

**Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Институт перспективной инженерии  
Департамент цифровых, робототехнических систем и электроники**

**ОТЧЕТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1  
дисциплины «Программирование на Python»**

Выполнила:  
Федорова Дарья Юрьевна  
2 курс, группа ИВТ-б-о-24-1,  
09.03.01 «Информатика и  
вычислительная техника»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем», очная  
форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р. А.

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2025 г.

Тема: Исследование основных возможностей Get и GitHub

Цель: исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.

Ход работы:

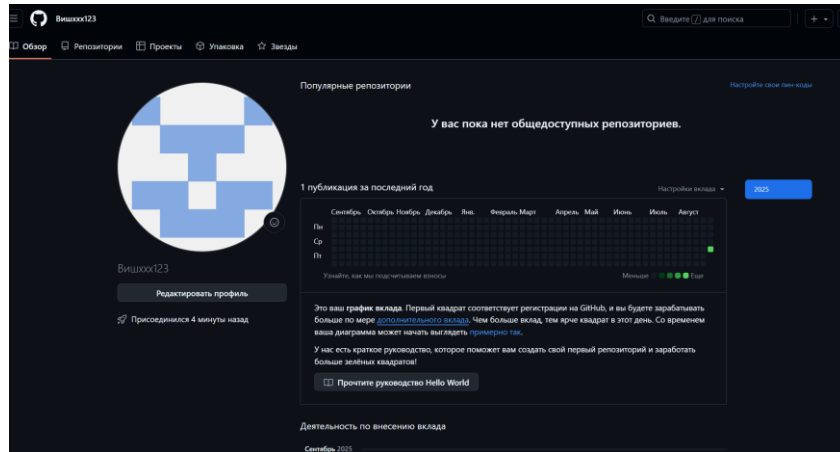


Рис. 1 – создание персонального аккаунта на сайте GitHub.

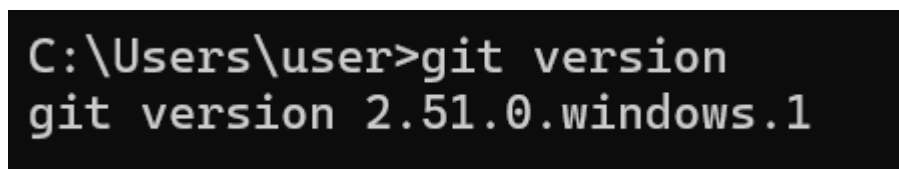


Рис. 2 – установка git, команда для проверки версии.

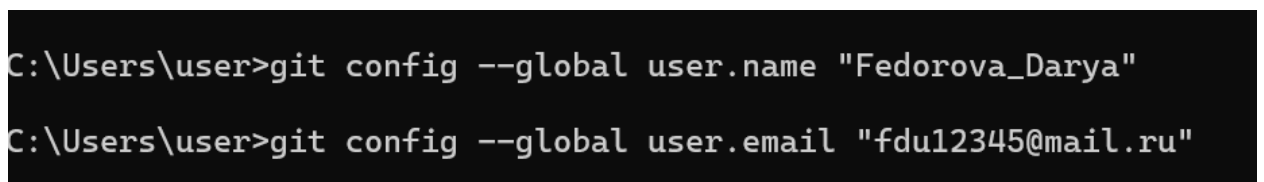


Рис. 3 – добавление в настройки имени, почты.

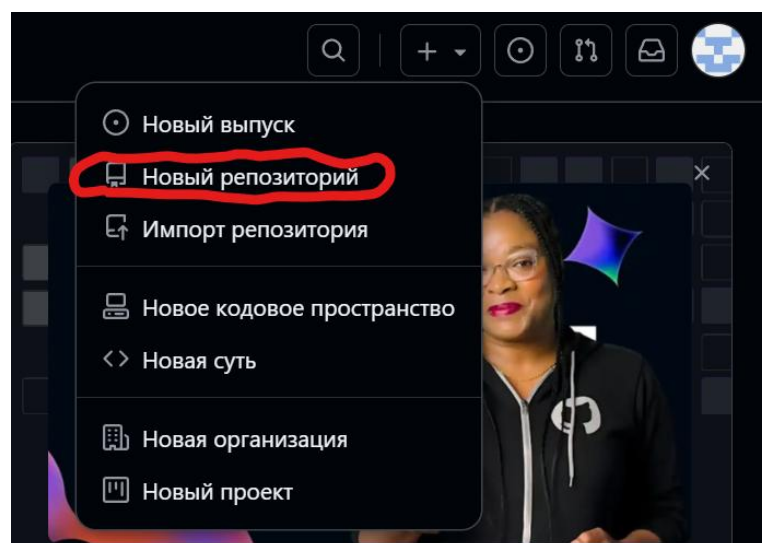


Рис. 4 – переход на страницу создания репозитория.

