

SE-Assignment-6

Question 1

Python is a widely known programming language recognized for being easy to read and write, plus flexibility.

Their key features that make them popular is:

Flexibility, where you are able to alterate the type of a variable lacking any issues. Variables in Python doesn't need a clear declaration to backup memory space. As it happens automatically once the commands are allocated to a variable. Code example: 'a = 5 enter then a = "Hello World" '.

Interpretation, in order to make it easy to test and debug Python runs code line by line. Code example 'x = 10 enter y = 400 enter print (x + y) '.

Not difficult to read, where Python's syntax is direct and clear. Code example 'print("Hello, world")'.

The ability to have multiple styles, which allows developers to use various programming styles relating to functional or object-oriented.

They have various and many built-in functions and having numerous collective programming tasks like connecting altering files, reading and working with data.

Examples are game development like Pygame that lets you to create multimedia applications and 2D games. By using codes 'import pygame and pygame.init()'

Another use as an example is automation and scripting by writing scripts to automate on going and repetitive tasks.

Question 2

Step 1: Clicked on the link <http://www.python.org>

Step 2: Clicked on “Python 3.12.3”. For Windows

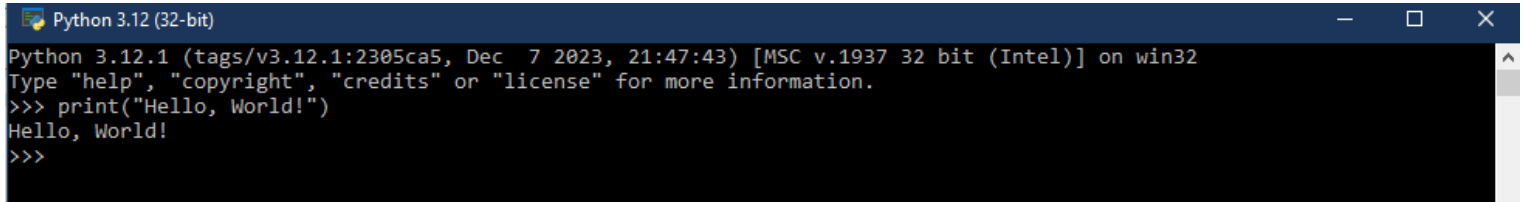
Step 3: Selected “Install” and began installing thereafter finally installed.

Step 4: Making sure that Python is in my path in the environment variable. Now all done!

To verify the installation I'll open my command line prompt and type in 'python --version' therefore will see the version of the installed Python. For my Windows Operating System.

To set up my virtual environment, I'd use my command line prompt and enter in “pip install virtualenv” then to make a virtual environment, therefore I'd go to my project directory then after make a virtual environment which I'd type 'python -m venv venv'. Then after I'll type 'venv\Scripts\activate' for me to get the Virtual environment activated.

Question 3

A screenshot of a Python 3.12 (32-bit) console window. The title bar reads 'Python 3.12 (32-bit)'. The console text shows the Python version and build information: 'Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32'. It then prompts the user to type 'help', 'copyright', 'credits', or 'license' for more information. The user enters '>>> print("Hello, World!")' and the output 'Hello, World!' is displayed. The prompt '>>>' is shown again at the bottom.

```
Python 3.12 (32-bit)
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
Hello, World!
>>>
```

The line `print("Hello, World!")` is the essential part to this program.

