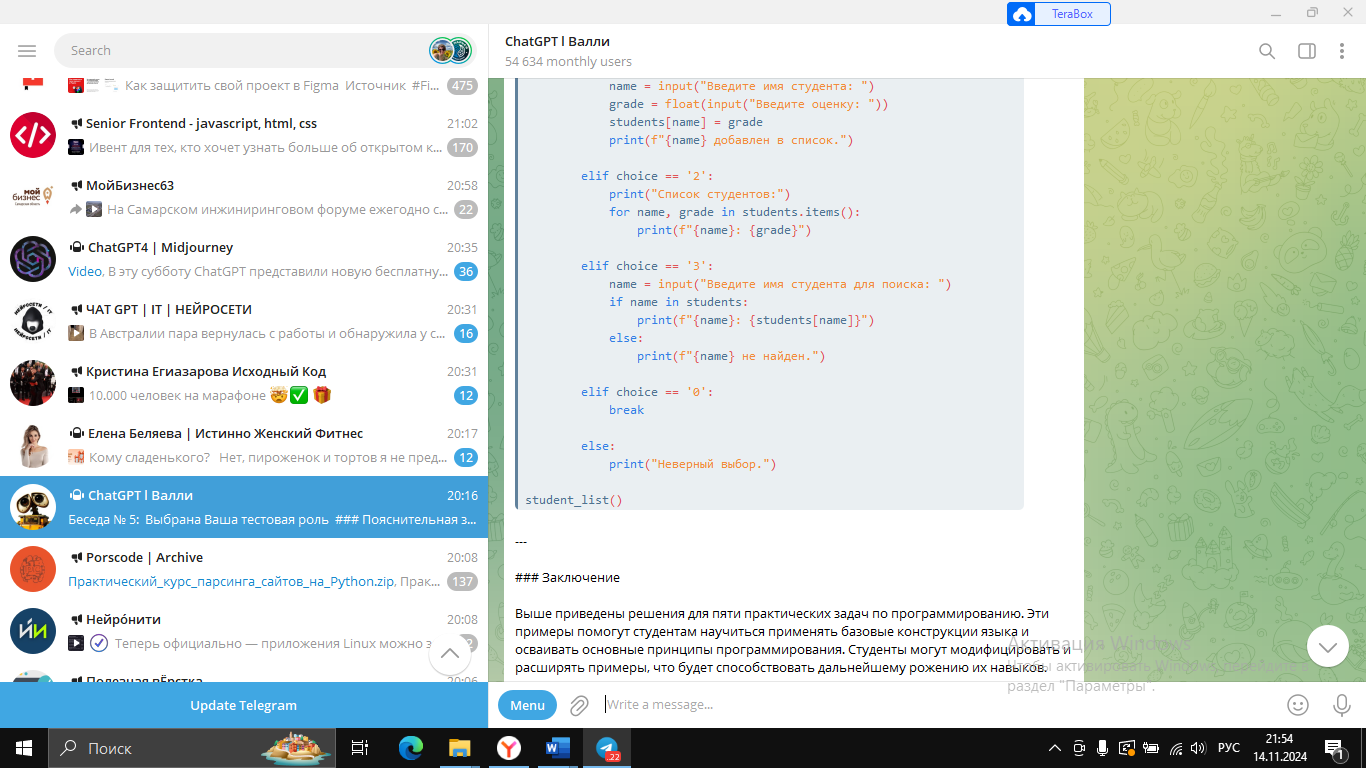
Задание 4: Конвертер валют





Задание 5: Список студентов





**Практические задания и ответы к пункту 2.2.**

1. Создание нового репозитория:

- Ответ:

1. Откройте терминал (или командную строку).

2. Перейдите в папку, где хотите создать репозиторий:

bash cd /путь/к/вашей/папке

3. Выполните команду для инициализации нового репозитория:

bash git init

4. Создайте новый текстовый файл:

bash echo "Описание моего проекта" > README.txt

5. Добавьте файл в индекс:

bash git add README.txt

6. Сделайте первый коммит:

bash git commit -m "Первоначальный коммит: добавлено описание проекта"

2. Работа с ветками:

- Ответ:

1. Создайте новую ветку:

bash git branch feature/login

2. Переключитесь на новую ветку:

bash git checkout feature/login

3. Добавьте новый файл:

bash echo "Код для логина" > login.py

4. Добавьте файл в индекс и сделайте коммит:

bash git add login.py git commit -m "Добавлен файл для логина"

5. Вернитесь на основную ветку:

bash git checkout main

6. Объедините изменения из ветки feature/login:

bash git merge feature/login

3. Работа с удаленным репозиторием:

- Ответ:

1. Создайте новый репозиторий на GitHub или GitLab (например, "my-project").

