PYTHON

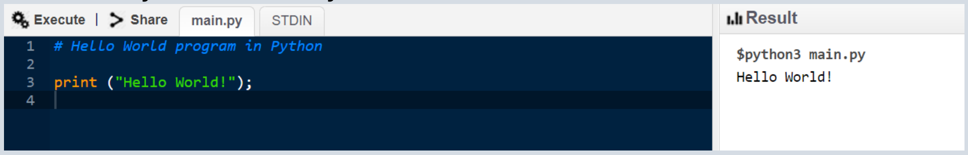
1. What is Python & history
2. Install Python and install an IDE such as Pycharm
3. Print your first code (hello world)
4. What are Variables and show some simple examples in your IDE
5. How do you swap variables + examples
6. Variable types
7. Variable number arithmetic operators

Answers:

1. Guido Van Rossum published the first version of Python code (version 0.9.0) at all sources in February 1991. This release already included exception handling, functions, and the core data types of lists, dict, strand others. It was also object-oriented and had a module system. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

It is used for:

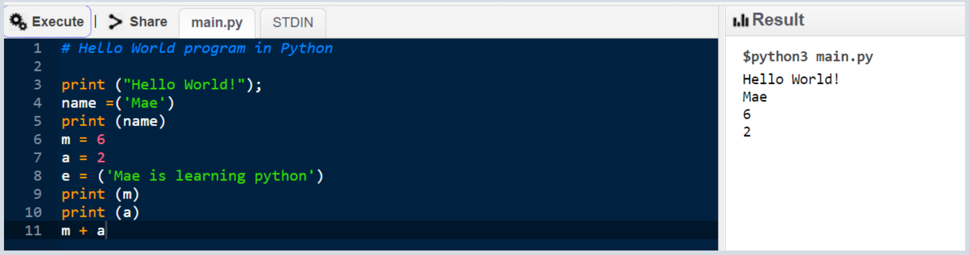
- web development (server-side),
- software development,
- mathematics,
- system scripting.

2. Installed Pycharm already
3. 

```
# Hello World program in Python

print ("Hello World!");
```

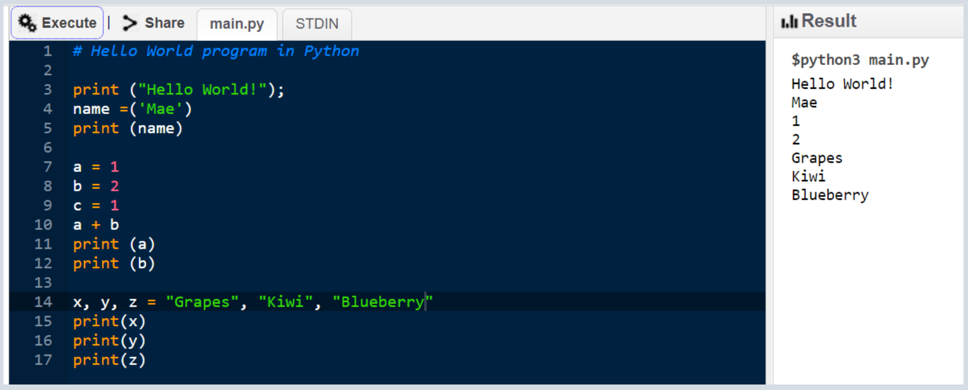
```
$python3 main.py
Hello World!
```

4. **Variables** - A variable can be seen as a container (or some say a pigeonhole) to store certain values. While the program is running, variables are accessed and sometimes changed, i.e., a new value will be assigned to a variable.



```
# Hello World program in Python

print ("Hello World!");
name =('Mae')
print (name)
m = 6
a = 2
e = ('Mae is learning python')
print (m)
print (a)
m + a
```

```
$python3 main.py
Hello World!
Mae
6
2
```

5. 

```
# Hello World program in Python

print ("Hello World!");
name =('Mae')
print (name)

a = 1
b = 2
c = 1
a + b
print (a)
print (b)

x, y, z = "Grapes", "Kiwi", "Blueberry"
print(x)
print(y)
print(z)
```

```
$python3 main.py
Hello World!
Mae
1
2
Grapes
Kiwi
Blueberry
```

