

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 № 2.15**  
**дисциплины «Анализ данных»**  
**Вариант 15**

Выполнил:  
Кенесбаев Хилол Куат улы  
2 курс, группа ИВТ-б-о-22-1,  
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 \_\_\_\_\_

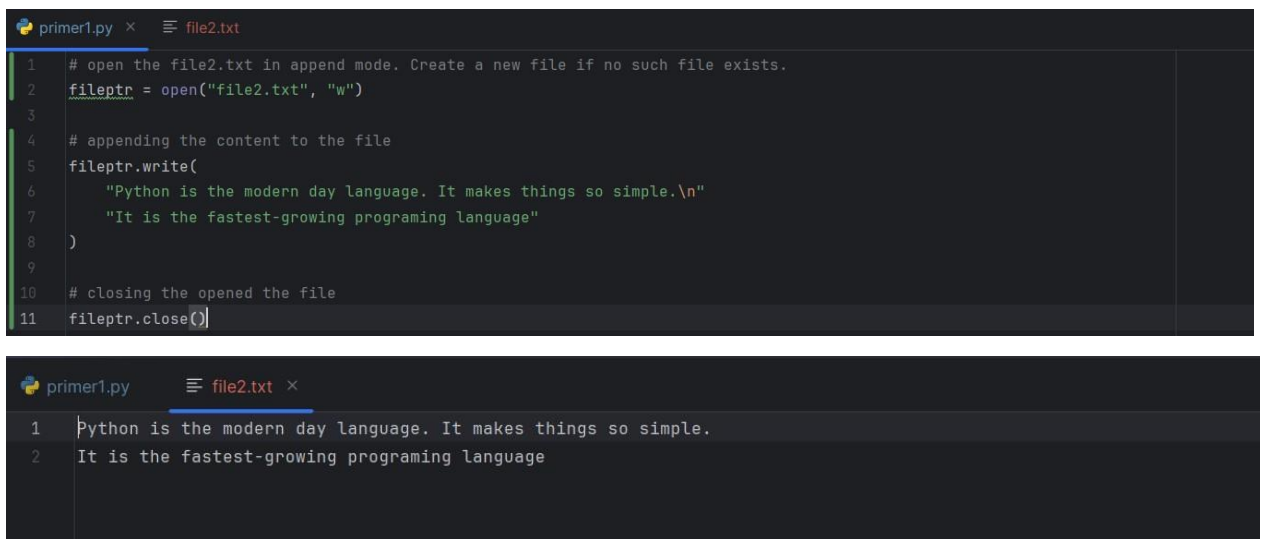
Ставрополь, 2024 г.

## Тема: Работа с файлами в языке Python

**Цель работы:** приобретение навыков по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучение основных методов модуля os для работы с файловой системой, получение аргументов командной строки.

### Порядок выполнения работы:

#### Пример 1:



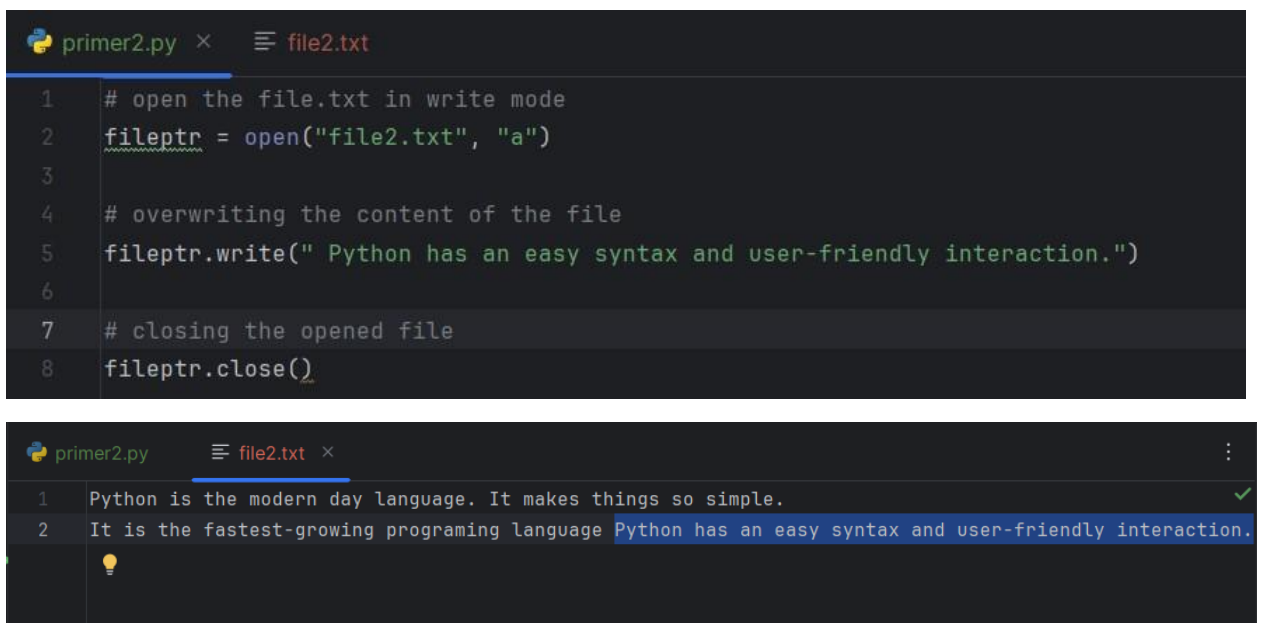
The image shows two screenshots from a code editor. The top screenshot displays the code in `primer1.py`, which opens `file2.txt` in append mode, writes two lines of text, and closes the file. The bottom screenshot shows the contents of `file2.txt`, which now contains the two lines of text written by the script.

```
1 # open the file2.txt in append mode. Create a new file if no such file exists.
2 fileptr = open("file2.txt", "a")
3
4 # appending the content to the file
5 fileptr.write(
6     "Python is the modern day language. It makes things so simple.\n"
7     "It is the fastest-growing programming language"
8 )
9
10 # closing the opened the file
11 fileptr.close()
```

```
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programming language
```