### Создайте новый репозиторий

Репозитории содержат файлы проекта и историю версий. У вас есть проект в другом месте? [Импортируйте репозиторий](#).

Обязательные поля отмечены звёздочкой (\*).

- Общая информация**

Владелец \* Vishhh123 / Название репозитория \* Repozitori

✓ Repoziṡoriy доступен.

Хорошие названия репозитория должны быть короткими и запоминающимися. Как насчёт **вымышленного-эврика?**

Описание

0 / 350 символов
- Конфигурация**

Выберите видимость \* Публичный

Выберите, кто сможет просматривать этот репозиторий и вносить в него изменения

Добавить README ВЫКЛ

README можно использовать для более подробных описаний. [О README](#)

Добавить .gitignore Питон

Файл .gitignore сообщает Git, какие файлы не нужно отслеживать. [Об игнорировании файлов](#)

Добавить лицензию MIT License

Лицензии объясняют, как другие могут использовать ваш код. [О лицензиях](#)

**Создать репозиторий**

Рис. 5 – создание, настройка репозитория.

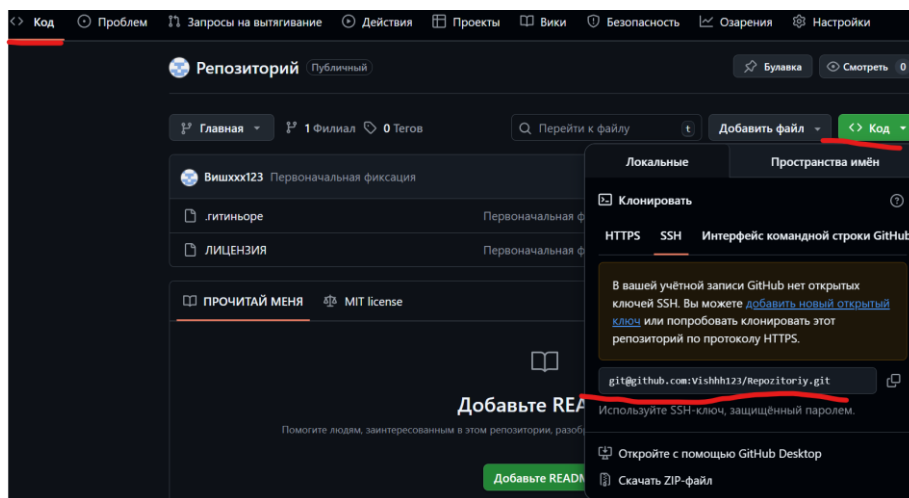


Рис. 6 – копирование адреса репозитория.

```
C:\Users\user>git clone https://github.com/Vishhh123/Repozitori.git
Cloning into 'Repozitori'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (4/4), done.
```

Рис. 7 – копирование репозитория на устройство.

```
C:\Users\user>cd Repozitoriy
```

Рис. 8 – переход к файлу проекта.

```
C:\Users\user\Repozitoriy>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
C:\Users\user\Repozitoriy>
```