Swapping variables

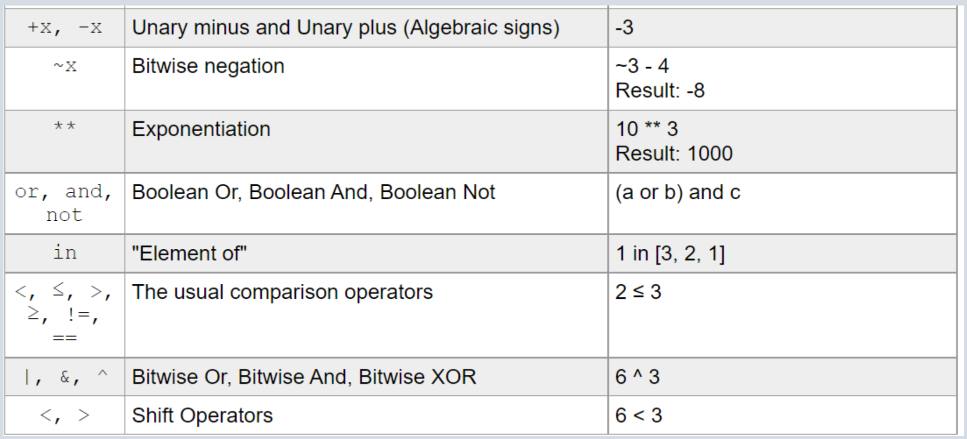
6. ***Variable types*** - The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types −

- Numbers - Number data types store numeric values. Number objects are created when you assign a value to them.
- String - Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes.
- List - Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]).
- Tuple - A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.
- Dictionary - Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object.
- Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

7. Arithmetic operators These operators can be applied to all numeric types

+, -	Addition, Subtraction
*, %	Multiplication, Modulo
/	Division
//	Truncation Division



`+x, -x`	Unary minus and Unary plus (Algebraic signs)	-3
`~x`	Bitwise negation	~3 - 4 Result: -8
`**`	Exponentiation	10 ** 3 Result: 1000
`or, and, not`	Boolean Or, Boolean And, Boolean Not	(a or b) and c
`in`	"Element of"	1 in [3, 2, 1]
`<, ≤, >, ≥, !=, ==`	The usual comparison operators	2 ≤ 3
`|, &, ^`	Bitwise Or, Bitwise And, Bitwise XOR	6 ^ 3
`<, >`	Shift Operators	6 < 3

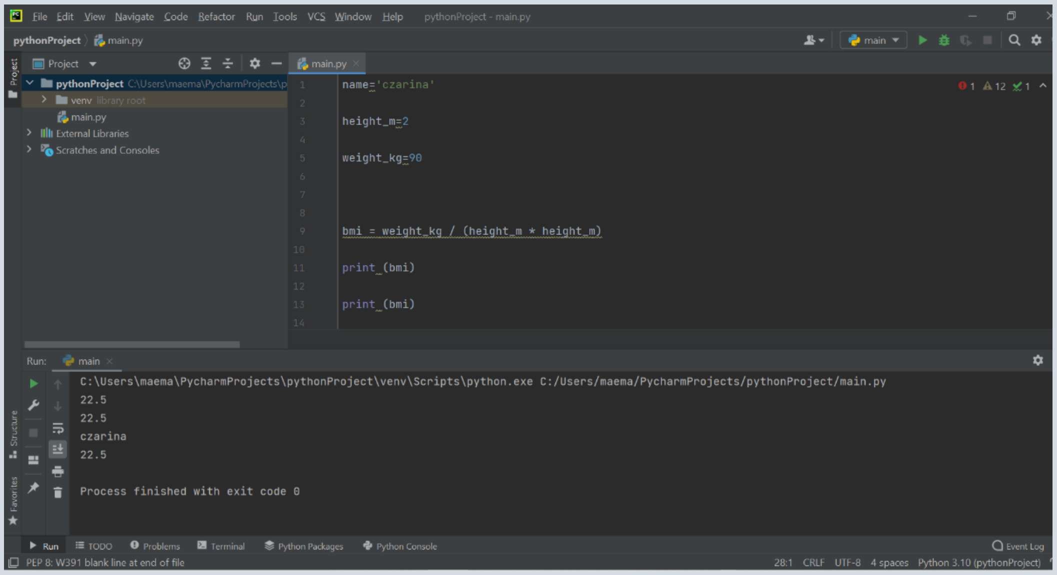
Part 2

8. if clause
9. else
10. equal to (==)
11. BMI calculator
12. Functions
13. BMI Calculator (using functions)