Рисунок 1. Результат работы

#### Пример 2:



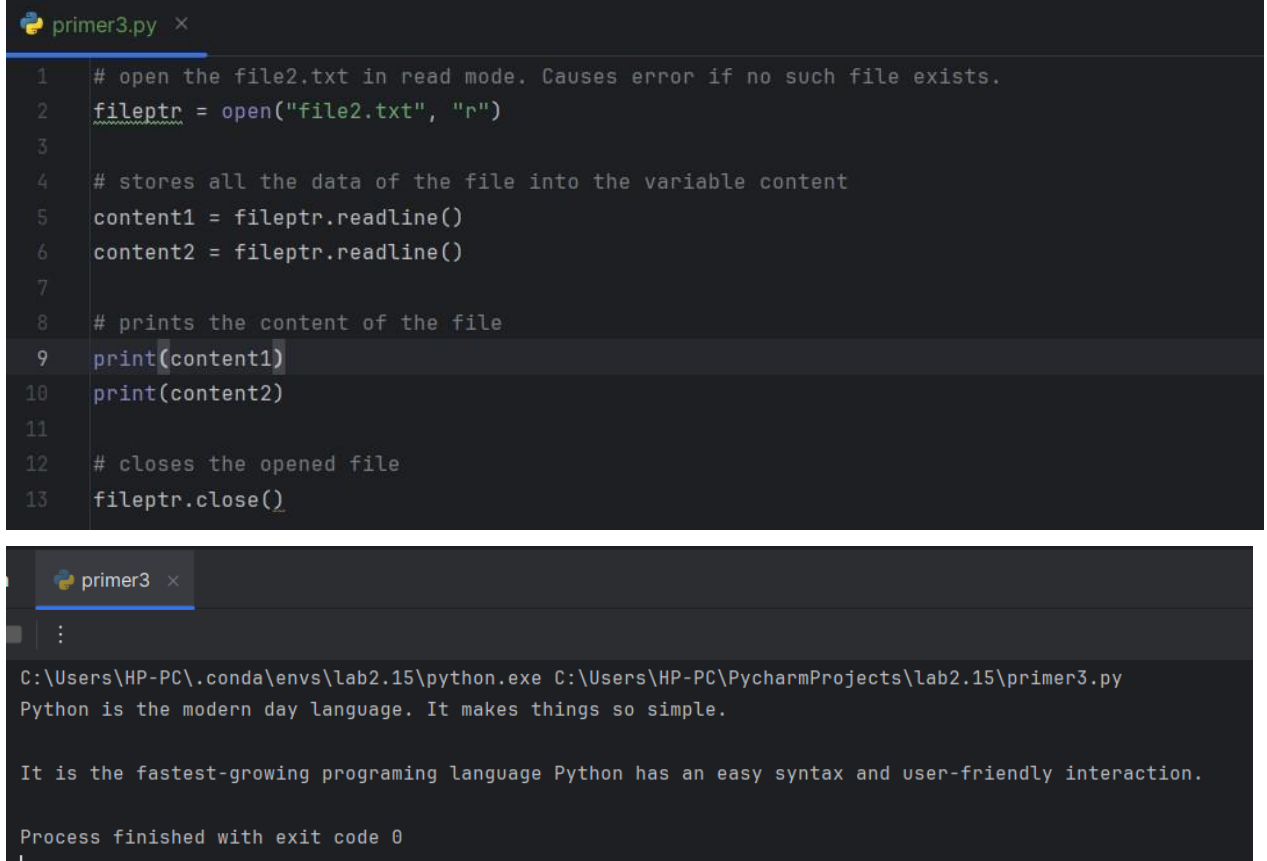
The image shows two screenshots from a code editor. The top screenshot displays the code in `primer2.py`, which opens `file2.txt` in write mode, overwrites the content with a single line, and closes the file. The bottom screenshot shows the contents of `file2.txt`, which now contains the single line of text written by the script.

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

```
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programming language Python has an easy syntax and user-friendly interaction.
```

Рисунок 2. Результат работы

### Пример 3:



```
primer3.py x
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 content1 = fileptr.readline()
6 content2 = fileptr.readline()
7
8 # prints the content of the file
9 print(content1)
10 print(content2)
11
12 # closes the opened file
13 fileptr.close()
```

primer3 x

C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\primer3.py

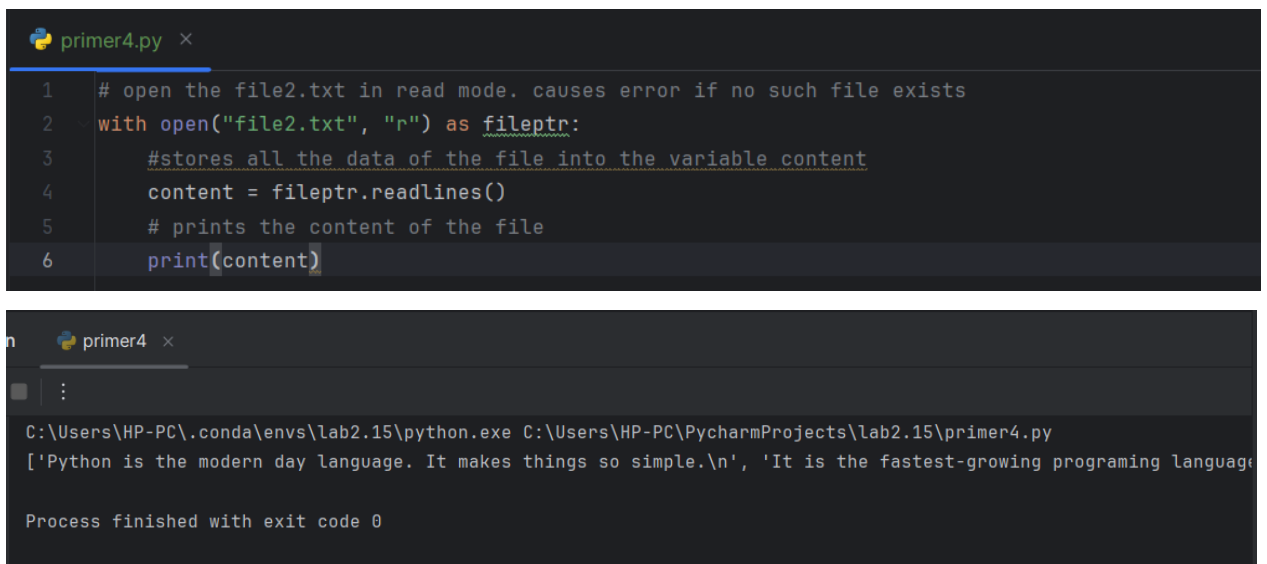
Python is the modern day language. It makes things so simple.

It is the fastest-growing programing language Python has an easy syntax and user-friendly interaction.

Process finished with exit code 0

Рисунок 3. Результат работы

### Пример 4:



```
primer4.py x
1 # open the file2.txt in read mode. causes error if no such file exists
2 with open("file2.txt", "r") as fileptr:
3     #stores all the data of the file into the variable content
4     content = fileptr.readlines()
5     # prints the content of the file
6     print(content)
```

primer4 x

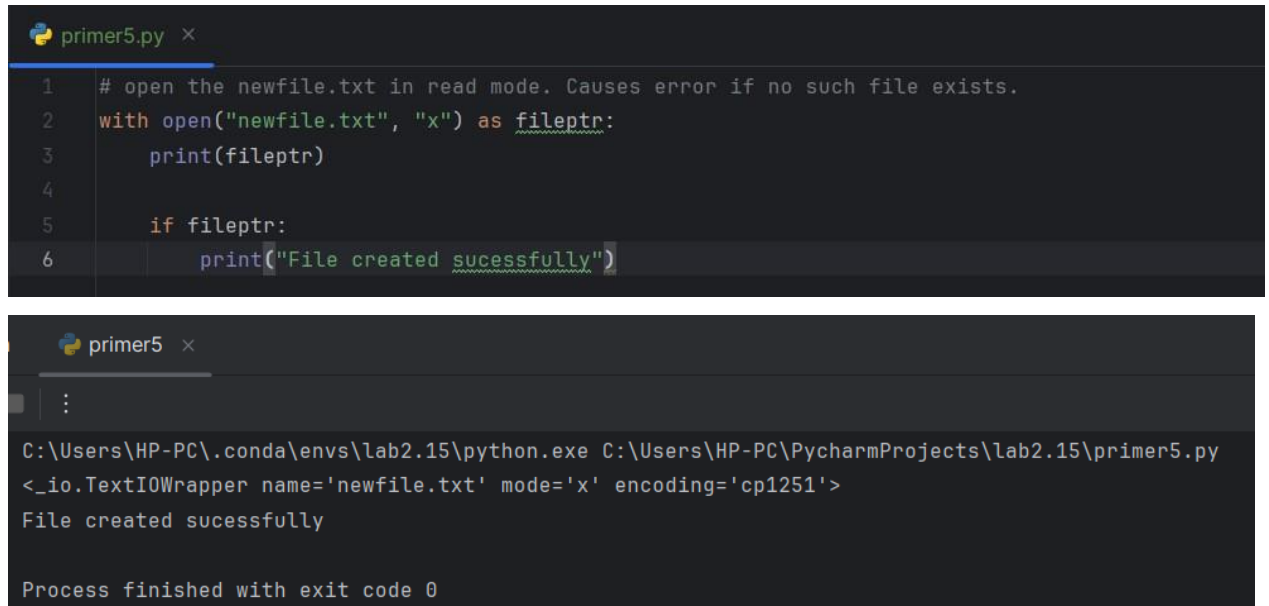
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\primer4.py