Рис. 9 - проверка состояния репозитория

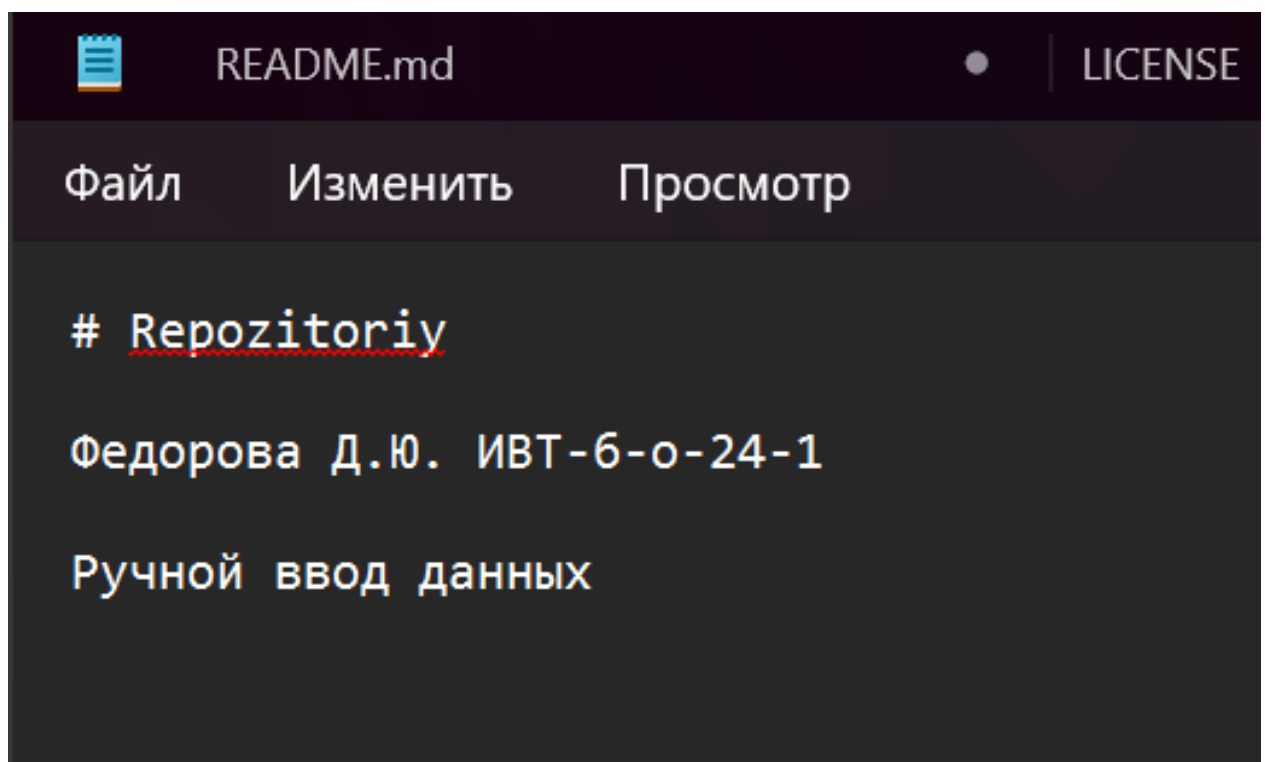


Рис. 10 – переход в файл README.md (его изменение) – ручное изменение.

```
C:\Users\user\Repozitoriy>notepad README.md

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Ввод с консоли"
[main 8710d2e] Ввод с консоли
 1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\user\Repozitoriy>
```

Рис. 11 – добавление коммита.  
(локальное сохранение).

```
C:\Users\user\Repozitoriy>git commit -m "Ввод с консоли"
[main 9387edc] Ввод с консоли
 1 file changed, 3 insertions(+), 1 deletion(-)

C:\Users\user\Repozitoriy>type README.md
# Repozytoriy

Федорова Д.Ю. ИВТ-6-о-24-1
```

Рис. 12 – фиксирование изменений в файлах на определенный момент времени.