Answers:

8. ***If clause*** – an if statement is written with ***if*** keyword
9. ***Else*** – the else keyword can catch the preceding conditions that isn't caught
10. ***Equal to – a ==b command***
11. ***BMI***

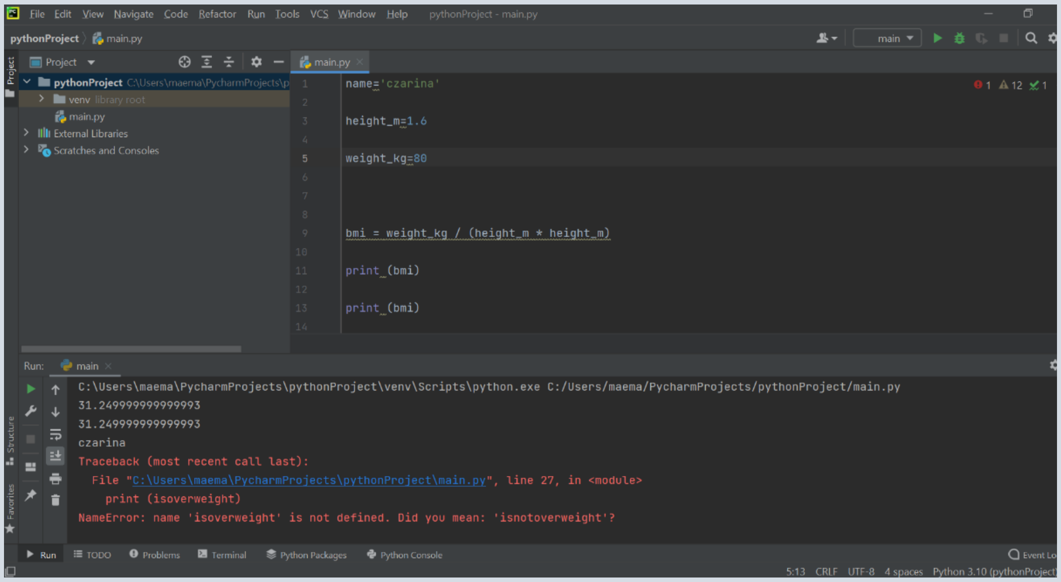
PYTHON

1. 
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonProject - main.py
pythonProject main.py
Project
pythonProject C:\Users\maema\PycharmProjects\p
venv library root
main.py
External Libraries
Scratches and Consoles
Run: main
Run TODO Problems Terminal Python Packages Python Console Event Log

```python
name='czarina'

height_m=2

weight_kg=90



bmi = weight_kg / (height_m * height_m)

print (bmi)

print (bmi)
```

```
C:\Users\maema\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/maema/PycharmProjects/pythonProject/main.py
22.5
22.5
czarina
22.5

Process finished with exit code 0
```

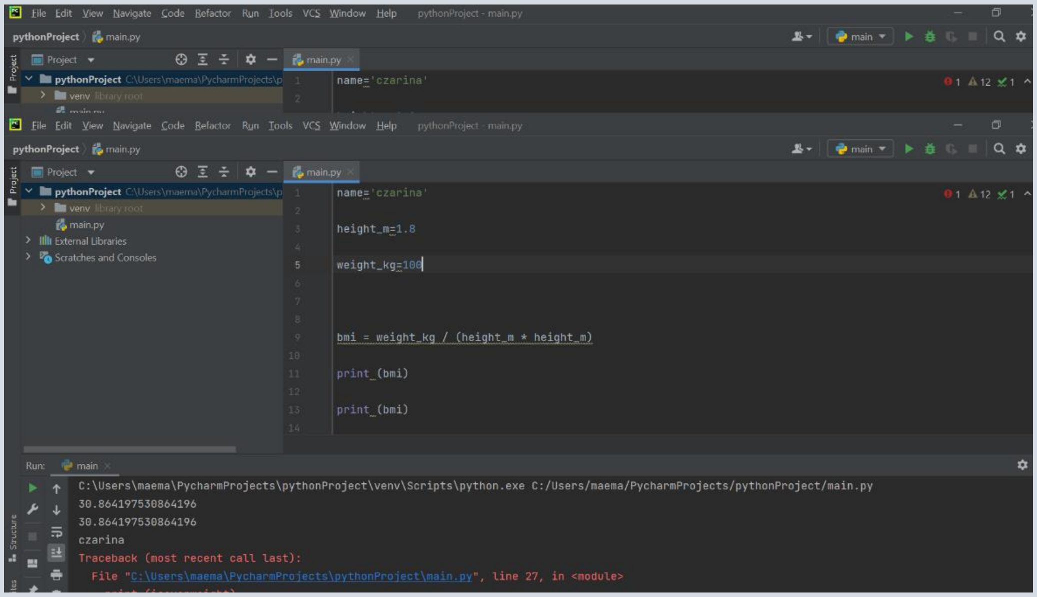
```
PEP 8: W391 blank line at end of file
28:1  CRLF  UTF-8  4 spaces  Python 3.10 (pythonProject)
```

