Операционные системы

Отчёт по 1 этапу проекта

Когенгар Ришард

7 марта 2025

Российский университет дружбы народов, Москва, Россия

<u>Цели и задачи</u>

Цель лабораторной работы

Добавить к сайту данные о себе.

Выполнение лабораторной работы

Файл об авторе

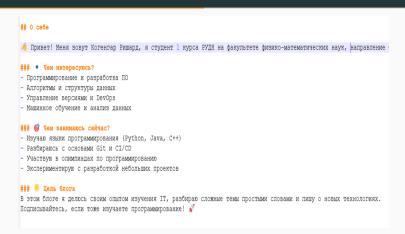


Рис. 1: Файл об авторе

Файл для поста

Итоги недели

Эта неделя была насыщенной и продуктивной. Вот несколько ключевых моментов:

翰 Учёба

- Разобрался с основами дискретной математики множества, логика, булевы функции.
- В программировании углубился в работу с указателями в С++. Интересная, но местами коварная тема.

📏 Практика

- Освоил работу с GitHub, научился делать pull request и разбирать конфликты.
- Написал небольшой скрипт на Python для автоматизации скучных задач.

🔲 Саморазвитие

- Прочитал несколько глав из "Чистой архитектуры" Роберта Мартина.
- Начал изучать основы Docker, понял, насколько это мощный инструмент.

4 Вывод

Неделя прошла полезно, но всегда есть куда расти. В планах — больше практики и погружение в алгоритмы. Как прошла ваша неделя? Делитесь в комментариях!

√

Рис. 2: Файл для поста

Файл для публикации

```
## Управление версиями: зачем это нужно и как работает Git?
### • Что такое управление версиями?
Управление версиями (VCS - Version Control System) - это система, которая отслеживает изменения в коле и позволяет управлять г
Вез VCS разработчики сталкиваются с проблемами:
- Потеря важных изменений
- Работа над разными версиями кода в хаотичном порядке
- Трупности при совместной разработке
Git - это самая популярная система управления версиями, которая повроляет:
- Хранить историю изменений
- Откатываться к предыдущим версиям
- Работать в команде, создавая параллельные ветки
- Автоматизировать процессы с помощью CI/CD
### . Kar paforaer Git?
Git использует **распределённую архитектуру**. В отличие от централизованных VCs, где все изменения хранятся на одном сервере
✓ Можно работать без постоянного подключения к серверу
✓ Все изменения фиксируются локально, а потом отправляются в удалённый репозиторий
✓ Меньше риска потерять данные
### • Основные концепции Git.
- **Peповиторий** - хранилище проекта с историей изменений
- **Kommut (commit) ** - сохранение изменений в репозитории
- **Berka (branch) ** - парадлельная линия разработки
- **Слияние (merge) ** - объединение изменений из разных веток
- **KOHOREKT (conflict) ** - CUTVALIUS, KOPEA USMEHENUS B DASHNX BETKAX EDOTUBODEVAT EDVE UDVEV
- **Удалённый реповиторий** - облачное хранилише (например, GitHub, GitLab, Bitbucket)
```

Рис. 3: Файл для публикации



Результаты выполнения лабораторной работы

Добавили к сайту данные о себе.