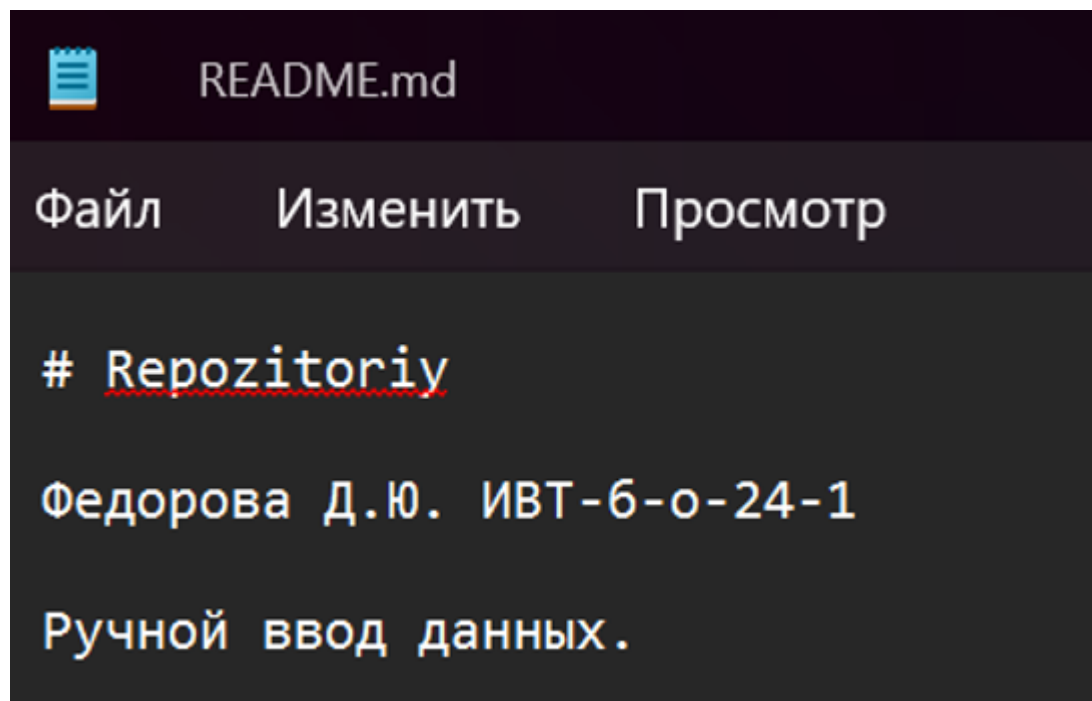
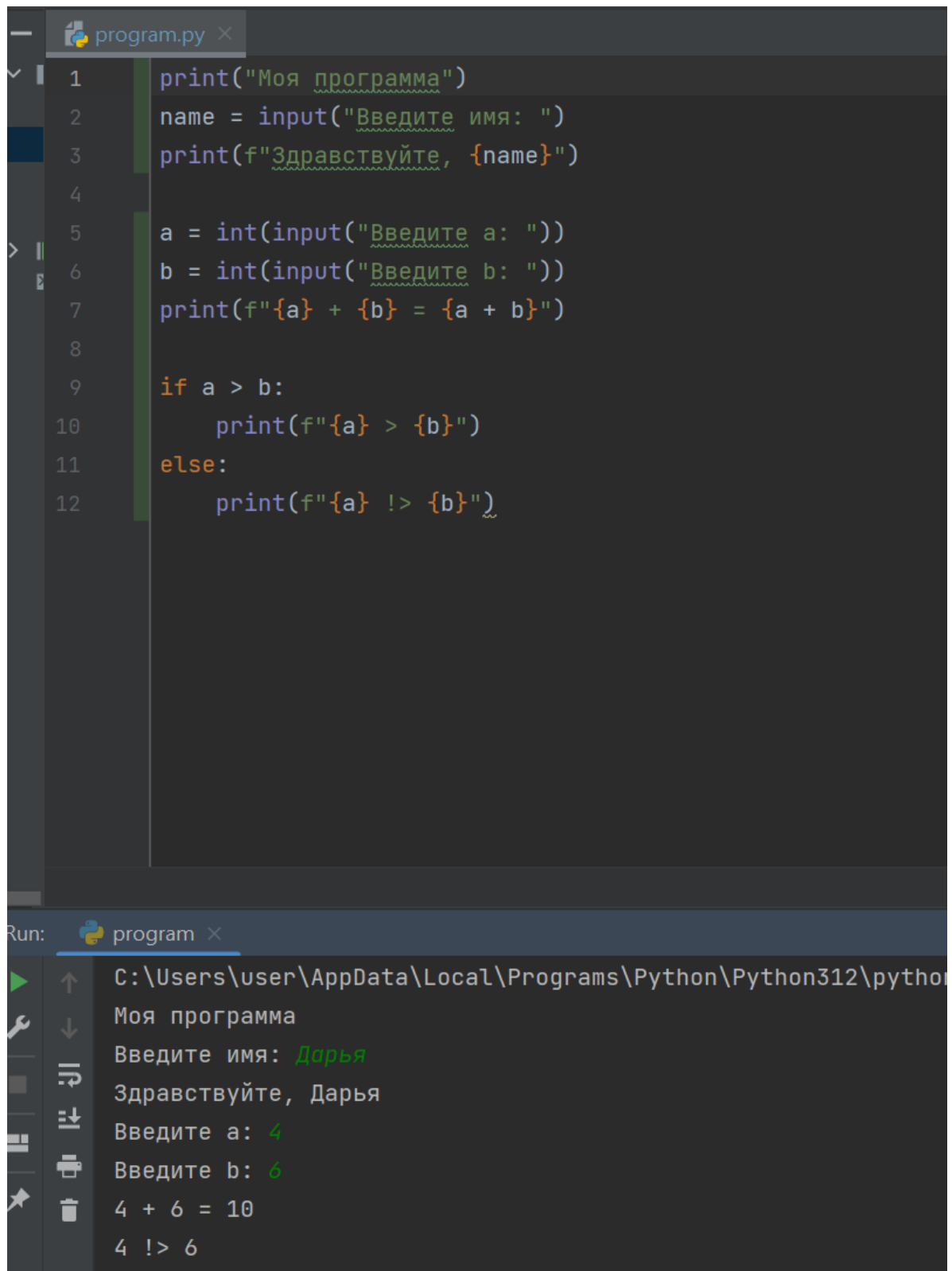