2. Склонируйте его на локальный компьютер:

bash git clone https://github.com/ваш\_логин/my-project.git

3. Перейдите в созданную папку:

bash cd my-project

4. Добавьте файл (например, README.md):

bash echo "# Мой проект" > README.md git add README.md git commit -m "Добавлен файл README"

5. Отправьте изменения в удаленный репозиторий:

bash git push origin main

4. Решение конфликтов:

- Ответ:

1. Создайте ветку для функциональности 1:

bash git checkout -b feature/feature1

2. Внесите изменения в shared\_file.txt и сделайте коммит.

3. Переключитесь на ветку main:

bash git checkout main

4. Создайте другую ветку для функциональности 2:

bash git checkout -b feature/feature2

5. Внесите изменения в shared\_file.txt в этой ветке и сделайте коммит.

6. Переключитесь обратно на ветку main и объедините первую функциональность:

bash git merge feature/feature1

7. После успешного слияния, объедините вторую ветку:

bash git merge feature/feature2

8. Если возник конфликт, Git сообщит об этом:

CONFLICT (content): Merge conflict in shared\_file.txt

9. Откройте shared\_file.txt и разрешите конфликт, выбрав нужные изменения.

10. После решения конфликта добавьте изменения и выполните коммит:

bash git add shared\_file.txt git commit -m "Разрешены конфликты при слиянии"

5. Обзор истории:

- Ответ:

1. Используйте команду для просмотра истории коммитов:

bash git log

2. Найдите последний коммит. Запомните его хеш (например, abc123).

3. Вернитесь к последнему коммиту (изменение состояния на последний коммит):

```bash

git checkout abc123

4. Чтобы вернуться обратно на основную ветку: bash

git checkout main

```

6. Работа с идеями и задачами:

- Ответ:

1. Создайте новую ветку для новой функциональности:

bash git checkout -b feature/new-feature

2. Реализуйте новую функциональность и сделайте коммит:

bash echo "Новая функциональность" > new\_feature.py git add new\_feature.py git commit -m "Реализована новая функциональность"

3. Создайте другую ветку для исправления ошибки:

bash git checkout -b bugfix/fix-bug

4. Исправьте ошибку и сделайте коммит:

bash echo "Исправление ошибки" >> bug.py git add bug.py git commit -m "Исправлена ошибка"

5. Вернитесь на основную ветку:

bash git checkout main

6. Объедините обе ветки в основную ветку:

bash git merge feature/new-feature git merge bugfix/fix-bug

**ПРИЛОЖЕНИЕ 2**

Государственное автономное профессиональное образовательное учреждение Самарской области «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

ПРОЕКТ

По дисциплине: «Информатика»

На тему: «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Выполнил:

Студент (-ка) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(группа)

Специальность \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О.)

Преподаватель:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара, 202\_

**ПРИЛОЖЕНИЕ 3**

**Пример оформления Оглавления**

**ОГЛАВЛЕНИЕ**

Введение...................................................................................................................1 Глава 1. Название главы .........................................................................................2

1.1. Название параграфа ........................................................................................3

1.2. Название параграфа ........................................................................................4

Глава 2. Название главы ........................................................................................5

2.1. Название параграфа ........................................................................................6

2.2. Название параграфа .......................................................................................7

Глава 3. Название главы ........................................................................................8

3.1. Название параграфа ........................................................................................9

3.2. Название параграфа ......................................................................................10 Заключение............................................................................................................ 11

Список использованных источников и литературы............................................12

Приложение………………………………………………………………………13

**ПРИЛОЖЕНИЕ 4**

**Введение**

В современном мире проблема [основная тема исследования] становится всё более актуальной. [Краткое введение в контекст исследования. Например: «С учетом стремительных изменений в глобальном климате, растущих потребностей общества и технологического прогресса, исследование воздействия этих факторов на…»] Эта тема имеет огромное значение для [указать, для кого важна эта тема, например, для исследователей, бизнес-сообщества, государственных структур и т.д.], так как [обосновываете актуальность, например: «это влияет на устойчивое развитие, экономическую эффективность и качество жизни»].

Несмотря на значительное количество исследования в данной области, остаётся множество вопросов, требующих дальнейшего изучения. В частности, [упомяните конкретные аспекты или пробелы в знаниях, на которые вы сосредоточитесь в своей работе. Например: «недостаточно исследованы механизмы влияния... на... или последствия для...»].

Цель данной исследовательской работы состоит в [формулировка цели или целей исследования, например: «в подробном анализе... для выявления основных факторов риска»]. В процессе работы будут рассмотрены следующие ключевые вопросы:

- [Вопрос 1]

- [Вопрос 2]

- [Вопрос 3]

Структура работы предполагает [кратко изложите структуру вашей работы, например: «введение, обзор литературы, методология, основные результаты и заключение»]. В итоге, данное исследование направлено на [укажите, как ожидаете дополнить существующие знания, например: «расширение существующих теоретических пониманий, что может помочь в дальнейшем развитии методов...»].