['Python is the modern day language. It makes things so simple.\n', 'It is the fastest-growing programing language']

Process finished with exit code 0

Рисунок 4. Результат работы

## Пример 5:



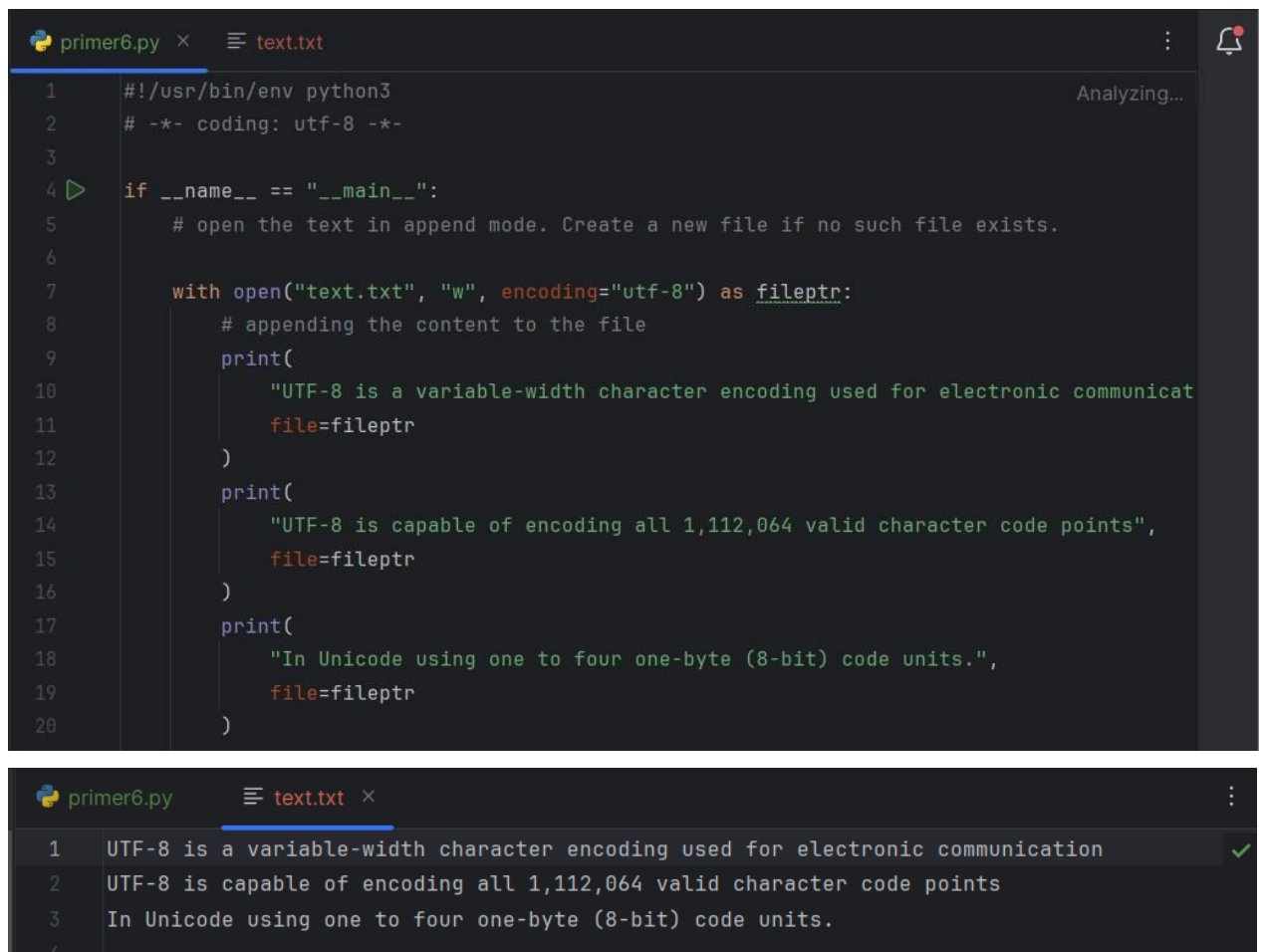
```
primer5.py x
1 # open the newfile.txt in read mode. Causes error if no such file exists.
2 with open("newfile.txt", "x") as fileptr:
3     print(fileptr)
4
5     if fileptr:
6         print("File created sucessfully")

primer5 x
:
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\primer5.py
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='cp1251'>
File created sucessfully

Process finished with exit code 0
```

Рисунок 5. Результат работы

## Пример 6:

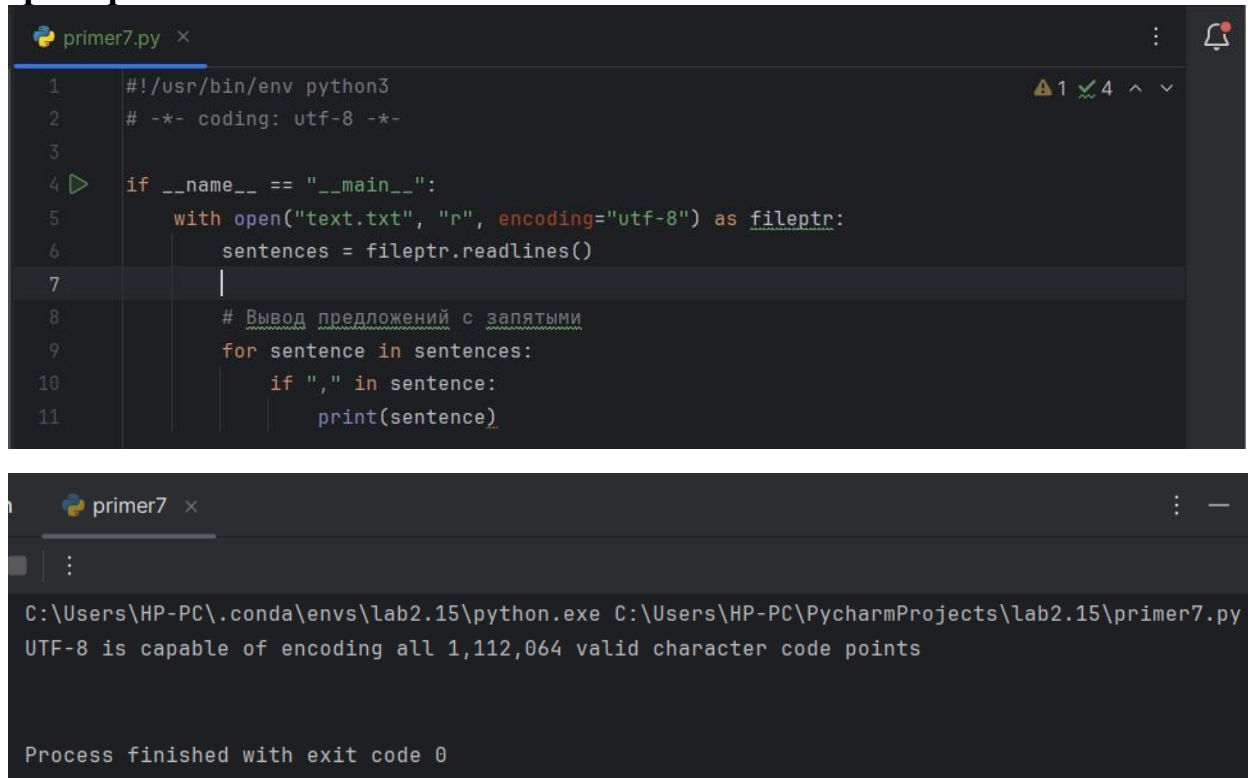


```
primer6.py x text.txt
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == "__main__":
5     # open the text in append mode. Create a new file if no such file exists.
6
7     with open("text.txt", "w", encoding="utf-8") as fileptr:
8         # appending the content to the file
9         print(
10             "UTF-8 is a variable-width character encoding used for electronic communicat
11             file=fileptr
12         )
13         print(
14             "UTF-8 is capable of encoding all 1,112,064 valid character code points",
15             file=fileptr
16         )
17         print(
18             "In Unicode using one to four one-byte (8-bit) code units.",
19             file=fileptr
20         )

primer6.py text.txt x
1 UTF-8 is a variable-width character encoding used for electronic communication ✓
2 UTF-8 is capable of encoding all 1,112,064 valid character code points
3 In Unicode using one to four one-byte (8-bit) code units.
4
```

