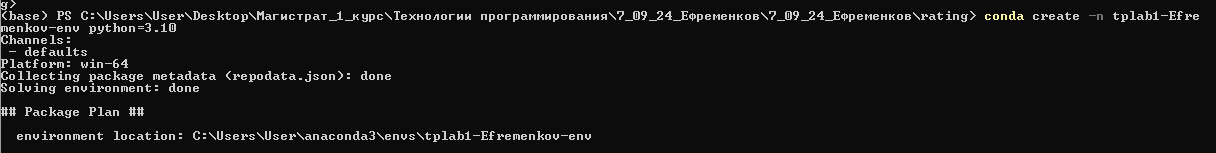
6. Запустите командный интерпретатор **Anaconda Powershell Prompt** (см. рисунок 1).

7. Перейдите в командном интерпретаторе в директорию с проектом:

**>** cd C:\Users\...\rating

8. Создайте виртуальное окружение Python с версией Python 3.10. Если работа осуществляется на компьютере общего пользования, имя создаваемого окружения должно содержать фамилию пользователя:

**>** conda create -n **tplab1-env** python=3.10

Подтвердите установку нажатием клавиши y. 

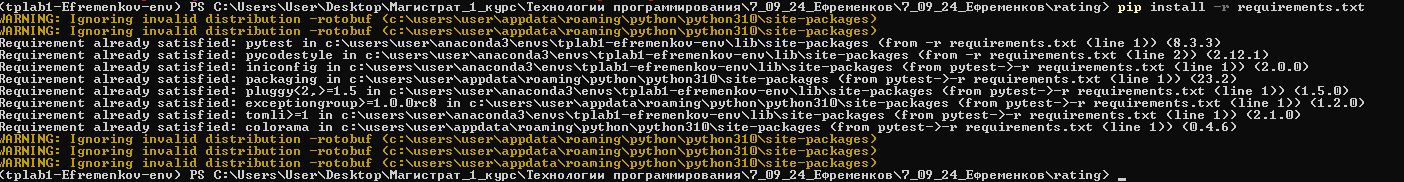
9. Убедитесь в успешной установке окружения. Активируйте созданное окружение командой:

**> conda activate tplab1-env (tplab1-Efremenkov-env)**



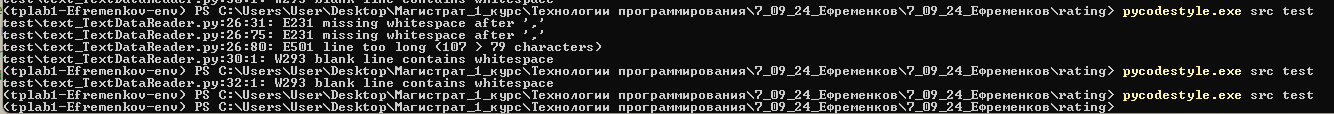
10. Установите в созданном виртуальном окружении необходимые пакеты:

**>** pip install –r **requirements.txt**



11. Проверьте соответствие программного кода стандарту PEP8. Если при копировании содержимого файлов нарушилось оформление, то может наблюдаться несоответствие проекта стандарту. В этом случае, возникшие ошибки необходимо исправить, руководствуясь их описанием.

**> pycodestyle.exe src test**



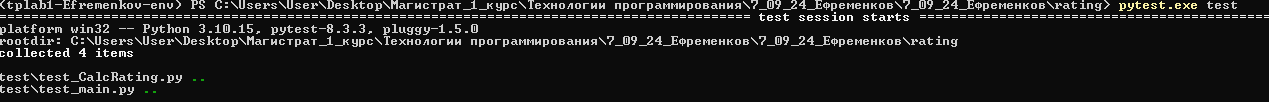
12. Для корректного определения путей к исходным файлам установите переменную окружения PYTHONPATH:

**> $env:PYTHONPATH = "**./:./src"



13. Запустите тестирование проекта и проверьте вывод на отсутствие ошибок:

**> pytest.exe test**



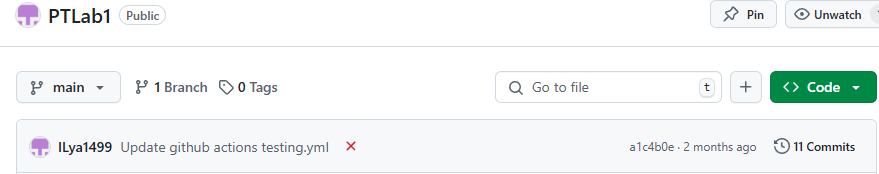
14. Запустите программу и проверьте ее работоспособность:

**>** python.exe .\src\main.py –p .\data\data.txt



15. Зайдите на сайт **https://github.com** под своей учетной записью (см. рисунок 2). Если ее нет, зарегистрируйтесь на сайте. В процессе регистрации необходимо указать действующий адрес электронной почты (на него придет код подтверждения) и придумать имя пользователя и пароль.

16. После регистрации или авторизации появится область, показанная на рисунке 3. Создайте новый репозиторий, нажав на кнопку «**Create repository**».



17. Укажите параметры создаваемого репозитория, как показано на рисунке 4. Имя репозитория: **PTLab1**, репозиторий публичный. После успешного создания репозитория появится область, приведенная на рисунке 5, со списком возможных команд для работы с ним.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Теперь необходимо настроить консольный клиент Git на компьютере с проектом rating. Запустите консольный клиент **Git CMD** через меню «Пуск», как показано на рисунке 1, и зайдите в директорию с проектом:

22. Теперь необходимо настроить консольный клиент Git на компьютере с проектом rating. Запустите консольный клиент **Git CMD** через меню «Пуск», как показано на рисунке 1, и зайдите в директорию с проектом:

**> cd C:\Users\...\rating **

23. Создайте новый локальный репозиторий в папке с проектом:

**>** git init

После выполнения этой команды в директории должна появиться папка **.git**.

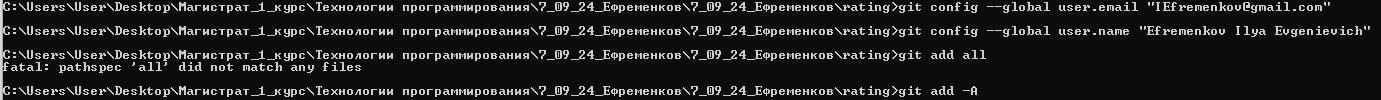
24. Сконфигурируйте имя пользователя и почтовый адрес клиента Git:

**>** git config --global user.email "**Ivanov@example.com**"

**>** git config --global user.name "**Ivanov Ivan Ivanovich**"

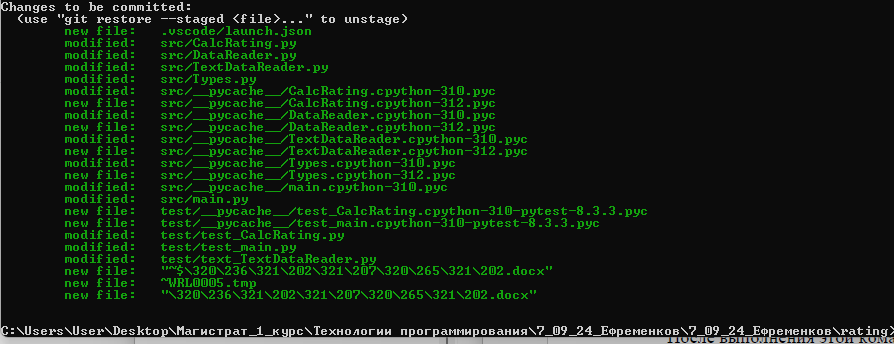
25. Проиндексируйте все файлы и папки, содержащиеся в директории rating:

**> git add all**

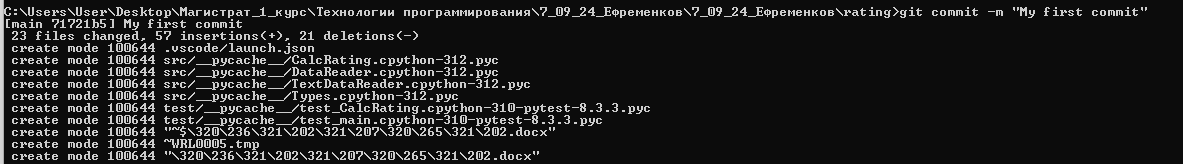


26. Проверьте статус репозитория:

**> git status**



27. Зафиксируйте вновь добавленные файлы в локальной части распределенной системы контроля версий Git, указав комментарий (желательно, осмысленный). Проверьте новый статус репозитория.