**Примечания по написанию введения:**

1. Чёткая формулировка темы: Введение должно сразу же «зацепить» читателя, поэтому начните с общего контекста темы или актуальной проблемы.

2. Актуальность: Обоснуйте, почему выбранная тема важна, какие вопросы остаются неразрешёнными и почему на них следует обратить внимание.

3. Цели и вопросы: Чётко обозначьте цели вашего исследования и ключевые вопросы, которые будут рассмотрены в вашей работе, чтобы читатель знал, чего ожидать.

4. Структура: Укажите, как вы планируете организовать свою работу, это поможет читателю сориентироваться в содержании.

5. Стиль: Используйте научный и формальный стиль, избегайте избыточных сложностей и жаргона.

Данное введение служит примером и ориентиром. Вам следует адаптировать его под свою конкретную тему и исследование.

**ПРИЛОЖЕНИЕ 5**

**ГОСТ Р 7.0.100-2018 «Библиографическая запись. Библиографическое описание. Общие требования и правила составления»**

Примеры библиографического описания по ГОСТ Р 7.0.100-2018

 [[Подробнее...](https://yspu.org/%D0%9E%D1%84%D0%BE%D1%80%D0%BC%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%B0_%D0%BB%D0%B8%D1%82%D0%B5%D1%80%D0%B0%D1%82%D1%83%D1%80%D1%8B)]

* **СХЕМЫ ОПИСАНИЯ ИСТОЧНИКОВ В ЦЕЛОМ**

Официальные документы.

***Конституции, конвенции, договоры, соглашения, концепции, доктрины:***

Всеобщая декларация прав человека. — Москва : Права человека, 1996. — 16 с.

Конституция Российской Федерации : принята 12 декабря 1993 г. — Москва : Юрист, 2012. — 48 с.

***Постановления:***

О лицензировании деятельности по разработке и производству средств защиты конфиденциальной информации : постановление Правительства РФ от 03.03.2012 № 171 // Собрание законодательства РФ. — 2012. — № 11. — Ст. 1297.

Законодательные материалы:

Российская Федерация. Законы. Об общих принципах организации местного самоуправления в Российской Федерации : Федеральный закон № 131-ФЗ : [принят Государственной Думой 16 сент. 2003 г. : одобрен Советом Федерации 24 сент. 2003 г.]. — Москва : Проспект ; Санкт-Петербург : Кодекс, 2017. — 158 с.

О библиотечном деле : Федеральный закон № 78-ФЗ от 29 дек. 1994 г. : принят Государственной Думой 23 нояб. 1994 г. // Собрание законодательства Российской Федерации. — 1995. — № 1. — Ст. 2.

Книги с одним автором:

*В примерах этого раздела приведены разные варианты описания издательств (один город и два издательства, несколько городов со своими издательствами, отсутствие сведений об издательстве).*

Василевская Н. С. Философия и методология научной деятельности : учеб. пособие / Н. С. Василевская ; ЯГПУ им. К. Д. Ушинского. — Ярославль : РИО ЯГПУ, 2017. — 266 с.

*один город и два издательства*

Вакуленко Л. С. Воспитание и обучение детей с нарушениями речи. Психология детей с нарушениями речи : учеб.-метод. пособие / Л. С. Вакуленко. — Москва : ФОРУМ : ИНФРА-М, 2019. — 271 с.

*При наличии серии; несколько городов со своими издательствами*

Алешина Л. С. Ленинград и окрестности : справ.-путеводитель / Л. С. Алешина. — 3-е изд., испр. и доп. — Москва : Искусство ; Лейпциг : Эдицион, 1990. — 479 с. : ил. — (Памятники искусства Советского Союза).

*Отсутствие сведений об издательстве*

Симоненко В. Е. Схемы разводки в русских народных хорах и хороводах : графическое пособие / В. Е. Симоненко. — Санкт-Петербург : [б. и.], 1998. — 11 с. : ил.

Книги с двумя авторами:

Слепко Ю. Н. Психология учебной деятельности школьника : системогенетический подход : монография / Ю. Н. Слепко, Ю. П. Поваренков ; ЯГПУ им. К. Д. Ушинского. — Ярославль : РИО ЯГПУ, 2019. — 263 с.

Kay S. Inside Out : Students book : Upper intermediate / S. Kay, V. Jones. — Oxford : Macmillan Heinemann, 2001. — 160 p.

Книги с тремя авторами:

Шадриков В. Д. Методология и методы изучения способностей и одаренности : монография / В. Д. Шадриков, В. А. Мазилов, Ю. Н. Слепко ; ЯГПУ им. К. Д. Ушинского. — Ярославль : РИО ЯГПУ, 2018. — 159 с.