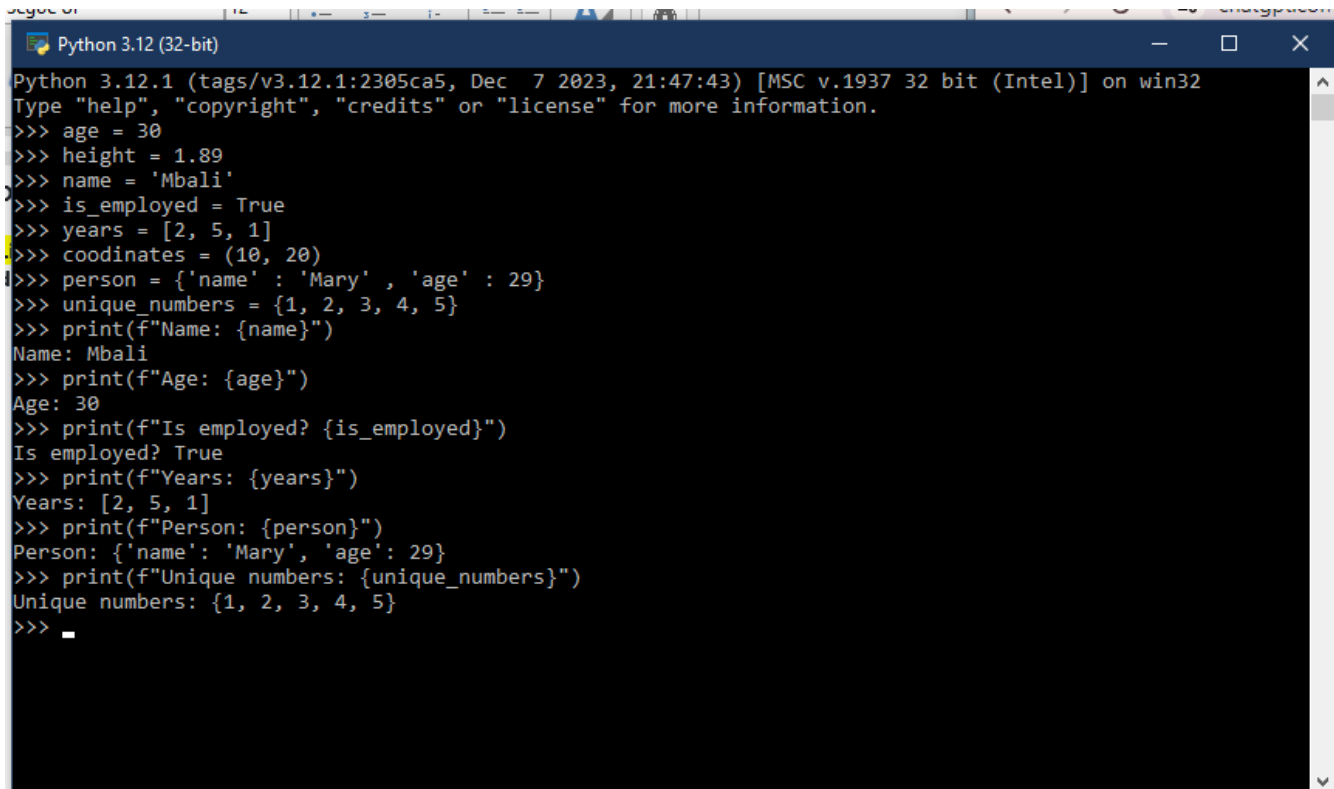
`print()` is important for showing variables or text on the console.

Within `print("Hello, World!")`, that has `"Hello, World!"` is a section of text surrounded in double quotes described as (`"`), which is called a string. The `print()` function shows this text on the screen.

Question 4

- Float which is decimal numbers, typed as “float” in Python.
- List which are changeable or exchangeable list of items.
- Integer are whole numbers like 1, 2, 3, 4 typed as “int” in Python.
- String are just text like “my world” typed as “str” in Python.
- Set which is a collection of unique items like {1, 2, 3, 4}
- Tuple, which are unalterable lists of items.
- Boolean which are True or False and is typed as “bool”.

Here's a short script that I created to demonstrate how to use different data types.