**>** git commit –m "**My first commit**" 

**> git status**

28. Переименуйте основную ветку проекта в main. Вновь проверьте статус репозитория.

**>** git branch –M main

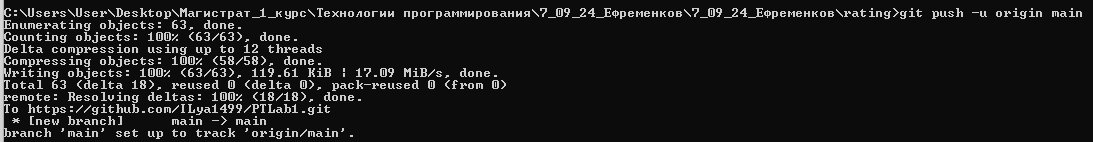
**> git status**

29. Укажите клиенту Git путь к удаленному репозиторию с псевдонимом origin:

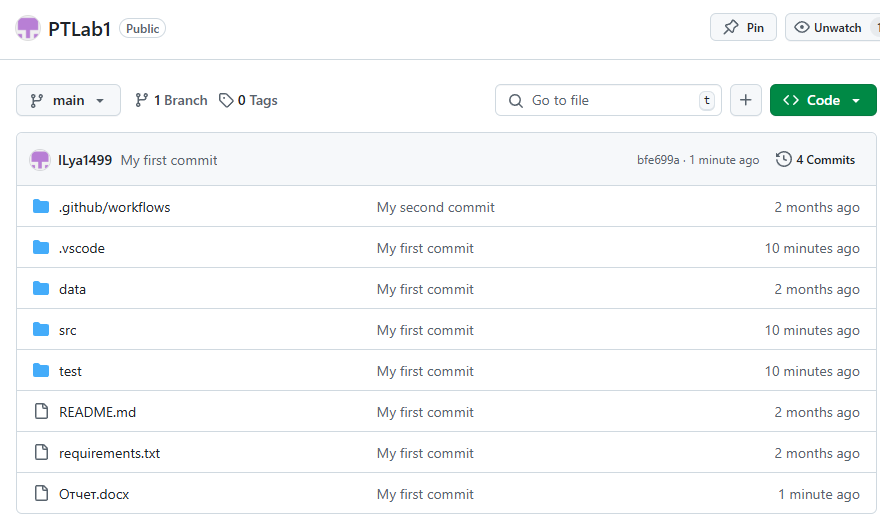
**> git remote add origin** https://github.com/<username>/PTLab1.git

30. Передайте изменения в локальной части репозитория в его серверную часть:

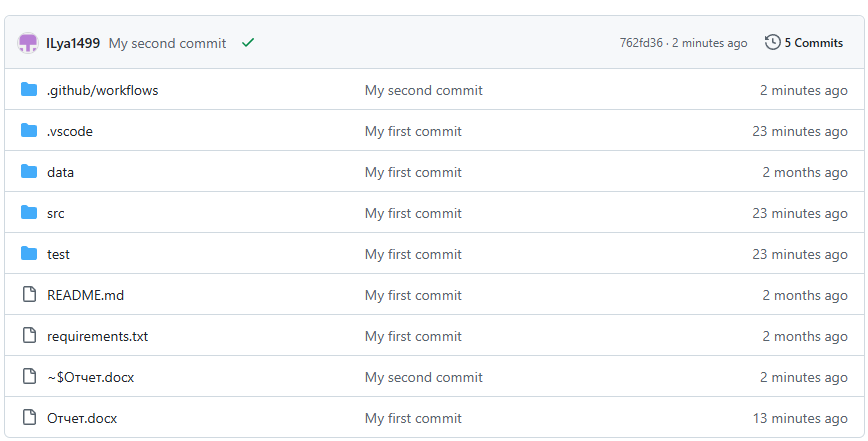
> git push –u origin main



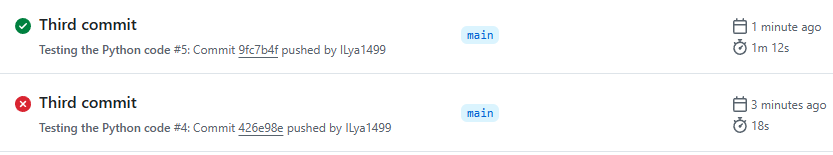
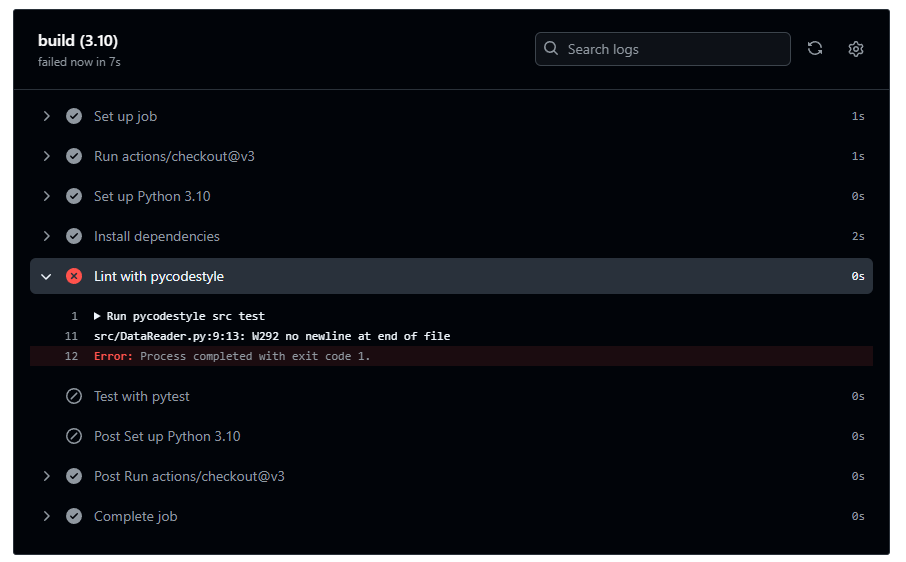
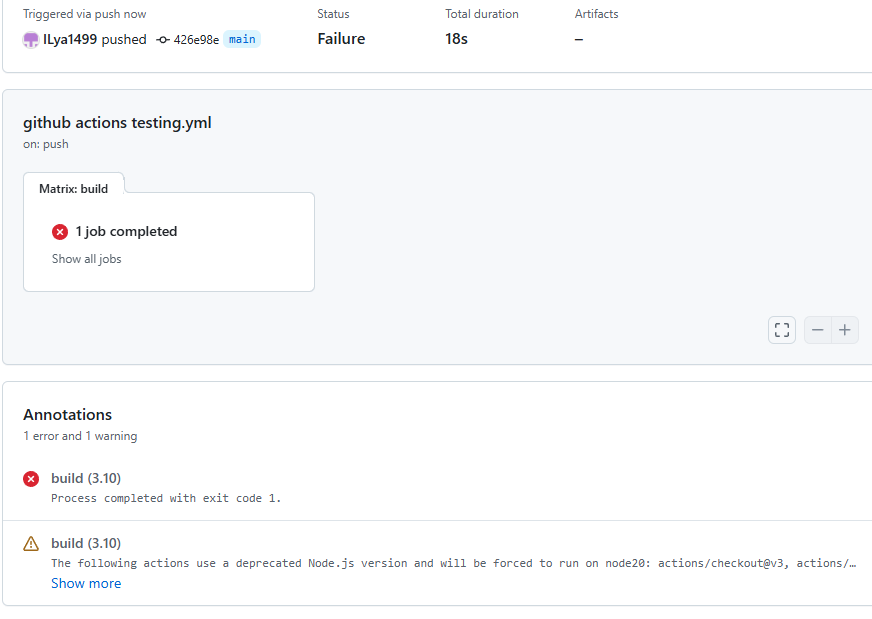
31. Откройте сайт **https://github.com** и проверьте наличие проекта в глобальном репозитории (см. рисунок 9).



