Booting on Python

Programming Languages

There are Two types of Programming Languages on the basis of Converting Code into Machine language.

1. Compiled Languages
2. Interpreted Languages

Programming Languages

Programming Languages are also classified in the following two types

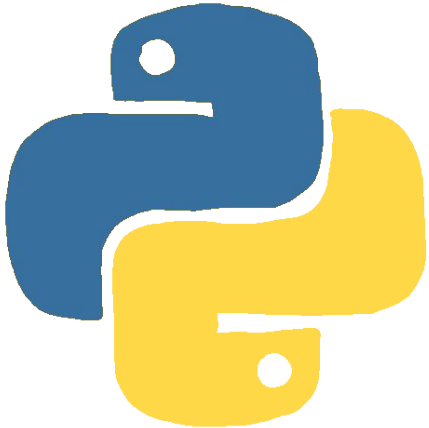
1. In which we have to declare the variables (Statically typed)

2. In which we do not have to declare the variables (Dynamically typed)

Python

Python is

1. Interpreted Language.
2. Dynamically Typed Language.



Compiler

High Level Language

```
#include <iostream>
using namespace std;

int main()
{
  int a = 5, b = 8;
  int result;
  result = a + b;
  cout << result;
  return 0;
}
```

Machine Language

1110 1101 1100

1000 0111 0110

1011 0001 1001

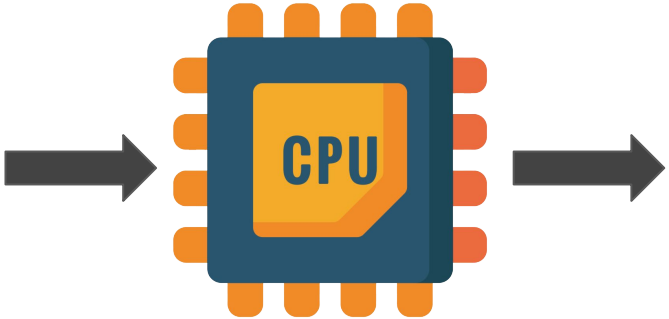
0110 1101 0100

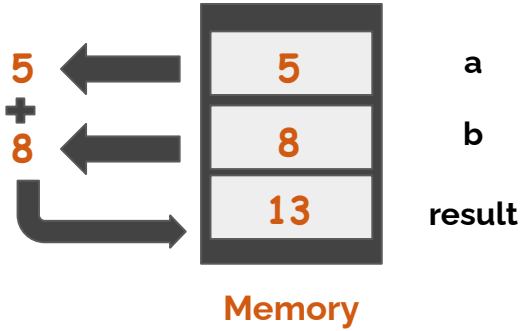
0010 1001 1101

0111 0000 0100

How Compiler Works

Machine Language
1110 1101 1100
1000 0111 0110
1011 0001 1001
0110 1101 0100
0010 1001 1101
0111 0000 0100

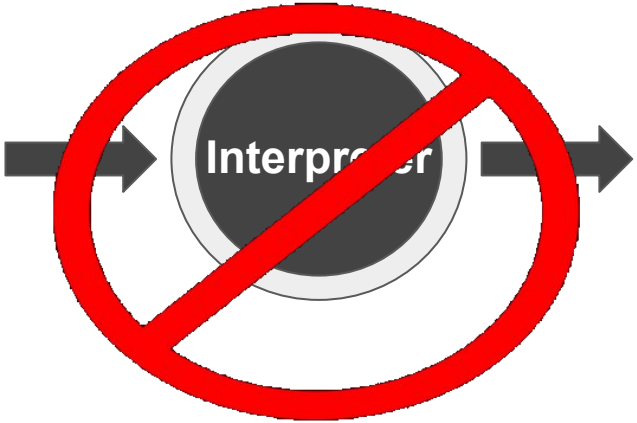




What is Interpreter

High Level Language

```
#include <iostream>
using namespace std;

int main()
{
  int a = 5, b = 8;
  int result;
  result = a + b;
  cout << result;
  return 0;
}
```