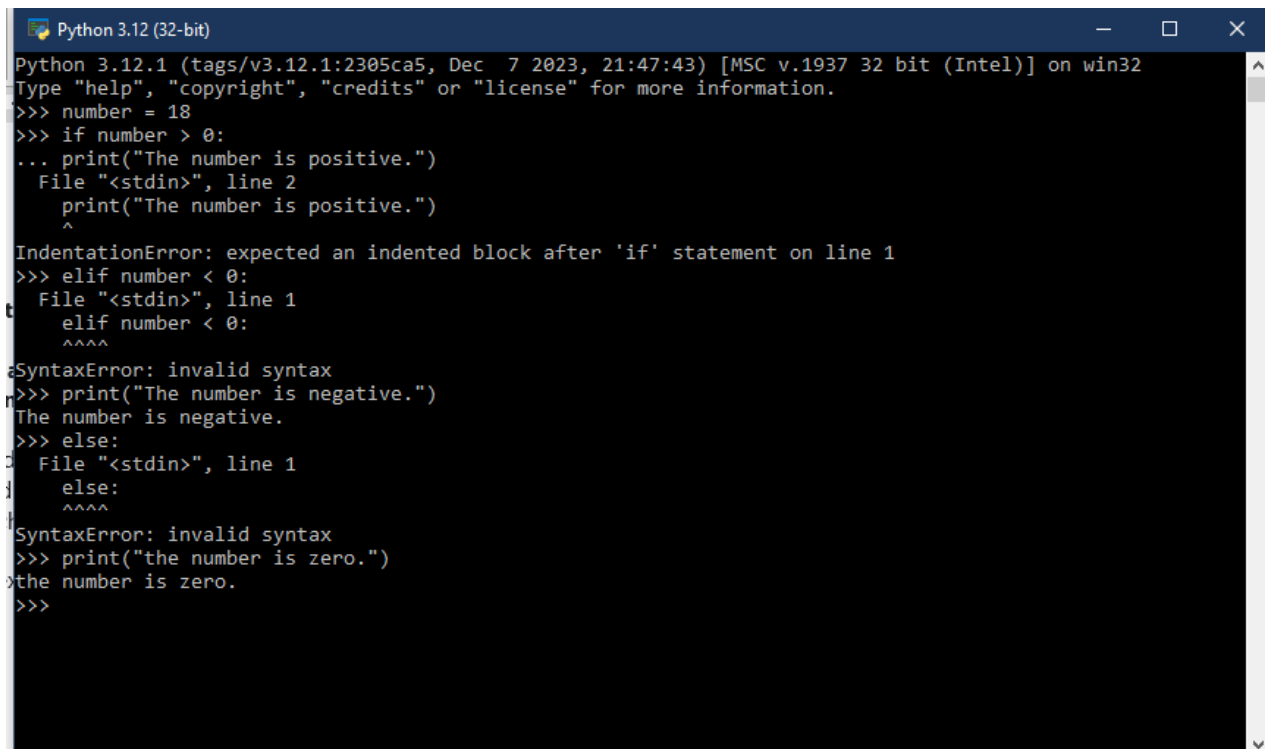


```
Python 3.12 (32-bit)
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> age = 30
>>> height = 1.89
>>> name = 'Mbali'
>>> is_employed = True
>>> years = [2, 5, 1]
>>> coordinates = (10, 20)
>>> person = {'name': 'Mary', 'age': 29}
>>> unique_numbers = {1, 2, 3, 4, 5}
>>> print(f"Name: {name}")
Name: Mbali
>>> print(f"Age: {age}")
Age: 30
>>> print(f"Is employed? {is_employed}")
Is employed? True
>>> print(f"Years: {years}")
Years: [2, 5, 1]
>>> print(f"Person: {person}")
Person: {'name': 'Mary', 'age': 29}
>>> print(f"Unique numbers: {unique_numbers}")
Unique numbers: {1, 2, 3, 4, 5}
>>> _
```

Question 5

Conditional statements allows you run assured quantities of your code simply if precise conditions are met. The simple conditional statement within Python is the “*if*” statement, which are able to be used through the “*elif*” which is else if and “*else*”.