Книги с четырьмя авторами:

*Описываются под заглавием. За косой чертой указывают всех авторов.*

Информационно-библиографическая культура : учеб. пособие / В. В. Брежнева, Т. В. Захарчук, А. А. Грузова, М. И. Кий ; СПбГИК. — Санкт-Петербург : СПбГИК, 2017. — 203 с.

Книги с пятью и более авторами:

*В новом ГОСТе при наличии информации о 5-ти и более авторах приводят имена первых 3-х и в квадратных скобках сокращение «[и др.]»*

Физическая культура и здоровый образ жизни : учеб. пособие / В. С. Кунарев, И. И. Башмашникова, В. Н. Бледнова [и др.] ; Учеб.-метод. об-ние по направлениям пед. образования, Рос. гос. пед. ун-т им. А. И. Герцена. — Санкт-Петербург : Изд-во Рос. гос. пед. ун-та им. А. И. Герцена, 2009. — 138 с.

Театр и мы / А. Б. Сергеев, А. Б. Иванов, А. Б. Петров [и др.]. — М. : [б.и.], 1978. — 115 с.

Методические пособия:

Оформление курсовых, дипломных, диссертационных работ : метод. рекомендации / сост. В. С. Крылова. — 2-е изд., доп. — Томск, 2002. — 37 с.

Логопедия. Нарушения голоса : учеб.-метод. пособие / сост. Е. Е. Васильева. — 2-е изд., испр. — Ярославль : Изд-во ЯГПУ, 2012. — 111 с.

Сборники:

Мир и война : очерки из истории рус. сов. драматургии 1946—1980 гг. / РАН, Гос. ин-т искусствознания; отв. ред. И. Л. Вишневская. — Москва : Ленанд, 2009. — 287 с.

Педагогическая практика студентов в системе профессиональной подготовки кадров : материалы межрегион. науч.-практ. конф. 19 февраля 2015 г. / под ред. Л. В. Байбородовой, ЯГПУ им. К. Д. Ушинского. — Ярославль : РИО ЯГПУ, 2015. — 143 с.

Естествознание. Полевая практика : метод. рек. / сост. А. В. Пизов, Т. Р. Ковригина, О. С. Бекиш. — Ярославль : Изд-во ЯГПУ, 2010. — 31 с.

Практика сказкотерапии : сборник сказок, игр и терапевт. программ / под ред. Н. А. Сакович. — Санкт-Петербург : Речь, 2007. — 217, [3] с. — (Психологический тренинг).

Work and Family : Policies for a Changing Work Force. — Washington : Nat. Acad. Press, 1991. — 260 p.

*Ответственных лиц, не авторов, разрешено сокращать до первого [и др]):*

Хрестоматия по культурологии. Т. 1. Самосознание мировой культуры / ред. И. Ф. Кефели [и др.]. — Санкт-Петербург : Петрополис : Изд-во Санкт-Петербург. ун-та МВД России, 1999. — 312 с.

Тома многотомного издания:

Пастернак Б. Л. Полное собрание сочинений с приложениями. В 11 т. Т. 7. Письма, 1905—1926 / Б. Л. Пастернак. — Москва : Слово / Slovo, 2005. — 823 с.

Хрестоматия по культурологии. Т. 1. Самосознание мировой культуры / ред. И. Ф. Кефели [и др.]. — Санкт-Петербург : Петрополис : Изд-во Санкт-Петербург. ун-та МВД России, 1999. — 312 с.

Диссертации и авторефераты:

Прозоров И. Е. Отечественная научно-вспомогательная литературная библиография (1917—1929 гг.) : тенденции развития и организационные формы : дис. … канд. пед. наук : 05.25.03 / Прозоров Иван Евгеньевич; науч. рук. О. Н. Ильина ; СПбГУКИ. — Санкт-Петербург, 2010. — 361 с.

Елинер И. Г. Развитие мультимедийной культуры в информационном обществе : автореф. дис. … д-ра культурологии : 24.00.01 / Елинер Илья Григорьевич ; СПбГУКИ. — Санкт-Петербург, 2010. — 34 с.

Ноты:

Рахманинов С. В. Три русские песни : перелож. для фп. и хора / С. В. Рахманинов. — Москва : Золотое Руно, 2007. — 24 с. : нот.

Сергеева Т. П. Инициалы : концерт для альт. домры, орк. рус. нар. инструментов и фп. / Т. П. Сергеева. — Москва : Композитор, 2008. — 64 с. : нот.

Чайковский П. И. Орлеанская дева : опера в 4 д. (6 карт.) / П. И. Чайковский; авт. текста : Ф. Шиллер, В. А. Жуковский. — Клавир. — Москва : Изд. П. Юргенсон, 1880. — 205 с. : нот.