Рис. 13 – вид файла.



The image shows a Python IDE with a dark theme. The top pane displays a Python script named 'program.py' with 12 lines of code. The code includes a print statement, an input statement for a name, a formatted string print statement, two input statements for integers 'a' and 'b', a formatted string print statement for their sum, and an if-else conditional statement comparing 'a' and 'b'. The bottom pane shows the execution output, which matches the program's logic: it prints 'Моя программа', prompts for a name (input: 'Дарья'), prints a greeting, prompts for 'a' (input: '4') and 'b' (input: '6'), prints the sum '4 + 6 = 10', and prints the comparison '4 !> 6'.

```
1 print("Моя программа")
2 name = input("Введите имя: ")
3 print(f"Здравствуйте, {name}")
4
5 a = int(input("Введите a: "))
6 b = int(input("Введите b: "))
7 print(f"{a} + {b} = {a + b}")
8
9 if a > b:
10     print(f"{a} > {b}")
11 else:
12     print(f"{a} !> {b}")
```

Run: program ×

С:\Users\user\AppData\Local\Programs\Python\Python312\python  
Моя программа  
Введите имя: Дарья  
Здравствуйте, Дарья  
Введите a: 4  
Введите b: 6  
4 + 6 = 10  
4 !> 6

Рис. 14 – создание простой программы.

```

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Создание основы программы с выводом приветствия"
[main 8815568] Создание основы программы с выводом приветствия
1 file changed, 12 insertions(+), 1 deletion(-)

C:\Users\user\Repozitoriy>notepad README.md

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Добавил ввод имени пользователя"
[main ebb7913] Добавил ввод имени пользователя
1 file changed, 9 deletions(-)

C:\Users\user\Repozitoriy>notepad README.md

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Добавление ввода имени"
[main d9be639] Добавление ввода имени
1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\user\Repozitoriy>notepad README.md

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Добавление ввода числа а"
[main 00c425e] Добавление ввода числа а
1 file changed, 1 insertion(+)

C:\Users\user\Repozitoriy>>notepad README.md

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Добавление ввода числа b"
[main 7f3f617] Добавление ввода числа b
1 file changed, 1 insertion(+)

C:\Users\user\Repozitoriy>notepad README.md

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Добавление операции сложения"
[main ed8f46c] Добавление операции сложения
1 file changed, 1 insertion(+)

C:\Users\user\Repozitoriy>notepad README.md

C:\Users\user\Repozitoriy>git add README.md

C:\Users\user\Repozitoriy>git commit -m "Добавление сравнения чисел"
[main b61b1f4] Добавление сравнения чисел
1 file changed, 4 insertions(+), 1 deletion(-)

```

Рис. 15 – построчное добавление коммитов параллельно с кодом.

```

C:\Users\user\Repozitoriuy>git log --oneline README.md
b61b1f4 (HEAD -> main) Добавление сравнения чисел
ed8f46c Добавление операции сложения
7f3f617 Добавление ввода числа b
00c425e Добавление ввода числа a
d9be639 Добавление ввода имени
ebb7913 Добавил ввод имени пользователя
8815568 Создание основы программы с выводом приветствия
c481951 Создание основы программы

```

Рис. 16 - коммиты к коду.