An example the picture below:



```
Python 3.12 (32-bit)
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> number = 18
>>> if number > 0:
... print("The number is positive.")
File "<stdin>", line 2
    print("The number is positive.")
    ^
IndentationError: expected an indented block after 'if' statement on line 1
>>> elif number < 0:
File "<stdin>", line 1
    elif number < 0:
    ^^^^^
SyntaxError: invalid syntax
>>> print("The number is negative.")
The number is negative.
>>> else:
File "<stdin>", line 1
    else:
    ^^^^^
SyntaxError: invalid syntax
>>> print("the number is zero.")
the number is zero.
>>>
```

So if the number is more than 0 then it prints the following “The number is positive.” Then if the number is less than 0 then it prints the following “The number is negative”. If the number is 0 it will then print “The number is zero.”

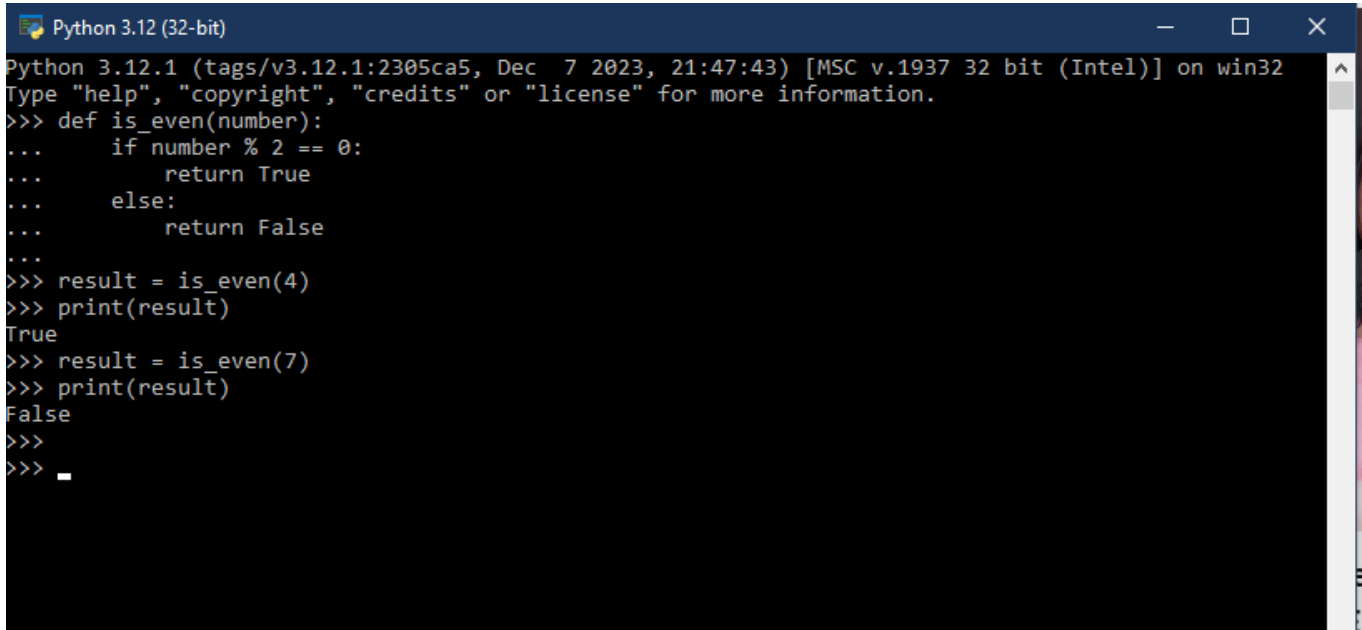
With regards to Loops, it allows you to repeat numerous times block of codes. The main types are while loops and for loops.

```
Python 3.12 (32-bit)
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> fruits = ["grape", "banana", "apple"]
>>>
>>> for fruit in fruits:
...     print(fruit)
...
grape
banana
apple
>>>
```

The “*for*” loop goes over every item in the fruits list and prints them.