Рисунок 6. Результат работы

## Пример 7:



The image shows two screenshots from a code editor. The top screenshot displays the source code for `primer7.py`. The code opens a file named `text.txt` in read mode, reads all lines into a list called `sentences`, and then iterates over each sentence, printing it if it contains a comma. The bottom screenshot shows the terminal output of running the script. It displays the command used to execute the file and the output of the `print` statements, which are the sentences from `text.txt`.

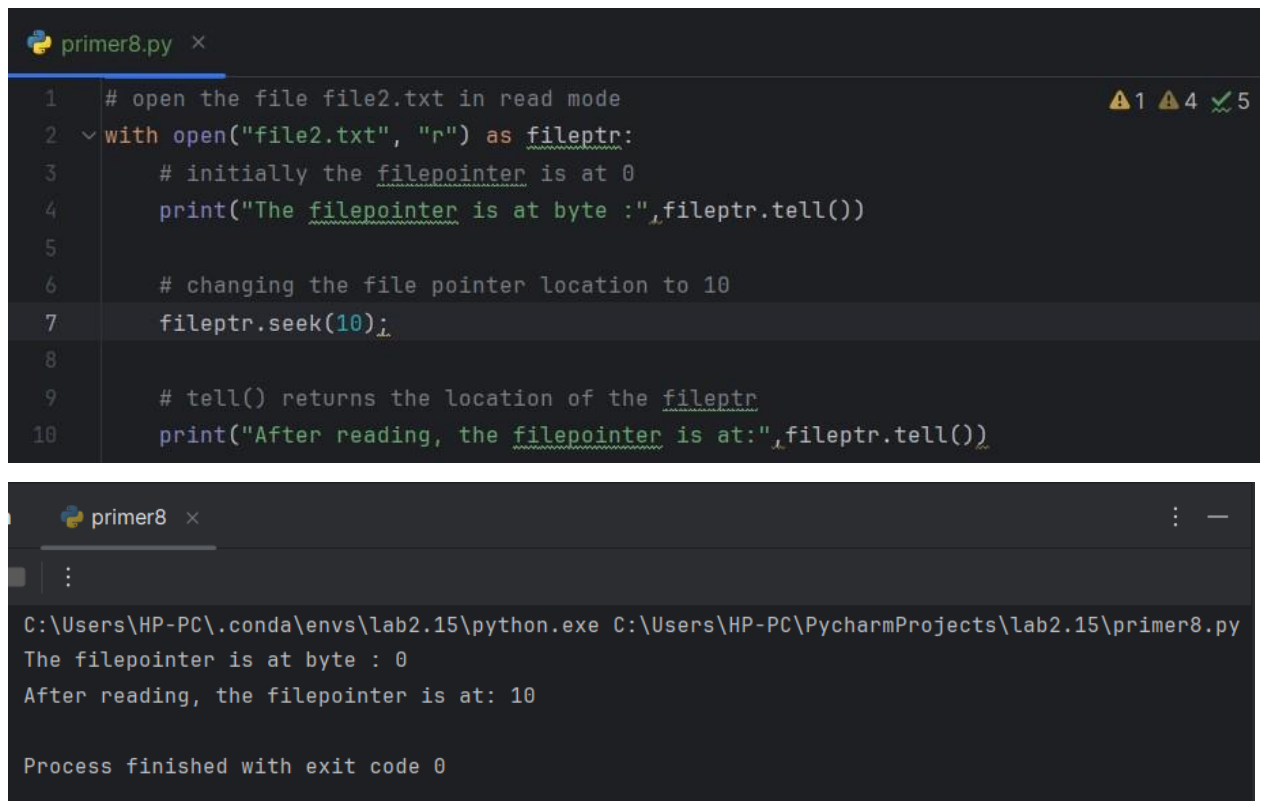
```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == "__main__":
5      with open("text.txt", "r", encoding="utf-8") as fileptr:
6          sentences = fileptr.readlines()
7
8          # Вывод предложений с запятыми
9          for sentence in sentences:
10             if "," in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

```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\primer7.py
UTF-8 is capable of encoding all 1,112,064 valid character code points

Process finished with exit code 0
```

Рисунок 7. Результат работы

## Пример 8:



The image shows two screenshots from a code editor. The top screenshot displays the source code for `primer8.py`. The code opens a file named `file2.txt` in read mode, prints the current file pointer position (0), changes the position to 10 using `seek(10)`, and then prints the position again (10). The bottom screenshot shows the terminal output of running the script. It displays the command used to execute the file and the output of the `print` statements, which are the file pointer positions before and after seeking.

```
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
6      # changing the file pointer location to 10
7      fileptr.seek(10);
8
9      # tell() returns the location of the fileptr
10     print("After reading, the filepointer is at:", fileptr.tell())
```

```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\primer8.py
The filepointer is at byte : 0
After reading, the filepointer is at: 10