Обновите страницу удаленного репозитория GitHub. Если автоматизированное тестирование с помощью GitHub Actions началось, в заголовке проекта будет наблюдаться желтая точка. После окончания тестирования она сменится или зеленой галочкой, показывающей успешное прохождение тестов (см. рисунок 10), или красным крестиком, символизирующим неудачу

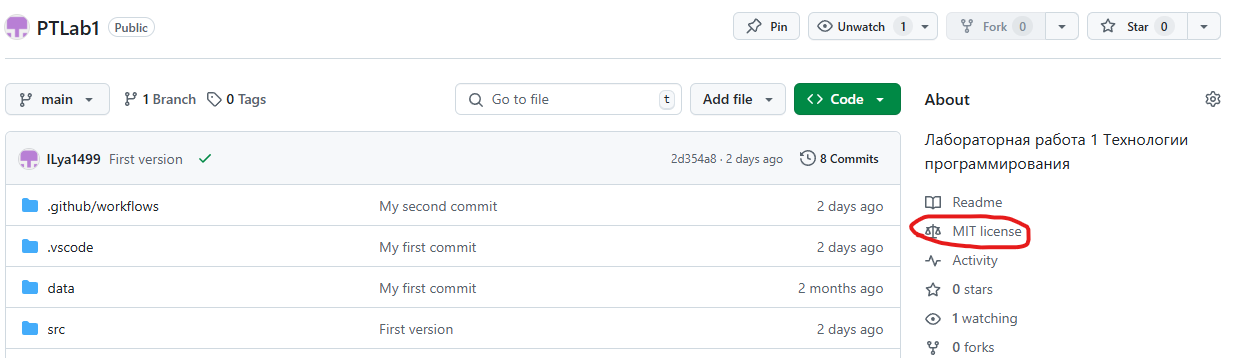


Добавьте ошибку в тексты исходных кодов, зафиксируйте изменения в Git и дождитесь отрицательного результата автоматизированного тестирования. Ознакомьтесь с выводом скрипта тестирования и иконкой статуса. Удалите ошибку и вновь зафиксируйте изменения. Убедитесь в успешном прохождении тестирования проекта.



Индивидуальное задание (6 вариант):

1.Выберите для Вашего проекта тип лицензии и добавьте файл с лицензией в проект.



2. Добавьте в проект файл .**gitignore** и сформируйте его содержимое.

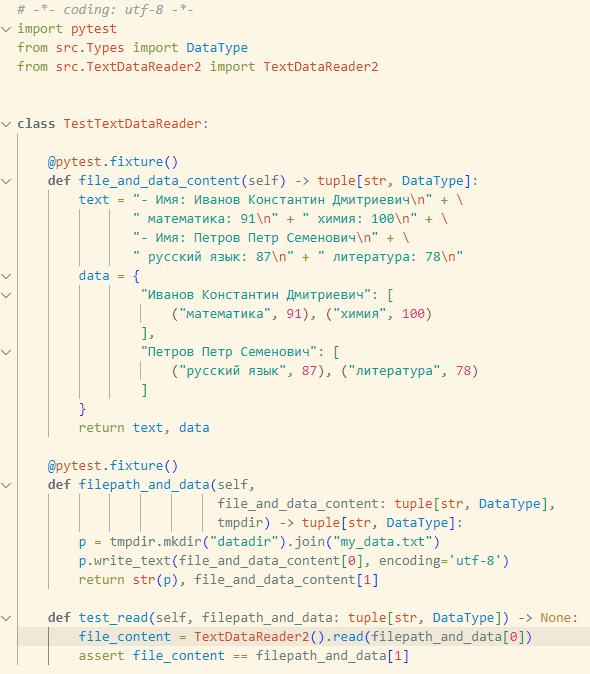


3. Добавьте в проект еще один класс – наследник класса DataReader, который должен обрабатывать входной файл определенного формата (*согласно индивидуальному варианту, см. таблицу*). Составьте модульные тесты для методов этого класса, постарайтесь покрыть тестами максимально возможный объем кода. Для работы с этим заданием создайте новую ветку кода на основе главной и фиксируйте в нее весь программный код в процессе разработки. Добейтесь выполнения всех тестов проекта, после чего объедините текущую ветку кода с главной.

Новый класс, наследник DataReader



Тесты TextDataReader2



Новая ветка кода



Модульные тесты