Question 6

In Python functions are blocks of codes that can be used multiple times that implement particular tasks. They help you reuse code plus create programs that are much easier to understand like maintaining, reading plus debugging.



```
Python 3.12 (32-bit)
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> def is_even(number):
...     if number % 2 == 0:
...         return True
...     else:
...         return False
...
>>> result = is_even(4)
>>> print(result)
True
>>> result = is_even(7)
>>> print(result)
False
>>>
>>> _
```

The 4 is moved to the *"is_even"* function where it returns the true since 4 is an even number, as 7 is moved into the function. Therefore returns false 7 to a number is odd or odd numbers.

Question 7

The differences:

Dictionaries are made using curly braces {} with key-value pairs alienated by colons :, in terms of syntax while lists are known made using [] which square brackets.

Dictionaries don't continue the order of elements until Python 3.7. As for lists it continues the order of elements.

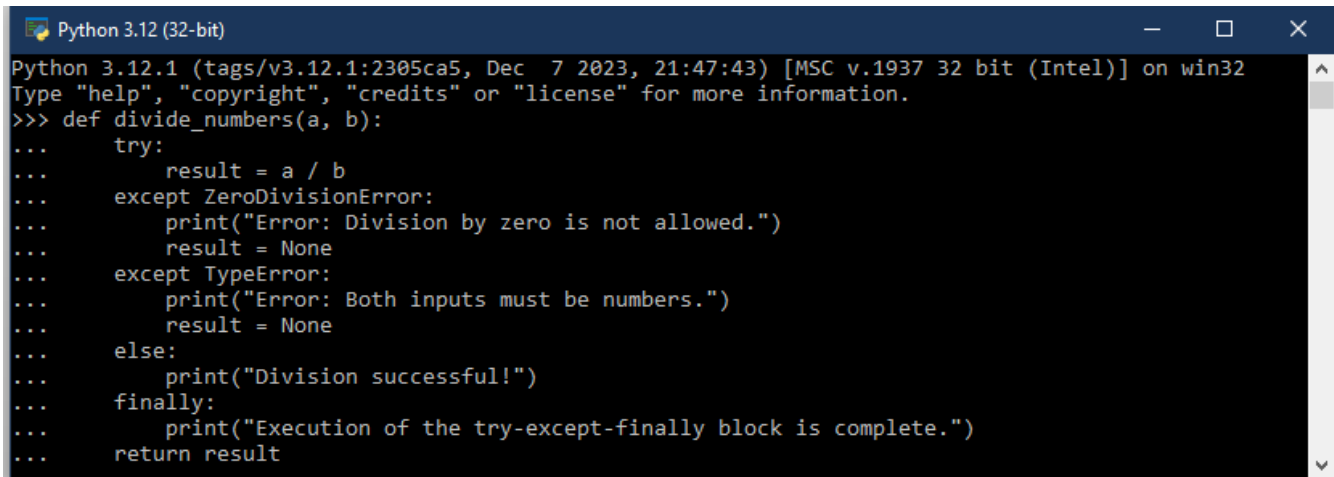
With dictionaries, values and keys are able to be different data types. While Lists keep elements of various data types.

Dictionaries on every element will be a pair containing of key and a value. While in Lists keep elements of various data types.

```
Python 3.12 (32-bit)
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> fruits = ["apple", "banana", "orange"]
>>> person = {"name": "John Doe", "age": 25, "city": "San Francisco", "is_student": True}
>>> print("Fruits:", fruits)
Fruits: ['apple', 'banana', 'orange']
>>> fruits.append("grape")
>>> print("Fruits after adding grape:", fruits)
Fruits after adding grape: ['apple', 'banana', 'orange', 'grape']
>>> fruits.remove("banana")
>>> print("Fruits after removing banana:", fruits)
Fruits after removing banana: ['apple', 'orange', 'grape']
>>> print("\nPerson:", person)
Person: {'name': 'John Doe', 'age': 25, 'city': 'San Francisco', 'is_student': True}
>>> print("Name:", person["name"])
Name: John Doe
>>> person["occupation"] = "Engineer"
>>> print("Person after adding occupation:", person)
Person after adding occupation: {'name': 'John Doe', 'age': 25, 'city': 'San Francisco', 'is_student': True, 'occupation': 'Engineer'}
>>> del person["is_student"]
>>> print("Person after removing is_student:", person)
Person after removing is_student: {'name': 'John Doe', 'age': 25, 'city': 'San Francisco', 'occupation': 'Engineer'}
>>>
```