Карты и атласы:

Атлас по истории средних веков : (с компл. контур. карт) / сост. и подгот. к изд. ПКО «Картография» в 1970 г. ; отв. ред. Е. Н. Регентова. — Испр. в 1999 г. — М. : Роскартография, 2000. — 1 атл. (20 с.)

Словари и энциклопедии:

Новейший культурологический словарь : термины, биогр. справки, иллюстрации / сост. В. Д. Лихвар, Е. А. Подольская, Д. Е. Погорелый. — Ростов-на-Дону : Феникс, 2010. — 411 с. : ил.

Новая Российская энциклопедия. В 12 т. Т. 8 (2). Когезия — Костариканцы / ред. А. Д. Некипелов. — Москва : Энциклопедия, 2011. — 480 с. : ил.

Примеры описания составной части

 [[Подробнее...](https://yspu.org/%D0%9E%D1%84%D0%BE%D1%80%D0%BC%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%B0_%D0%BB%D0%B8%D1%82%D0%B5%D1%80%D0%B0%D1%82%D1%83%D1%80%D1%8B)]

* **СХЕМЫ ОПИСАНИЯ СОСТАВНОЙ ЧАСТИ РЕСУРСА (статьи)**

*Порядок приведения авторов в статьях такой же, как в книгах.*

Статьи из книг:

Сукиасян Э. Р. От документа — к ресурсу / Э. Р. Сукиасян // Современное библиотечно-информационное образование / СПбГУКИ, Библ.-информ. фак. — Санкт-Петербург, 2008. — Вып. 9. — С. 9-11.

Минкина В. А. Участие службы информации в рекламной деятельности / В. А. Минкина, Н. В. Рудакова // Справочник информационного работника / ред. Р. С. Гиляревский, В. А. Минкина. — Санкт-Петербург, 2005. — С. 405—410.

Брежнева В. В. Профильная подготовка бакалавров библиотечно-информационной деятельности в СПбГИК / В. В. Брежнева, М. Н. Колесникова, Д. А. Эльяшевич // Труды Санкт-Петербургского государственного института культуры. — Санкт-Петербург, 2015. — Т. 205 : Непрерывное библиотечно-информационное образование. — С. 24-31.

**Статьи из периодических и продолжающихся изданий**

Журналов и газет:

Один автор:

Кулюкина И. П. Занимательная логопедия : мастер-класс для родителей детей с нарушениями речи / И. П. Кулюкина // Логопед. — 2019. — № 4. — С. 60-65.

Леонова Н. А. Младший школьник : опыт нравственного поведения / Н. А. Леонова // Начальная школа. — 2010. — № 5. — С. 22-26.

Два-три автора:

Терновская Г. И. Формирование произносительной стороны речи у дошкольников с тяжелыми нарушениями речи средствами логоритмики / Г. И. Терновская, В. Г. Смышляева // Логопед в детском саду. — 2019. — № 4. — С. 77-83.

Четыре автора, под заглавием:

Приоритетные направления научных исследований в области информатизации системы подготовки специалистов по физической культуре, спорту и туризму / А. И. Федоров, Р. В. Хоменко, Е. В. Антропова, И. В. Сафина // Теория и практика физической культуры. — 2018. — № 5. — С. 13-15.

Пять и более авторов:

*(за косой чертой указываем трех авторов и в квадратных скобках пишем [и др.]*

Выборы вице-президентов РБА : позиции кандидатов / Н. Н. Квелидзе-Кузнецова, И. Б. Михнова, Р. А. Барышев [и др.] // Университетская книга. — 2019. — № 3. — С. 20-29.

Goldina O. The Establishment of an Enterprise Information Service: The Case of the ECI Telecom Company / O. Goldina // Scientific and technical information processing. — 2009. — Vol. 36, № 2. — P. 112—115.

Статья опубликована в нескольких номерах журнала:

Воловник А. Эффект Бильбао / А. Воловник // Мир музея. — 2018. — № 8. — С. 48-49 ; № 9. — С. 44-46.

Статья из газеты:

Хохрякова С. Просто жить : итоги кинофестиваля «Сталкер» / С. Хохрякова // Культура. — 2010. — 23 дек. — С. 8.

Статьи из трудов, научных конференций:

Чернявская А. П. Модульный принцип обучения в магистратуре / А. П. Чернявская // Педагогика и психология современного образования : теория и практика : материалы конференции «Чтения Ушинского» 1 — 2 марта 2018 г. — Ярославль. — 2018. — Ч. 3. — С. 62-66.

Паскаро А. П. Особенности накопления морскими цветковыми растениями тяжелых металлов / А. П. Паскаро // Труды 13-й конференции молодых ученых биологического факультета МГУ (г. Москва, 15-17 марта 1982 г.). — Москва, 1982. — Ч. 2. — С. 12-24.