Process finished with exit code 0
```

Рисунок 8. Результат работы

### Пример 9:

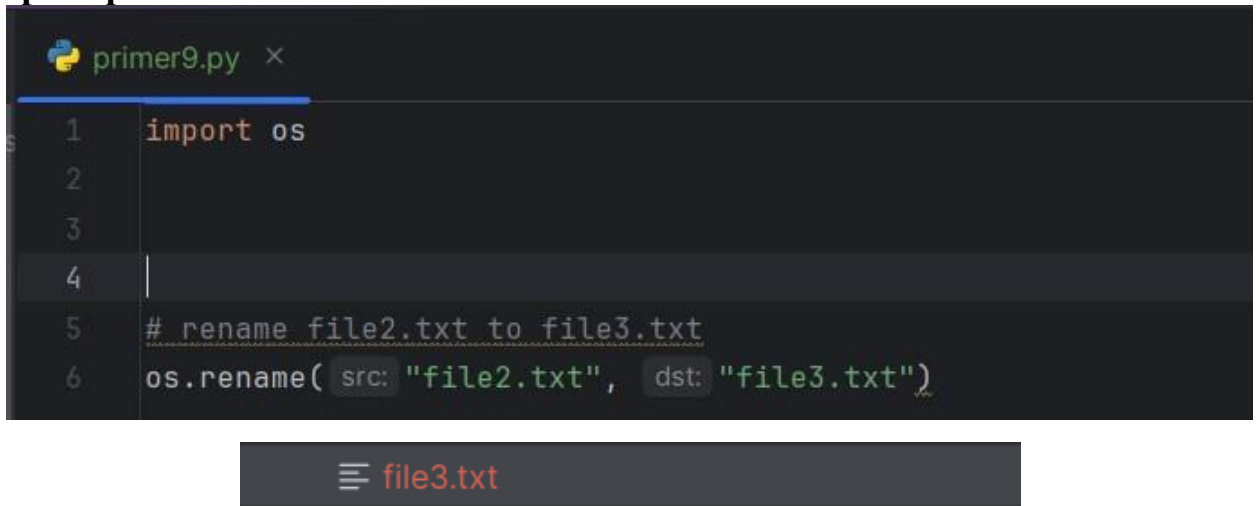


Рисунок 9. Результат работы

### Пример 10:

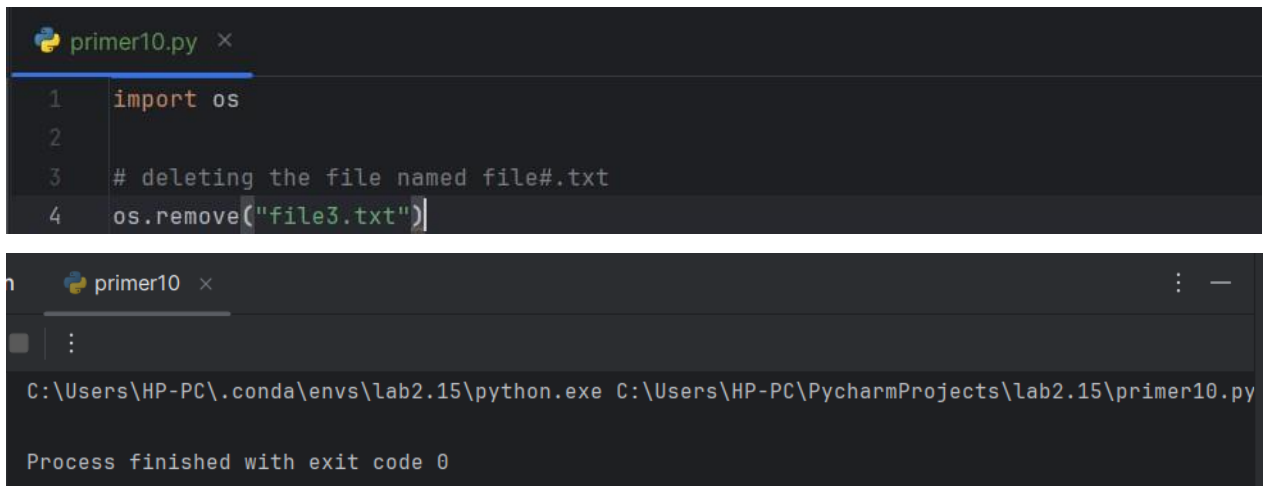


Рисунок 10. Результат работы

### Пример 11:

```
primer11.py ×  
1 import os  
2  
3 # creating a new directory with the name new  
4 os.mkdir("new")
```

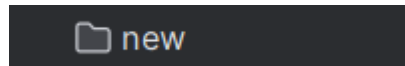


Рисунок 11. Результат работы

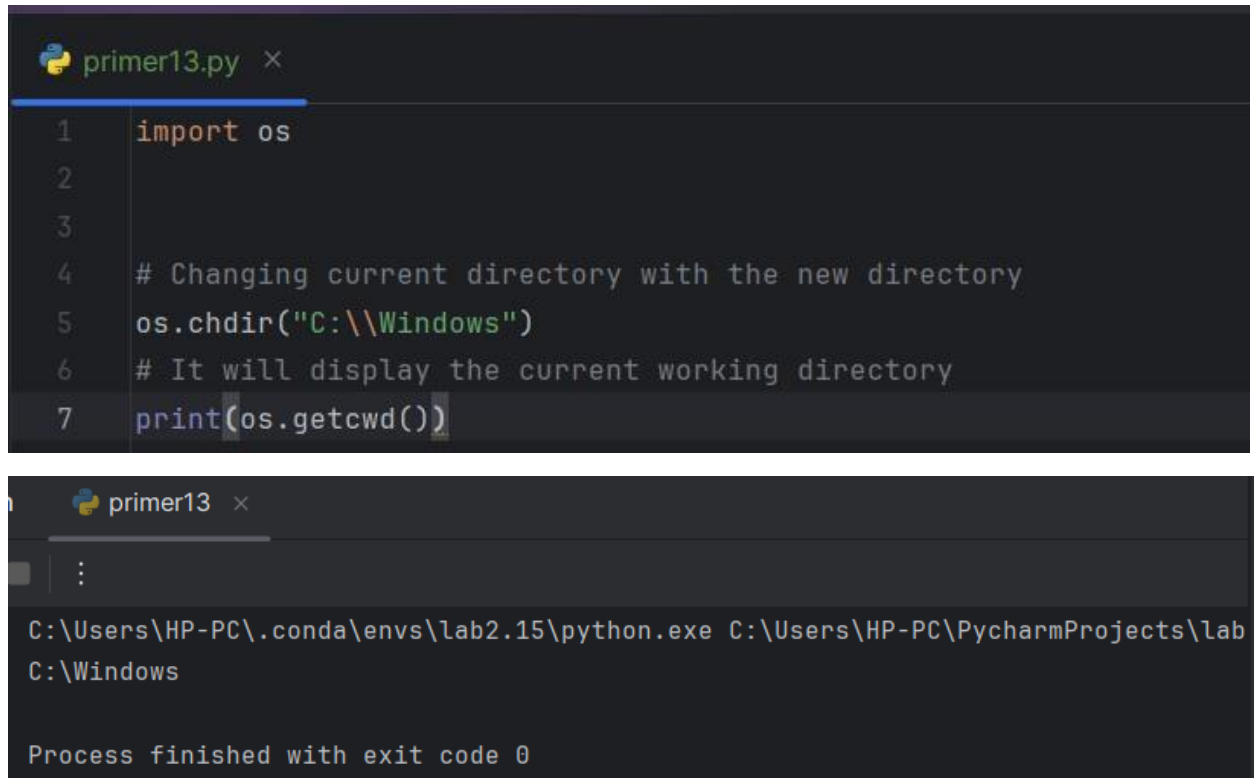
### Пример 12:

```
primer12.py ×  
1 import os  
2  
3 path = os.getcwd()  
4 print(path)
```

```
primer12 ×  
:  
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab  
C:\Users\HP-PC\PycharmProjects\lab2.15  
  