README.md

LICENSE

Файл

Изменить

Просмотр

```

# Repozitoriuy

Федорова Д.Ю. ИВТ-6-о-24-1

Ручной ввод данных.

print("Моя программа")
name = input("Введите имя: ")
print(f"Здравствуй, {name}")
a = int(input("Введите a: "))
b = int(input("Введите b: "))
print(f"{a} + {b} = {a + b}")
if a > b:
    print(f"{a} > {b}")
else:
    print(f"{a} !> {b}")

```

Рис. 17 - код программы в блокноте.



```

C:\Users\user\Repozitoriy>git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .idea/
        git
        notepad

nothing added to commit but untracked files present (use "git add" to track)

C:\Users\user\Repozitoriy>git pull origin main
From https://github.com/Vishhh123/Repozitoriy
 * branch          main          -> FETCH_HEAD
Already up to date.

C:\Users\user\Repozitoriy>git push origin main
Everything up-to-date

C:\Users\user\Repozitoriy>git push --set-upstream origin main
branch 'main' set up to track 'origin/main'.
Everything up-to-date

C:\Users\user\Repozitoriy>git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .idea/
        git
        notepad

nothing added to commit but untracked files present (use "git add" to track)

```

Рис. 18 - локальный и удаленный репозитории полностью синхронизированы.

git pull - проверка обновления на GitHub (все актуально)  
git push - отправка свои коммиты на GitHub (все отправлено)  
git push --set-upstream origin main - настройка отслеживания ветки.

Ответы на контрольные вопросы:

1. Что такое СКВ (git) и каково ее назначение?  
Система контроля версий (СКВ) — это программа для отслеживания изменений в файлах. Позволяет сохранять историю, возвращаться к предыдущим версиям и работать в команде.

2. В чем недостатки локальных и централизованных СКВ?

*Локальные:* нет совместной работы, риск потери данных.

*Централизованные:* единый сервер — если он упадет, работа остановится.

3. К какой СКВ относится Git?  
К распределенной (каждый разработчик имеет полную копию репозитория).

4. В чем концептуальное отличие Git от других СКВ?  
Git хранит не изменения файлов, а снимки всего проекта на каждый коммит.

5. Как обеспечивается целостность хранимых данных в Git?  
Через хеш-суммы (SHA-1). Каждый коммит и файл имеют уникальный хеш.
6. В каких состояниях могут находиться файлы в Git?  
modified (изменен),  
staged (подготовлен к коммиту),  
committed (сохранен в репозитории).
7. Что такое профиль пользователя в GitHub?  
Это ваша учетная запись на [github.com](https://github.com) с репозиториями, настройками и историей действий.
8. Какие бывают репозитории в GitHub?  
Public (открытые для всех),  
Private (только для вас и collaborators).
9. Основные этапы модели работы с GitHub:  
Клонировать(`git clone`) → Изменить(modify) → Добавить (`git add`) → Закоммитить(`git commit`) → Запустить (`git push`).
10. Как осуществляется первоначальная настройка Git после установки?  
`git config --global user.name "Ваше имя"`  
`git config --global user.email "ваша@почта.com"`
11. Этапы создания репозитория в GitHub:  
Зарегистрироваться → Нажать «New repository» → Ввести имя → Выбрать лицензию → Create.
12. Какие типы лицензий поддерживаются GitHub?  
MIT, GPL, Apache — они определяют правила использования вашего кода.
13. Как осуществляется клонирование репозитория?  
`git clone https://github.com/username/repo.git`  
Нужно, чтобы скопировать проект с GitHub на компьютер.
14. Как проверить состояние локального репозитория?  
`git status`
15. Как изменяется состояние после операций?  
`git add` — файлы из modified переходят в staged.  
`git commit` — файлы из staged переходят в committed.  
`git push` — изменения отправляются на сервер.
16. Как синхронизировать два компьютера с GitHub?  
На каждом компьютере:  
`git clone https://github.com/username/repo.git`  
После изменений:  
`git pull` - получить обновления  
`git add . && git commit -m "update" && git push` отправить свои
17. Другие сервисы, кроме GitHub?  
GitLab: больше функций для CI/CD, можно hosting свой. Вэб-платформа для управления репозиториями, проектами.  
Bitbucket: бесплатные private-репозитории.
18. Программы с графическим интерфейсом для Git?  
GitKraken, Sourcetree, GitHub Desktop.

Вместо команд `add/commit` — выделяете файлы и жмете кнопки «Stage» и «Commit».

Вывод: в ходе работы были исследованы базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.