2. 
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonProject - main.py
pythonProject main.py
Project
pythonProject C:\Users\maema\PycharmProjects\p
venv library root
main.py
External Libraries
Scratches and Consoles
Run: main
Run TODO Problems Terminal Python Packages Python Console Event Lo

```python
name='czarina'

height_m=1.6

weight_kg=80



bmi = weight_kg / (height_m * height_m)

print (bmi)

print (bmi)
```

```
C:\Users\maema\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/maema/PycharmProjects/pythonProject/main.py
31.249999999999993
31.249999999999993
czarina
Traceback (most recent call last):
  File "C:\Users\maema\PycharmProjects\pythonProject\main.py", line 27, in <module>
    print (isoverweight)
NameError: name 'isoverweight' is not defined. Did you mean: 'isnotoverweight'?
```

```
5:13  CRLF  UTF-8  4 spaces  Python 3.10 (pythonProject)
```

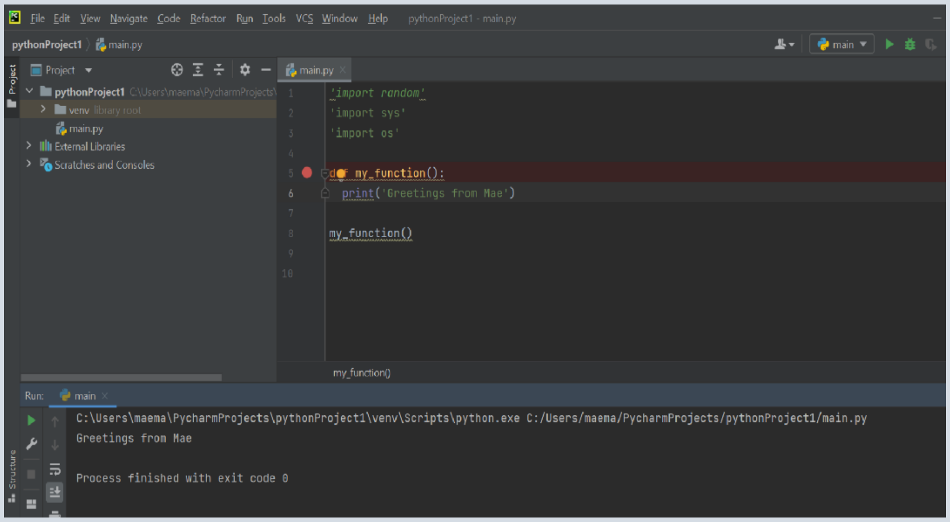
3. 

```
name='czarina'

height_m=1.8

weight_kg=100



bmi = weight_kg / (height_m * height_m)

print (bmi)

print (bmi)
```

```
C:\Users\maema\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/maema/PycharmProjects/pythonProject/main.py
30.864197530864196
30.864197530864196
czarina
Traceback (most recent call last):
  File "C:\Users\maema\PycharmProjects\pythonProject\main.py", line 27, in <module>
```

12. ***Functions –*** A function is a block of codes, which only runs when it is called. In python a function is defined by using ***def*** keyword.

Example:



```
'import random'
'import sys'
'import os'

def my_function():
  print('Greetings from Mae')

my_function()
```

```
C:\Users\maema\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/maema/PycharmProjects/pythonProject1/main.py
Greetings from Mae

Process finished with exit code 0
```

Part 3

Loops

1. *While – with the* <u>*while*</u> *loop as long the conditions are TRUE set of statements can be executed.*

2. For – with the for loop it executes a sequence of statements MULTIPLE TIMES and abbreviates the code of the loop variable.