Question 8

Exception handling in Python aids to accomplish errors that happen throughout program execution, avoiding crashes when an unplanned issues arise.



```
Python 3.12 (32-bit)
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> def divide_numbers(a, b):
...     try:
...         result = a / b
...     except ZeroDivisionError:
...         print("Error: Division by zero is not allowed.")
...         result = None
...     except TypeError:
...         print("Error: Both inputs must be numbers.")
...         result = None
...     else:
...         print("Division successful!")
...     finally:
...         print("Execution of the try-except-finally block is complete.")
...     return result
```

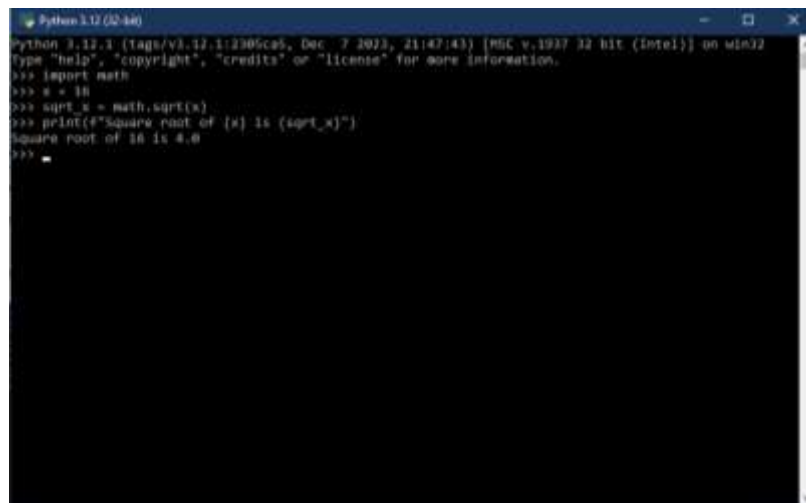
Question 9

With packages, it aids with the organization of codes into a structure. It's a group of modules grouped together in a directory. It has "`__init__.py`". And the module to keeps the variables and functions you can use again. Plus contains files in Python code.

For me to import and use modules:

I'll use the "import" keyword that is led by the module name. Then after to get the function I'll use "module_name,function_name"

An example of math module.



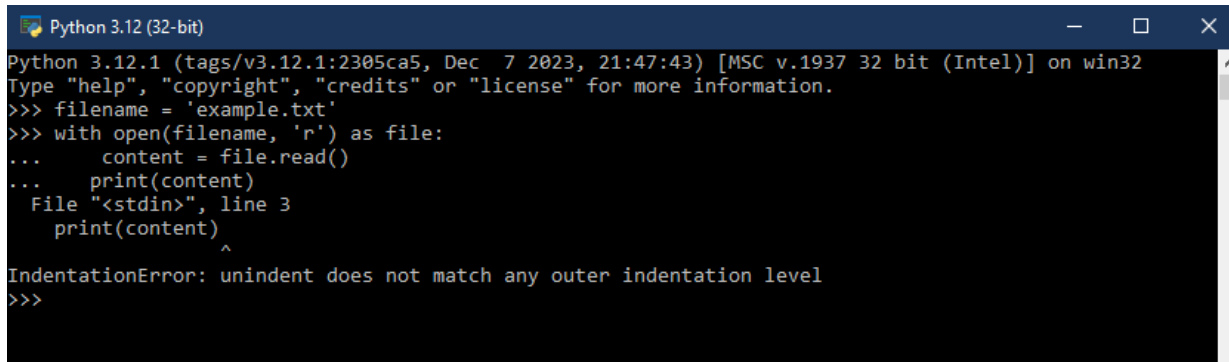
```
Python 3.12 (tags/v3.12.1:1305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more (information).
>>> import math
>>> x = 16
>>> sqrt_x = math.sqrt(x)
>>> print(f"Square root of {x} is {sqrt_x}")
Square root of 16 is 4.0
>>>
```

Question 10

For writing a file in python are when you open a file which is "w", that is write mode. And Write individual line from the list to the file, monitored by a newline character ('\n').