Брежнева В. В. Профильная подготовка бакалавров библиотечно-информационной деятельности в СПбГИК / В. В. Брежнева, М. Н. Колесникова, Д. А. Эльяшевич // Труды Санкт-Петербургского государственного института культуры. — Санкт-Петербург, 2015. — Т. 205 : Непрерывное библиотечно-информационное образование. — С. 24-31.

Из главы, раздела, параграфа:

Первиков Ю. В. Иммунные комплексы при вирусных инфекциях / Ю. В. Первиков, Л. Б. Эльберт. — Москва, 1984. — Гл. 4 : Активация иммунными комплексами системы комплемента. — С. 78-85.

Статьи с сайтов:

Янина О. Н. Особенности функционирования и развития рынка акций в России и за рубежом / О. Н. Янина, А. А. Федосеева // Социальные науки: social-economic sciences. — 2018. — № 1. — URL: <http://academymanag.ru/journal/Yanina_Fedoseeva_2.pdf> (дата обращения: 04.06.2018).

**Электронные ресурсы:**

*Перед электронным адресом приводится аббревиатура URL. После электронного адреса обязательно указывать дату обращения к ресурсу.*

Сайты в сети интернет:

Российская государственная библиотека : официальный сайт. — Москва, 1999. — URL : [http://www.rsl.ru](http://www.rsl.ru/) (дата обращения 18.02.2020).

Правительство Российской Федерации : официальный сайт. — Москва. — URL : [http://government.ru](http://government.ru/) (дата обращения: 19.02.2020).

Непейвода С. И. Грим : учеб. пособие / С. И. Непейвода. — 3-е, стер. — Санкт-Петербург : Лань : Планета музыки, 2019. — URL: <https://e.lanbook.com/book/112770> (дата обращения 24.05.2019).

Статьи с сайтов:

Янина О. Н. Особенности функционирования и развития рынка акций в России и за рубежом / О. Н. Янина, А. А. Федосеева // Социальные науки: social-economic sciences. — 2018. — № 1. — URL: <http://academymanag.ru/journal/Yanina_Fedoseeva_2.pdf> (дата обращения: 04.06.2018).

Книги из ЭБС:

Лалаева Р. И. Логопедия в таблицах и схемах : учеб. пособие / Р. И. Лалаева, Л. Г. Парамонова, С. Н. Шаховская. — Москва : ПАРАДИГМА, 2012. — 216 c. — URL : <http://www.iprbookshop.ru/13024.html> (дата обращения: 07.02.2020).

Прищепова И. В. Логопедия : дизорфография у детей : учеб. пособие / И. В. Прищепова. — Москва : Юрайт, 2019. — 201 с. — URL: <https://biblio-online.ru/bcode/444464> (дата обращения: 07.02.2020).

Ахутина, Т. В. Диагностика речевых нарушений школьников : практическое пособие / Т. В. Ахутина, Т. А. Фотекова. — 3-е изд., испр. и доп. — Москва : Юрайт, 2019. — 157 с. — URL: <https://biblio-online.ru/bcode/438213> (дата обращения: 07.02.2020)

Диски:

Менеджмент качества и деятельность библиотек / Ком. по культуре Санкт-Петербурга, Центр. гор. публ. б-ка им. В. В. Маяковского. — Санкт-Петербург : Центр. гор. универс. б-ка им. В. В. Маяковского, 2009. — 1 электрон. опт. диск (CD-ROM).

**Примеры библиографических ссылок**

*затекстовая, вынесенная за текст документа или его части*

…поддержка творческой активности и инициативы, инновационной деятельности молодежи, ее социальная защита, поощрение студенческого самоуправления и самоуправления [1, с. 11—12].

Компетентностный подход, акцентирующий внимание на результативности образования, заключается не в определенной сумме приобретенных школьником или студентом знаний или количестве усвоенной информации, а в способности «человека действовать в различных проблемных ситуациях» [4, с. 6].

…изучение нейрохимических процессов обучения и памяти [2, с. 3].

**ПРИЛОЖЕНИЕ 6**

**ЗАКЛЮЧЕНИЕ**

В ходе проведенного исследования мы рассмотрели ключевые аспекты, связанные с [тема исследования] и оценили их влияние на [указать область применения, например, образовательный процесс, развитие языковых навыков и т.д.]. Основной целью работы было [указать цель], и нам удалось достичь ее, проанализировав теоретические и практические аспекты вопроса.

В первой главе мы исследовали существующие теории и концепции, связанные с [темой]. Обзор литературы позволил нам понять, какие исследования уже проведены в данной области и какие пробелы остаются, что обосновывает актуальность нашего исследования. Мы выделили основные факторы, влияющие на [указать предмет исследования], и описали теоретические основы, служащие фундаментом для практической части.