Process finished with exit code 0
```

Рисунок 12. Результат работы

### Пример 13:



The image shows two windows from a PyCharm IDE. The top window, titled 'primer13.py', contains the following Python code:

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

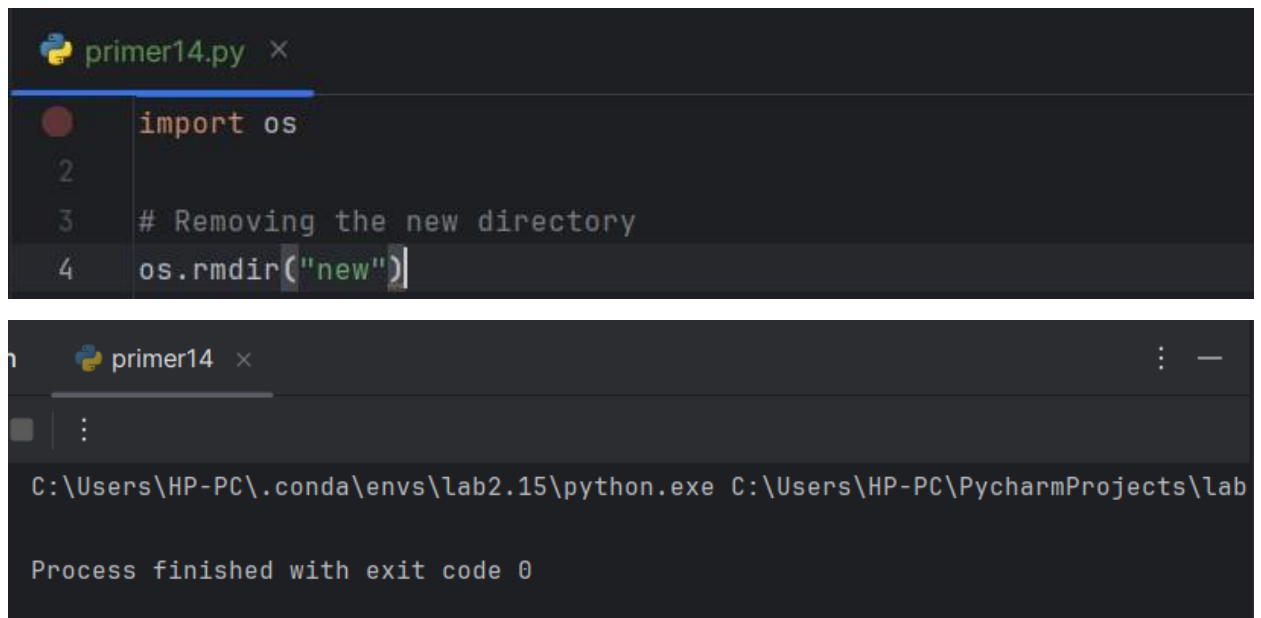
The bottom window, titled 'primer13', shows the terminal output:

```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab
C:\Windows

Process finished with exit code 0
```

Рисунок 13. Результат работы

### Пример 14:



The image shows two windows from a PyCharm IDE. The top window, titled 'primer14.py', contains the following Python code:

```
1 import os
2
3 # Removing the new directory
4 os.rmdir("new")
```

The bottom window, titled 'primer14', shows the terminal output:

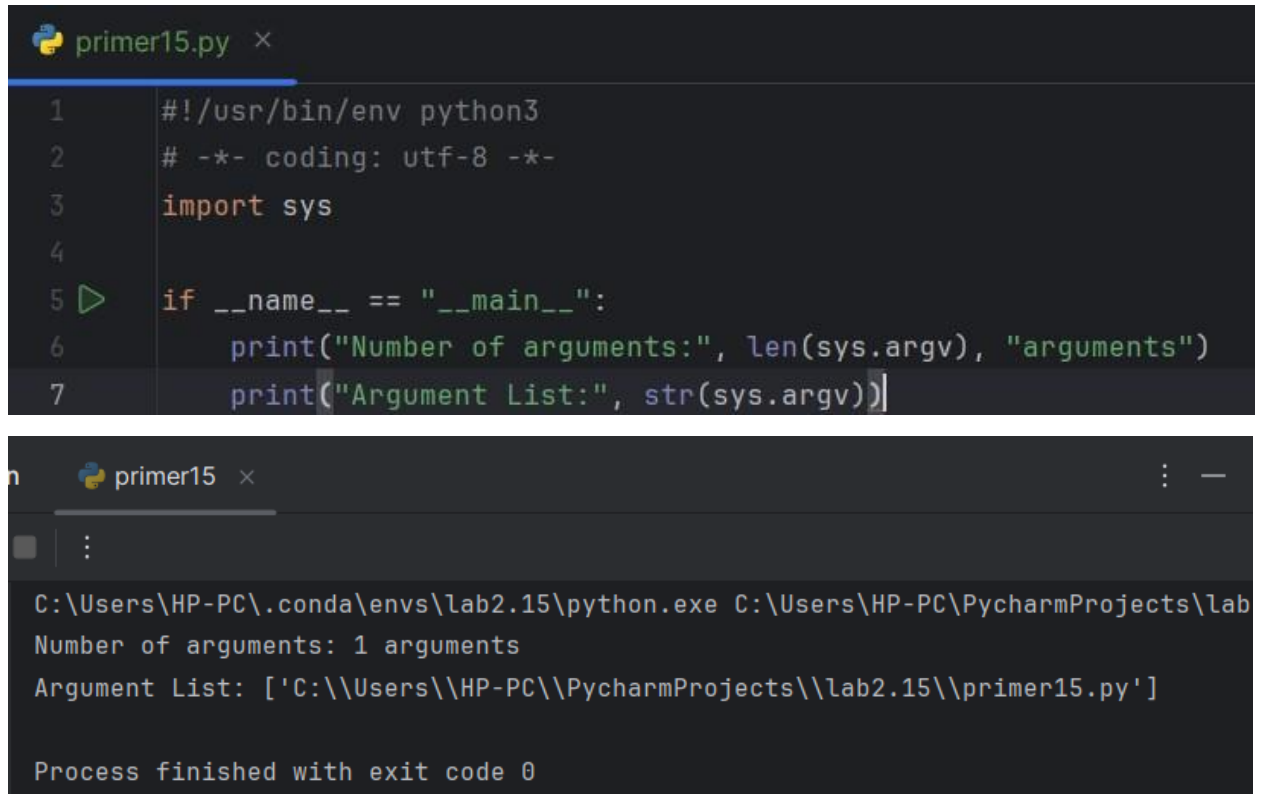
```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab

Process finished with exit code 0
```

Рисунок 14. Результат работы



### Пример 15:



The image shows a code editor window titled 'primer15.py' with the following Python code:

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

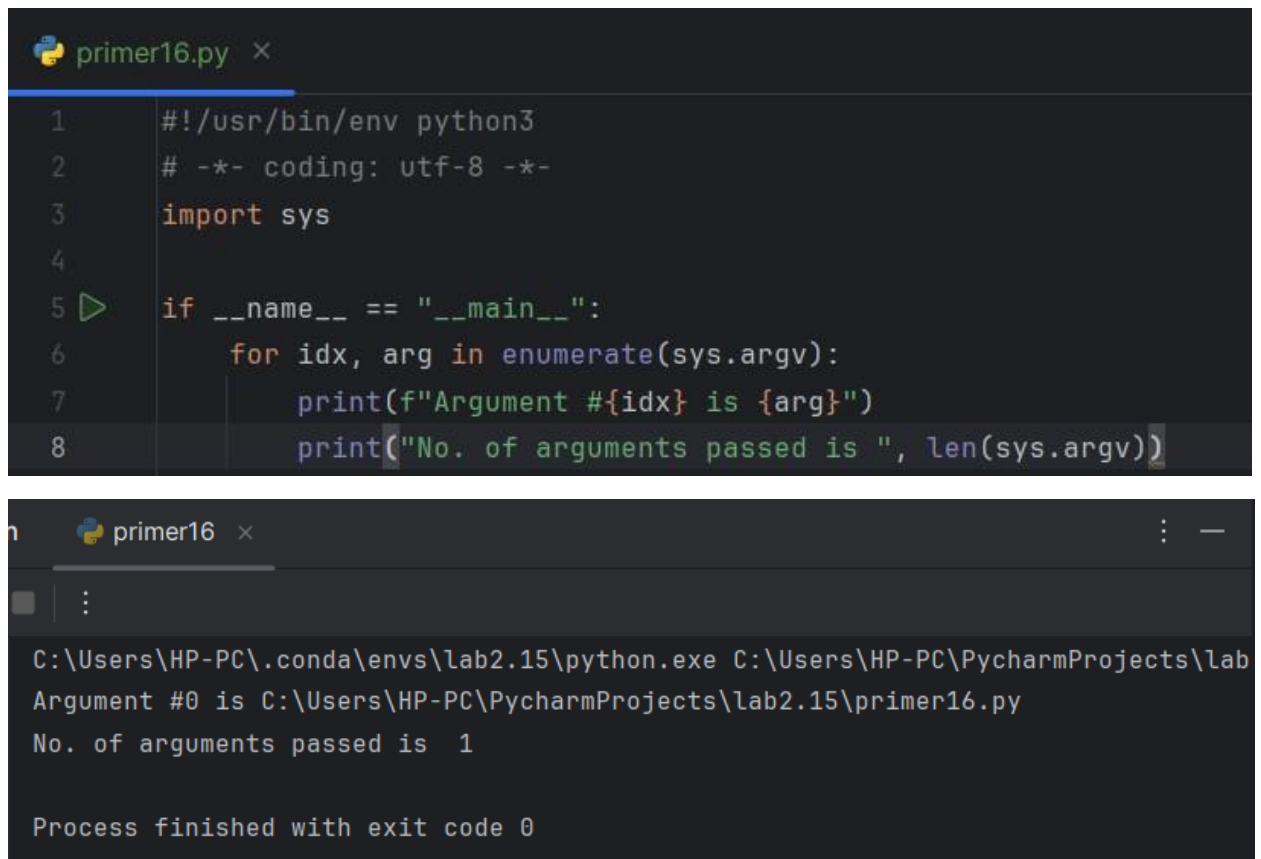
Below the code editor is a terminal window showing the execution of the script. The command prompt shows the path to the Python interpreter and the script file. The output of the script is displayed in the terminal:

```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab
Number of arguments: 1 arguments
Argument List: ['C:\\Users\\HP-PC\\PycharmProjects\\lab2.15\\primer15.py']

Process finished with exit code 0
```

Рисунок 15. Результат работы

### Пример 16:



The image shows a code editor window titled 'primer16.py' with the following Python code:

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

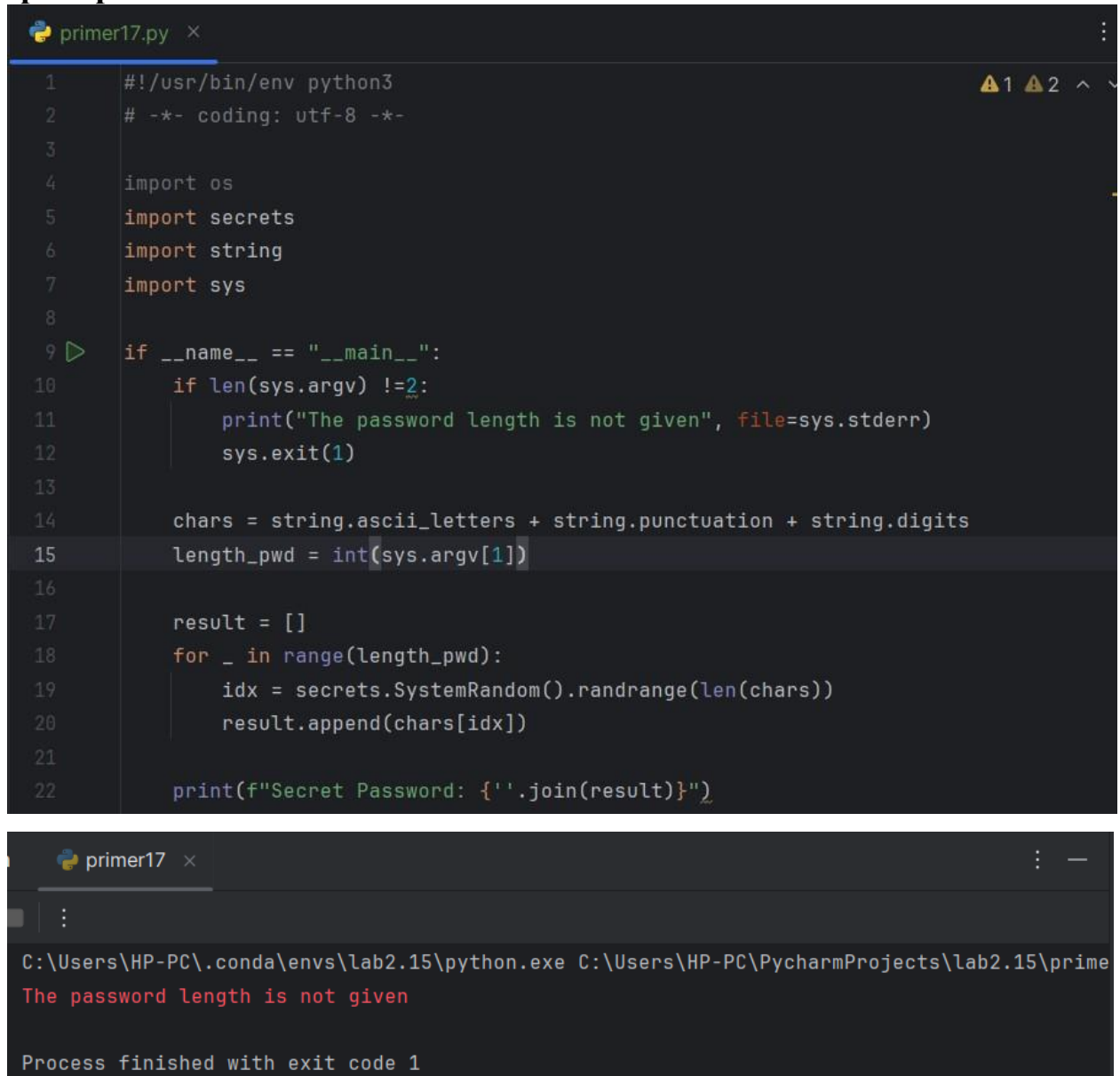
Below the code editor is a terminal window showing the execution of the script. The command prompt shows the path to the Python interpreter and the script file. The output of the script is displayed in the terminal:

```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab
Argument #0 is C:\Users\HP-PC\PycharmProjects\lab2.15\primer16.py
No. of arguments passed is 1

Process finished with exit code 0
```

Рисунок 16. Результат работы

## Пример 17:



The image shows two screenshots from a PyCharm IDE. The top screenshot displays the source code of a Python script named `primer17.py`. The script generates a random password of a specified length. The bottom screenshot shows the terminal output of the script, demonstrating an error handling case where the password length is not provided.

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

Terminal output:

```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\prime
The password length is not given

Process finished with exit code 1
```

Рисунок 17. Результат работы

## Сделал индивидуальное задание 1:

Условие : Написать программу, которая считывает текст из файла и выводит на экран сначала вопросительные, а затем восклицательные предложения

```
def get_type(sentence):
    if sentence.endswith('?'):
        return '?'
    elif sentence.endswith('!'):
        return '!'
    else:
        return None

# Чтение текста из файла
file_path = 'example.txt'
with open(file_path, 'r', encoding='utf-8') as file:
    text = file.read()

# Разделение текста на предложения
sentences = [sentence.strip() for sentence in text.split('.')]

# Разделение по типу предложения
question_sentences = []
exclamation_sentences = []

for sentence in sentences:
    sentence_type = get_type(sentence)
    if sentence_type == '?':
        question_sentences.append(sentence)
    elif sentence_type == '!':
        exclamation_sentences.append(sentence)

# Вывод вопросительных предложений
print("Вопросительные предложения:")
for sentence in question_sentences:
    print(sentence)

# Вывод восклицательных предложений
print("\nВосклицательные предложения:")
for sentence in exclamation_sentences:
    print(sentence)
```

Рисунок 18. Код индивидуальной работы 1

```
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\prog\individual.py
Вопросительные предложения:
Что такое инверсия ?