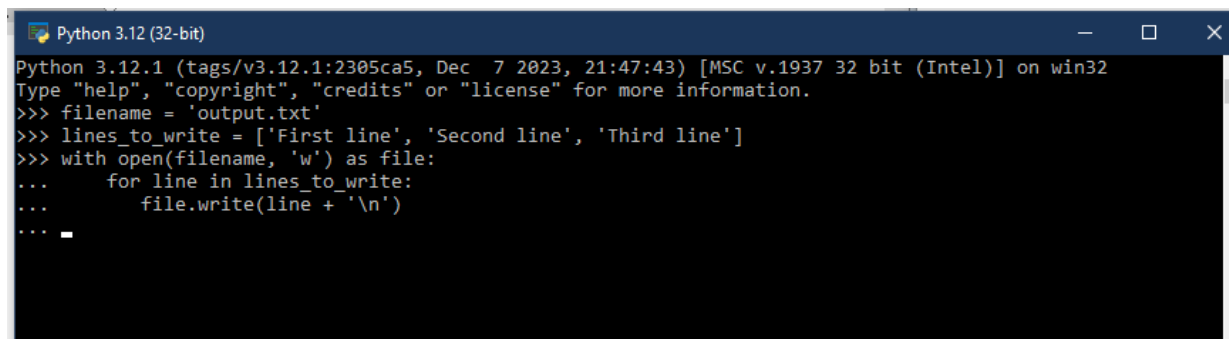
And reading from a file I'll open the file in read mode ('r'), then after read the content, thereafter print the content.

Example of read:

A screenshot of a Python 3.12 (32-bit) terminal window. The window title is "Python 3.12 (32-bit)". The terminal shows the following code:

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> filename = 'example.txt'
>>> with open(filename, 'r') as file:
...     content = file.read()
...     print(content)
...     print(content)
File "<stdin>", line 3
    print(content)
    ^
IndentationError: unindent does not match any outer indentation level
>>>
```

Example of writing:

A screenshot of a Python 3.12 (32-bit) terminal window. The window title is "Python 3.12 (32-bit)". The terminal shows the following code:

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec 7 2023, 21:47:43) [MSC v.1937 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> filename = 'output.txt'
>>> lines_to_write = ['First line', 'Second line', 'Third line']
>>> with open(filename, 'w') as file:
...     for line in lines_to_write:
...         file.write(line + '\n')
...     _
```

Citation

- Research, A. (2023) 'What is Python and its Many Use Cases,' Azumo, 9 December. <https://azumo.com/insights/what-is-python-and-its-many-use-cases#:~:text=Python%20is%20an%20object%2Doriented,easier%20to%20understand%20and%20maintain.>
- UNStOP - competitions, quizzes, hackathons, scholarships and internships for students and corporates (nd). <https://unstop.com/blog/features-of-python.>
- GeeksforGeeks (2023) How to install Python on Windows? <https://www.geeksforgeeks.org/how-to-install-python-on-windows/>.
- Mandaokar, S. (2023) 'Python: Conditional Statements and Loops - TheTechieGuys - Medium,' Medium, 28 April. <https://medium.com/thetechieguys/python-conditional-statements-and-loops-ad4631778b01.>
- Biswal, A. (2024) Understanding Python If-Else Statement. <https://www.simplilearn.com/tutorials/python-tutorial/python-if-else-statement.>
- Python Modules (With Examples) (nd). <https://www.programiz.com/python-programming/modules.>