3. Nested – it can use one or more loop ANY ANOTHER WHILE LOOP OR FOR LOOP

Functions

*1. Creating a function – it begins with the word **def** followed by the function name and parentheses (())*

*2. Print(me) function – **def** printme (str)*

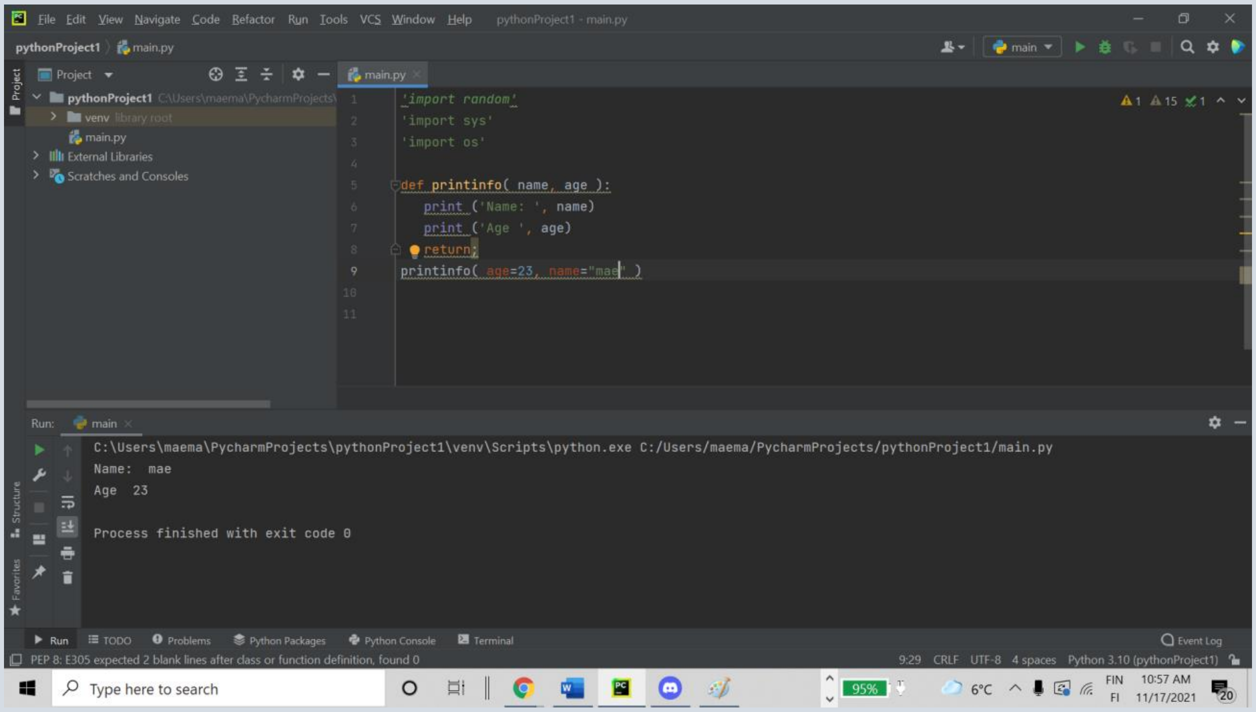
"This python will prints as interesting"

Print str

Return;

3. Keyword arguments functions – keyword arguments are related to the function calls. When using the keyword functions it identifies the argument by its parameter name. It allows to skip the arguments.

Example:



File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonProject1 - main.py
pythonProject1 main.py
Project
pythonProject1 C:\Users\maema\PycharmProjects\
venv library root
main.py
External Libraries
Scratches and Consoles
main
Run: main
Run TODO Problems Python Packages Python Console Terminal
Event Log
9:29 CRLF UTF-8 4 spaces Python 3.10 (pythonProject1)
Type here to search
95%
6°C
FIN FI
10:57 AM 11/17/2021

```python
'import random'
'import sys'
'import os'

def printinfo( name, age ):
    print ('Name: ', name)
    print ('Age ', age)
    return;
printinfo( age=23, name="mae" )
```

```
C:\Users\maema\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/maema/PycharmProjects/pythonProject1/main.py
Name:  mae
Age  23

Process finished with exit code 0
```