Во второй главе мы подробно описали методологию, использованную для сбора и анализа данных, а также провели исследование среди [указать группу исследуемых, например, студентов, преподавателей и т.д.]. Полученные результаты, представленные в виде графиков и таблиц, подтвердили [или опровергли] нашу гипотезу о [сформулировать гипотезу]. Анализ данных показал, что [выделить основные результаты], что соответствует [или не соответствует] ожиданиям, основанным на теоретической части.

На основании проведенного исследования можно сделать несколько выводов. Во-первых, [первый вывод]. Во-вторых, [второй вывод]. В-третьих, результаты нашего исследования подчеркивают необходимость [обсудить рекомендации или предложения для дальнейших исследований, практического применения и т.д.].

Таким образом, указанное исследование не только углубляет понимание [темы], но и открывает новые горизонты для будущих направлений изучения. Мы надеемся, что результаты нашей работы окажутся полезными для [указать целевую аудиторию, например, преподавателей, студентов и т.д.], а также послужат основой для дальнейших исследований в данной области.

**ПРИЛОЖЕНИЕ 7**

**Рекомендации по дополнительным курсам**

1. Онлайн-курсы

- Stepik: Платформа с курсами по программированию, анализу данных, машинному обучению и многому другому. Есть курсы от российских вузов и компаний.

- GeekBrains: Онлайн-образование в области IT с курсами по программированию, дизайну, аналитике и другим направлениям.

- Skillbox: Здесь можно найти курсы по различным направлениям в IT, включая веб-разработку, маркетинг, аналитику и т.д.

- Фоксфорд: Платформа для онлайн-обучения, предлагающая курсы по информатике и программированию для школьников и студентов.

2. Специализированные образовательные платформы

- Курсеры: Доступ к русскоязычным курсам по программированию и IT от разных учебных заведений и компаний.

Рекомендации по литературе

1. Книги по программированию и алгоритмам

- "Программирование на Python" – Станислав Меньшиков

- "Алгоритмы: построение и анализ" – Т. Х. Кормен, Ч. Е. Лейзерсон, Р. Л. Ривест, К. Штайн (перевод доступен на русском)

- "Изучаем Python" – Эрик Маттес

2. Книги по анализу данных и машинному обучению

- "Глубокое обучение" – Иан Гудфеллоу, Йошуа Бенджио, Аарон Курвиль (перевод на русский язык)

- "Python и анализ данных" – Уэс МакКинни (перевод на русский язык)

3. Книги по архитектуре программного обеспечения и проектированию систем

- "Чистый код: Создание, анализ и развитие программного обеспечения" – Роберт Мартин (перевод на русский язык)

- "Изучаем системное проектирование" – Эрик Гамма и др.

Рекомендации по событиям

1. Конференции и семинары

- HighLoad: Конференция, посвященная масштабированию и разработке высоконагруженных систем, проходит в России.

- ConfTech: Конференция для разработчиков и технарей на разные темы в сфере технологий, информационных систем и разработок.

- Дни науки: Мероприятия, проводимые в различных университетах и научных учреждениях, где обсуждаются последние достижения в области информатики.

2. Хакатоны

- Хакатоны от крупных IT-компаний: Следите за анонсами хакатонов от компаний, таких как Яндекс, Сбер и др. Они часто объявляют о возможности участвовать в мероприятиях по программированию.

3. Митапы и сообщества

- Meetup: В России активно проводятся митапы по темам программирования, технологий и стартапов.

- Хабр: Платформа и сообщество для IT-специалистов, где можно найти много статей, обсуждений и анонсов событий.

**ПРИЛОЖЕНИЕ 8**

**Словарик информатических терминов**

API (Application Programming Interface): интерфейс, который позволяет различным программам взаимодействовать друг с другом.

Big Data: большие объемы данных, которые требуют специальных методов и инструментов для хранения, анализа и обработки.

DevOps: практика, объединяющая разработки и операционные процессы, направленная на улучшение сотрудничества между командами и автоматизацию процессов.

Агент: программа или устройство, которое выполняет конкретные задачи или принимает решения от имени пользователя или системы.

Алгоритм сортировки: метод упорядочивания элементов в массиве или списке (например, сортировка пузырьком, быстрая сортировка).

Алгоритм шифрования: метод, используемый для преобразования данных в защищённый формат, который может быть прочитан только с использованием ключа.

Алгоритм: последовательность инструкций (операций), которая описывает, как решить задачу или выполнить программу.

Анализ данных: процесс систематического применения статистических и логических методов для описания, прогнозирования и принятия решений на основе данных.

База данных (БД): организованная структура для хранения, управления и извлечения информации.

Блокчейн: децентрализованная технология хранения данных, обеспечивающая безопасность и прозрачность транзакций.