Восклицательные предложения:
Термин, обозначающий перестановку слов или обратное их написание !
```

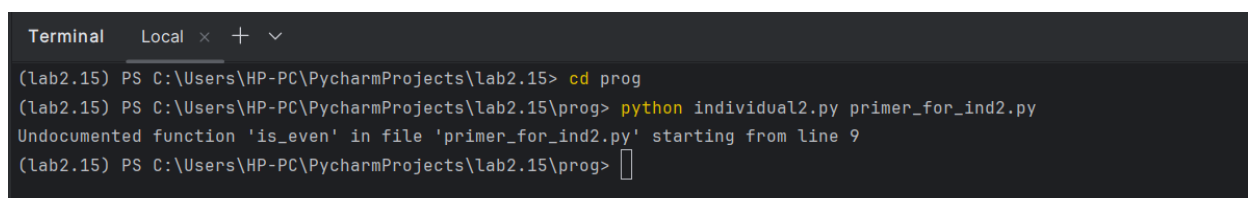
Рисунок 19. Результат работы программы

**Сделал индивидуальное задание 2 :** Напишите программу, которая будет проходить по файлу с исходным кодом на Python и искать функции, не снабженные блоком комментариев. Можно принять за аксиому, что строка, начинающаяся со слова `def`, следом за которым идет пробел, будет считаться началом функции. И если функция документирована, предшествующая строчка должна начинаться со знака `#`. Перечислите названия всех функций, не снабженных комментариями, вместе с именем файла и номером строки, с которой начинается объявление функции. Одно или несколько имен файлов с кодом на языке Python пользователь должен передать в функцию в качестве аргументов командной строки. Для файлов, которые не существуют или не могут быть открыты, должны выдаваться соответствующие предупреждения, после чего должна быть продолжена обработка остальных файлов.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import argparse
5
6  parser = argparse.ArgumentParser(description='Find undocumented functions in Python code')
7  parser.add_argument('name_or_flags', 'files', nargs='+', help='List of Python files to process')
8  args = parser.parse_args()
9
10
11 # Обработка каждого файла из списка
12 for file_name in args.files:
13     try:
14         with open(file_name, 'r') as file:
15             # Чтение файла по строкам
16             lines = file.readlines()
17
18             # Цикл по строкам файла
19             for i, line in enumerate(lines):
20                 if line.strip().startswith('def '):
21                     if i > 0 and not lines[i - 1].strip().startswith('#'):
22                         print(
23                             f"Undocumented function '{line.split('(', 1)[0].split(' ')[1]}' in file '{file_name}' starting from line {i + 1}")
24     except FileNotFoundError:
25         print(f"File '{file_name}' not found.")
26     except IOError:
27         print(f"Could not open '{file_name}'.")
```

**Рисунок 20. Код индивидуальной работы 2**



```
Terminal Local x + v
(lab2.15) PS C:\Users\HP-PC\PycharmProjects\lab2.15> cd prog
(lab2.15) PS C:\Users\HP-PC\PycharmProjects\lab2.15\prog> python individual2.py primer_for_ind2.py
Undocumented function 'is_even' in file 'primer_for_ind2.py' starting from line 9
(lab2.15) PS C:\Users\HP-PC\PycharmProjects\lab2.15\prog> █
```

**Рисунок 21. Результат работы программы**

Придумал задачу с модулем os :

Нужно переименовать все файлы с расширением .txt, на change\_name:

```
os.py x
1 import os
2
3 folder_path = "C:/Users/HP-PC/PycharmProjects/lab2.15/os" # Путь к папке с файлами
4 file_extension = ".txt" # Расширение файлов, которые мы ищем
5
6 # Получаем список файлов с указанным расширением в папке
7 files = [f for f in os.listdir(folder_path) if
8           os.path.isfile(os.path.join(folder_path, f)) and f.endswith(file_extension)]
9
10 # Переименовываем каждый файл, добавляя префикс "change_name"
11 for file in files:
12     file_path = os.path.join(folder_path, file)
13     new_file_name = "change_name" + file
14     new_file_path = os.path.join(folder_path, new_file_name)
15
16     print(f"Old filename: {file} -> New filename: {new_file_name}")
17
18     # Переименование файла
19     os.rename(file_path, new_file_path)
```

Рисунок 22. Код программы

```
os x
C:\Users\HP-PC\.conda\envs\lab2.15\python.exe C:\Users\HP-PC\PycharmProjects\lab2.15\prog\os.py
Old filename: example.txt -> New filename: change_nameexample.txt
Old filename: example2.txt -> New filename: change_nameexample2.txt

Process finished with exit code 0
|
```

```

v  os
   change_nameexample.txt
   change_nameexample2.txt
```

Рисунок 23. Результат работы программы

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

Метод `open()` с названием файла и параметром “r”

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

Метод `open()` с названием файла и параметром “w”

**3. Как прочитать данные из файла в языке Python?**

Метод `read()` считывает строку из файла. Он может читать данные как в текстовом, так и в двоичном формате

**4. Как записать данные в файл в языке Python?**

Метод `open()` с параметром “w”

**5. Как закрыть файл в языке Python?**

Метод `close()`

**6. Изучите самостоятельно работу конструкции `with ... as`. Каково ее назначение в языке?**

Конструкция `with ... as` в языке Python предназначена для обеспечения управления ресурсами с автоматическим освобождением этих ресурсов после завершения блока кода. Она часто используется с объектами, которые поддерживают протокол менеджеров контекста.

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

Метод `writelines()`:

Данный метод принимает список строк в качестве аргумента и записывает каждую строку списка в файл.

В Python также существует метод `readinto()`, который читает данные из файла и записывает их в предварительно выделенный буфер. Это может быть полезно, если нужно работать с данными напрямую в буфере, минуя дополнительные копирования в памяти.

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

`os.chmod (path, mode, *, dir_fd=None, follow_symlinks=True)` - смена прав доступа к объекту (`mode` - восьмеричное число).

`os.chown (path, uid, gid, *, dir_fd=None, follow_symlinks=True)` - меняет id владельца и группы (Unix).

`os.link (src, dst, *, src_dir_fd=None, dst_dir_fd=None, follow_symlinks=True)` - создаёт жёсткую ссылку.

`os.listdir (path=".")` - список файлов и директорий в папке.

`os.makedirs (path, mode=0o777, exist_ok=False)` - создаёт директорию,

создавая при этом промежуточные директории.

**Вывод:** в результате выполнения программы были приобретены навыки по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучены основные методы модуля `os` для работы с файловой системой, изучено получение аргументов командной строки.