PEP 8: E305 expected 2 blank lines after class or function definition, found 0

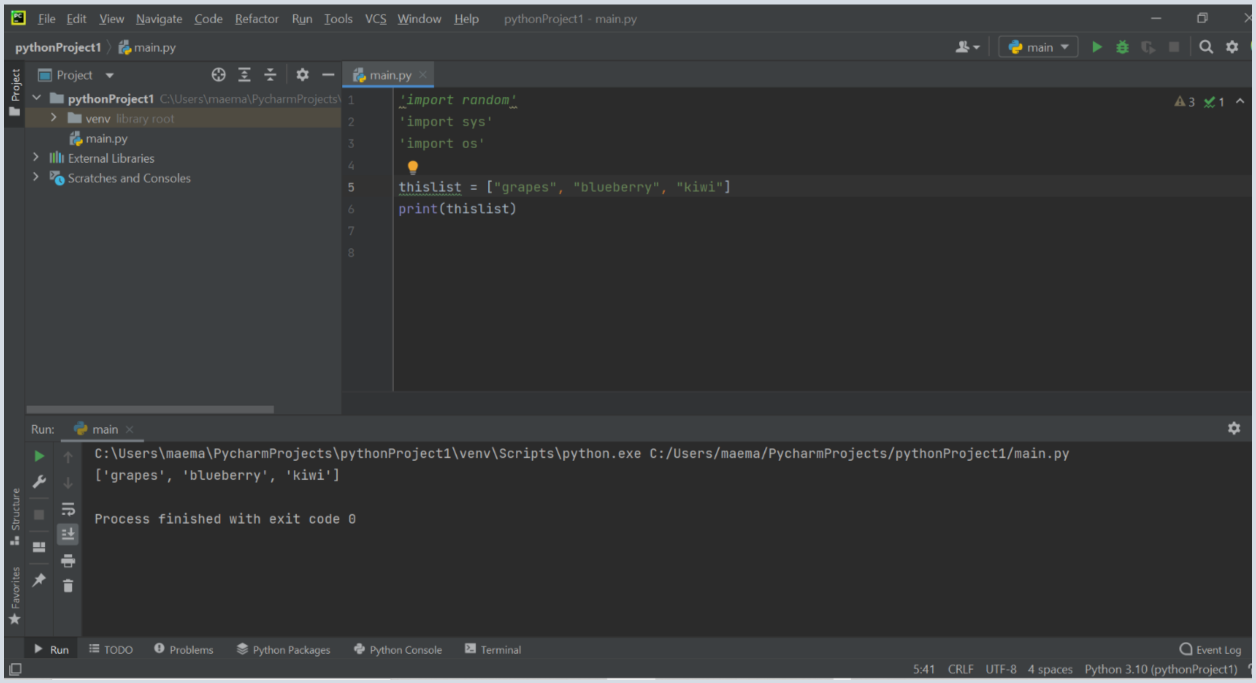
Lists

1. Creating lists

2. Accessing values in a list

3. Basic List operators

1. *List in python is the most versatile datatype available. It can be written comma-separated values between square brackets.*



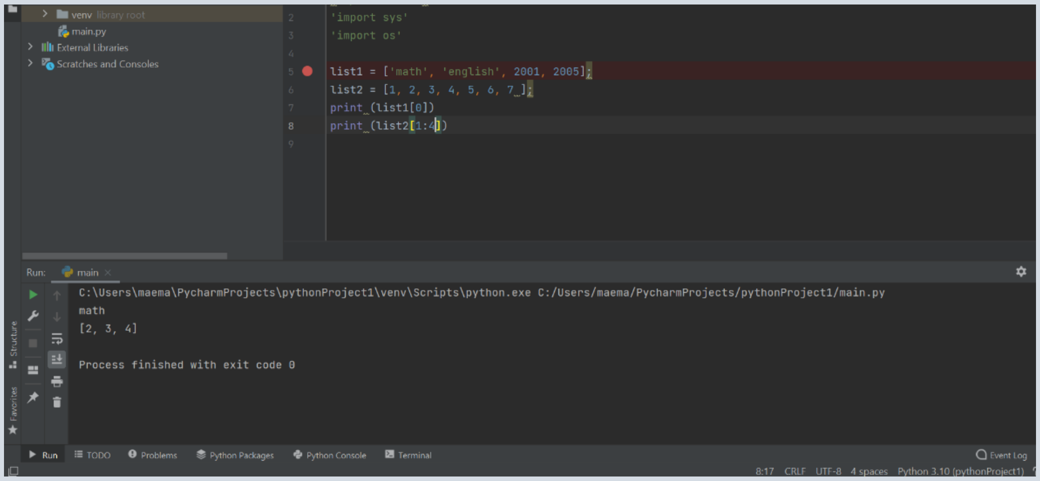
```
'import random'
'import sys'
'import os'

thislist = ["grapes", "blueberry", "kiwi"]
print(thislist)
```

```
C:\Users\maema\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/maema/PycharmProjects/pythonProject1/main.py
['grapes', 'blueberry', 'kiwi']

Process finished with exit code 0
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonProject1 - main.py
pythonProject1 main.py
Project
pythonProject1 C:\Users\maema\PycharmProjects\
venv library root
main.py
External Libraries
Scratches and Consoles
Run: main
Run TODO Problems Python Packages Python Console Terminal Event Log
5:41 CRLF UTF-8 4 spaces Python 3.10 (pythonProject1)

2. To access values in a lists, use square brackets for separating along the index or indices to obtain the value.

Example:



```
'import sys'
'import os'

list1 = ['math', 'english', 2001, 2005];
list2 = [1, 2, 3, 4, 5, 6, 7 ];
print (list1[0])
print (list2[1:4])
```

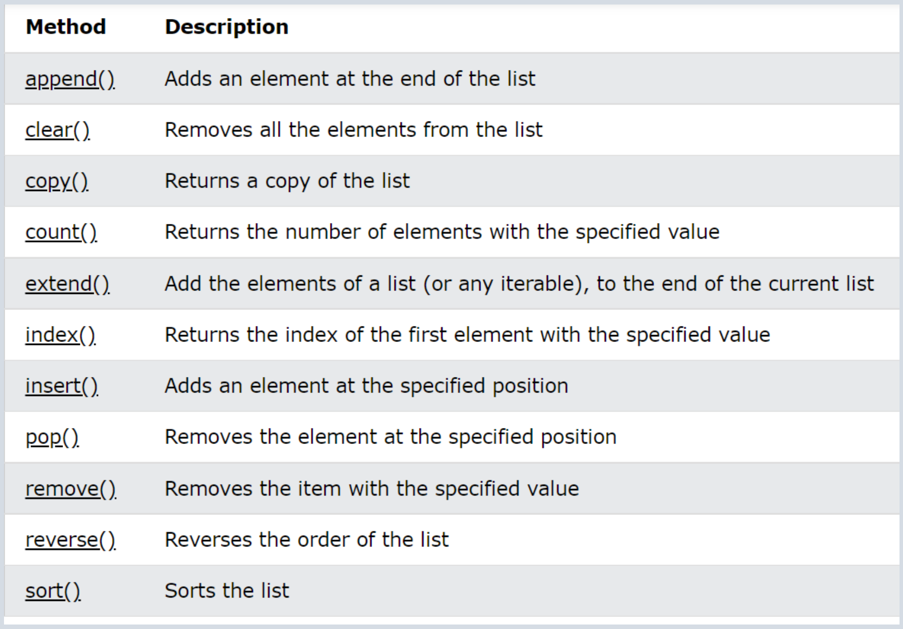
```
C:\Users\maema\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/maema/PycharmProjects/pythonProject1/main.py
math
[2, 3, 4]

Process finished with exit code 0
```

3. *Python basic lists operators*

Lists responds to the + and * operators

Python Expression	**Results**	**Description**
Len ([1, 2, 3,4])	4	Length
4 in [1,2,3,4]	TRUE	Membership
For y in [2,3,4]: print y,	2 3 4	Iteration
[‘Hello’ * 5	[‘Hello’, ‘Hello’, ‘Hello’, ‘Hello’,’Hello’]	Repetition
[4,5,6], + [7,8,9]	[4,5,6,7,8,9]	Concatenation

- Access lists items
- Change lists items
- Add lists items
- Remove lists items
- Loop lists
- List comprehensions
- Join lists
- **List methods**



Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

Sources:

URL: https://www.w3schools.com/python/python_lists_change.asp Accessed: 17.11.2021

URL: https://www.tutorialspoint.com/python/python_lists.htm Accessed: 17.11.2021