Бэкенд: часть веб-приложения, которая работает на сервере, обрабатывает запросы и управляет базами данных, обычно не видима пользователю.

Веб-разработка: процесс создания и поддержки веб-сайтов и веб-приложений.

Версионирование: система контроля версий, позволяющая отслеживать изменения в коде и управлять ими.

Виртуализация: создание виртуальных (не физических) версий ресурсов, таких как серверы, операционные системы и носители данных.

Виртуальная реальность (VR): компьютерно-сгенерированная среда, погружающая пользователя в иммерсивное взаимодействие с 3D-моделями.

Данные: факты, цифры или информация, которые могут быть обработаны, сохранены и извлечены.

Домен: уникальное имя, используемое для идентификации сайта в Интернете (например, www.example.com).

Идентификация и аутентификация: процессы определения и подтверждения личности пользователя в информационной системе.

Интернет вещей (IoT): концепция, при которой физические устройства связаны с Интернетом и могут обмениваться данными.

Интерфейс пользователя (UI): то, с чем взаимодействует пользователь в приложении, включая кнопки, меню, тексты и другие элементы.

Искусственный интеллект (ИИ): область информатики, занимающаяся созданием систем, способных выполнять задачи, требующие человеческого интеллекта, такие как распознавание речи, игра в шахматы и т.д.

Классы и объекты: классы — шаблоны для создания объектов, объекты — экземпляры классов, которые содержат свойства и методы.

Клиент-серверная архитектура: модель, в которой клиентские приложения запрашивают ресурсы или услуги у серверов через сеть.

Контейнеризация: технология, позволяющая упаковывать приложения и зависимости в единый контейнер для упрощения развертывания и управления.

Криптография: наука о методах защиты информации, включая шифрование и дешифрование данных.

Кэширование: процесс временного хранения данных, чтобы ускорить доступ к ним и повысить производительность системы.

Машинное обучение: подмножество ИИ, которое позволяет компьютерам учиться и улучшаться на основе данных без явного программирования.

Методология разработки: структурированный подход к разработке программного обеспечения, например, Agile, Waterfall, Scrum.

Микросервисы: архитектурный стиль, при котором приложение делится на небольшие, независимые сервисы, действующие взаимосвязанно.

Модуль: независимая часть программы, которая выполняет определённую функцию и может быть повторно использована в других приложениях.

Облачные вычисления: использование удалённых серверов и услуг через интернет для хранения данных и выполнения вычислений.

Объектно-ориентированное программирование (ООП): парадигма программирования, основанная на концепции "объектов", которые содержат данные и функции.

Операционная система (ОС): программное обеспечение, управляющее аппаратным обеспечением компьютера и предоставляющее платформу для выполнения приложений (например, Windows, Linux, macOS).

Пользовательский опыт (UX): общее впечатление пользователя от взаимодействия с продуктом, включая его удобство, доступность и радость.

Программирование: процесс написания, тестирования и поддержания кода для создания программного обеспечения.

Реляционная база данных: система хранения данных, основанная на таблицах с взаимосвязями между ними (например, MySQL, PostgreSQL).

Сетевая безопасность: практика и технологии, направленные на защиту компьютерной сети от киберугроз и несанкционированного доступа.

Сетевой протокол: набор правил и стандартов, которые определяют, как устройства в сети обмениваются данными.

Сетевые технологии: технологии, обеспечивающие передачу данных между компьютерами и устройствами в сети (например, TCP/IP, Ethernet).

Система контроля версий: инструмент, позволяющий отслеживать изменения в коде и управлять версиями файлов (например, Git, Subversion).

Система управления контентом (CMS): программное обеспечение, предназначенное для создания и управления цифровым контентом (например, WordPress, Joomla).

Скриптовый язык: язык программирования, который используется для написания выполнения небольших программ и автоматизации задач (например, JavaScript, Python).

СУБД (Система Управления Базами Данных): программное обеспечение для создания, управления и взаимодействия с базами данных.

Тестирование: процесс анализа и оценки программного обеспечения на наличие ошибок и соответствие требованиям.

Управление проектами: процесс планирования, выполнения и контроля проекта для достижения заданных целей и соблюдения сроков.

Файл: единица хранения данных в компьютере, которая может содержать текст, изображения, видео или другую информацию.

Файловая система: метод и структура, которые операционная система использует для управления файлами на носителе данных.

Фреймворк: набор инструментов и библиотек, который упрощает разработку приложений, предоставляя структурированные способы решения общих задач.

Фронтенд: часть веб-приложения, с которой взаимодействует пользователь, включает в себя интерфейс и визуальные элементы.

Шаблон проектирования: повторяющееся решение для распространённой проблемы в проектировании программного обеспечения (например, Singleton, Factory, Observer).