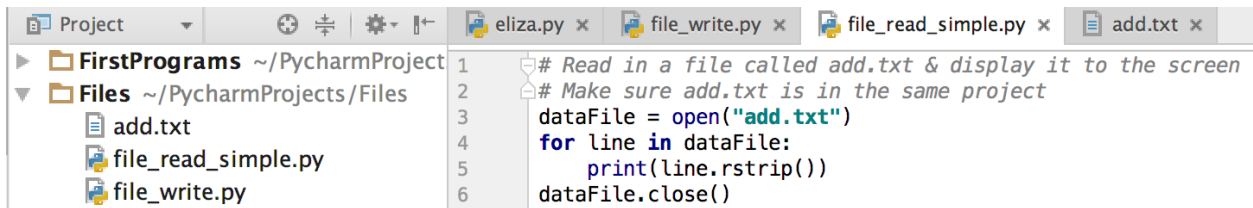


Reading/Writing Files Examples

23.code/file_read_simple.py

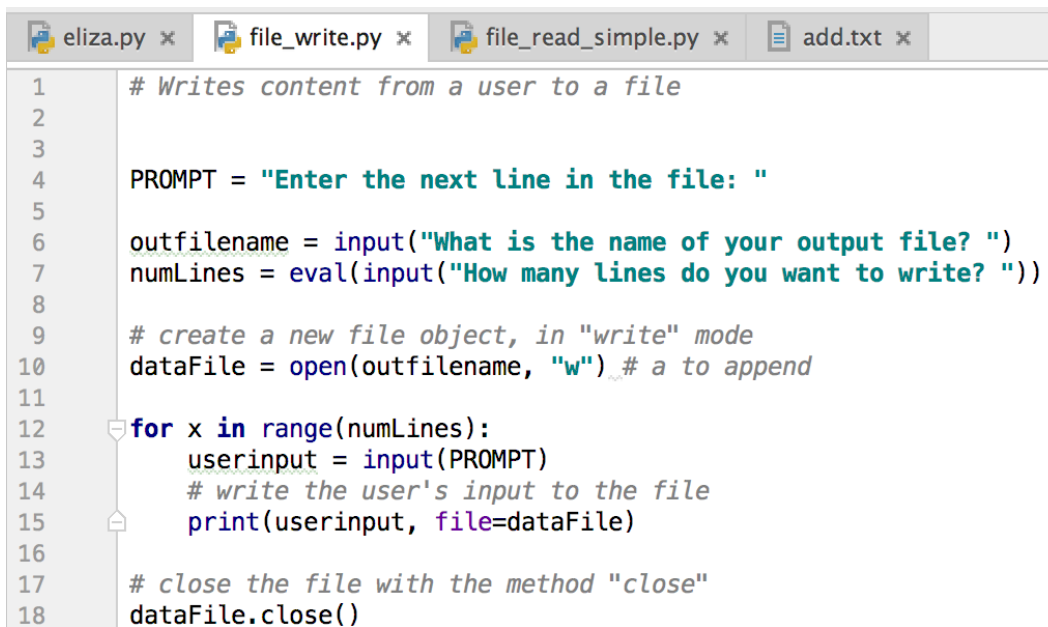
A screenshot of the PyCharm IDE interface. The top toolbar shows icons for Project, Run, Debug, Settings, and a file icon. The project browser on the left shows a project named 'FirstPrograms' with a subdirectory 'Files' containing 'add.txt', 'file_read_simple.py', and 'file_write.py'. The main editor window displays the code for 'file_read_simple.py' with line numbers 1 through 6. The code reads a file named 'add.txt' and prints its contents to the screen.

```
1 # Read in a file called add.txt & display it to the screen
2 # Make sure add.txt is in the same project
3 dataFile = open("add.txt")
4 for line in dataFile:
5     print(line.rstrip())
6 dataFile.close()
```

23.code/add.txt: <https://drive.google.com/file/d/1zdiM2rhlvikYc3GEtsP3ysMzX6UNJj5i/view?usp=sharing>

```
Hello
computer
We are learning computer science!
ha ha ha ha ha
```

23.code/file_write.py

A screenshot of the PyCharm IDE interface showing the code for 'file_write.py'. The top toolbar and project browser are visible. The main editor window displays the code for 'file_write.py' with line numbers 1 through 18. The code prompts the user for an output filename and the number of lines to write, then writes the user's input to the specified file.

```
1 # Writes content from a user to a file
2
3
4 PROMPT = "Enter the next line in the file: "
5
6 outfilename = input("What is the name of your output file? ")
7 numLines = eval(input("How many lines do you want to write? "))
8
9 # create a new file object, in "write" mode
10 dataFile = open(outfilename, "w") # a to append
11
12 for x in range(numLines):
13     userInput = input(PROMPT)
14     # write the user's input to the file
15     print(userinput, file=dataFile)
16
17 # close the file with the method "close"
18 dataFile.close()
```

Writing Files Practice

1. Write a python program that reads in a text file and prints all the lines back out with a dash in front.

Example program run:

```
> python3 read_file.py
```

Input File contents:

```
hello
goodbye
1234
```

Outputs:

```
-hello
-goodbye
-1234
```

2. Write a python program that asks the user for the name of a file, and then repeatedly asks the user to enter a number, entering the number '0' when finished. Output each of these numbers to the file on a separate line.

Example program run:

```
> python3 write_numbers.py
```

User inputs:

```
out.txt
25
34
14
0
```

Output file out . txt created with the following contents:

```
25
34
14
```

3. Write a python program that reads 3 files called test1.txt, text2.txt, and text3.txt, counts the number of lines in each file, and prints out the number of lines to a file counts.txt.

Example program run:

```
> python3 read3_files.py
```

text1.txt:

Hello
how are you?
Good.
See you later.

text2.txt:

Goodbye
Number 5
Nice knowing you

text3.txt:

End of semester
approaches!

Output file counts.txt created with the following contents:

```
text1.txt : 4  
text2.txt : 3  
text3.txt : 2
```

Hint: How to be DRY & not copy/paste the code 3 times? There are 2 approaches:

- a. Create a function with the filename as a parameter & call it 3 times
 - b. Create a list of the file names and loop over the file names
4. Create a program similar to above that counts the number of **words** in the files as well. After printing out information about each file, this program should also print the total number of lines & words in all 3 files. You can use the string split function to split a line of input into a list of words by splitting the line on spaces.

Example program run:

```
> python3 read3_files2.py
```

text1.txt:

Hello
how are you?
Good.
See you later.

text2.txt:

Goodbye
Number 5
Nice knowing you

text3.txt:

End of semester
approaches!

Output file counts.txt created with the following contents:

```
text1.txt : 4 lines, 8 words  
text2.txt : 3 lines, 6 words  
text3.txt : 2 lines, 4 words  
TOTAL: 9 lines, 18 words
```