Machine Language

1110 1101 1100

1000 0111 0110

1011 0001 1001

0110 1101 0100

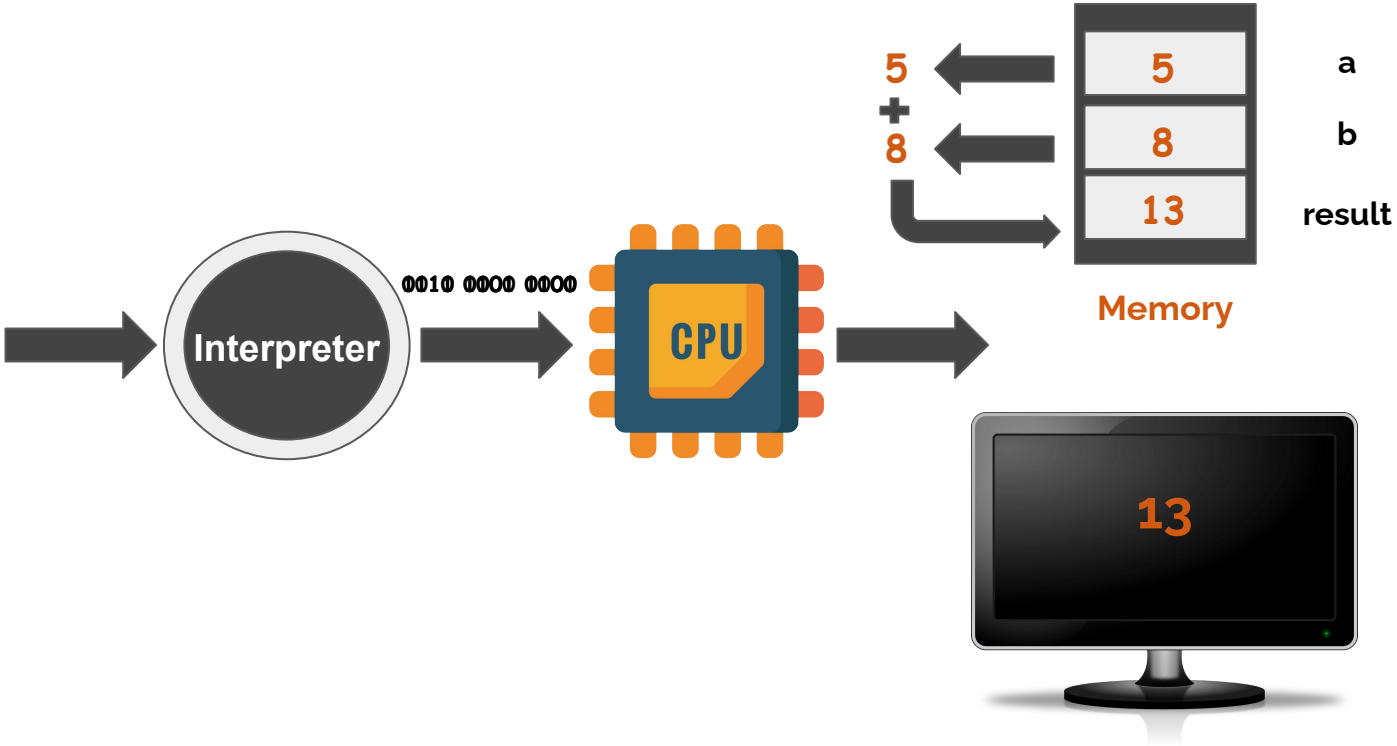
0010 1001 1101

0111 0000 0100

What is Interpreter

High Level Language

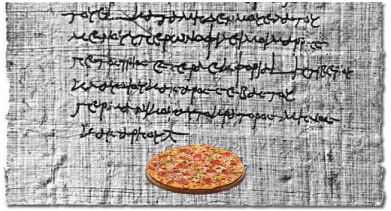
```
#include <iostream>
using namespace std;

int main()
{
  int a = 5, b = 8;
  int result;
  result = a + b;
  cout << result;
  return 0;
}
```



Interpreter: Real World Example

1

Compiler



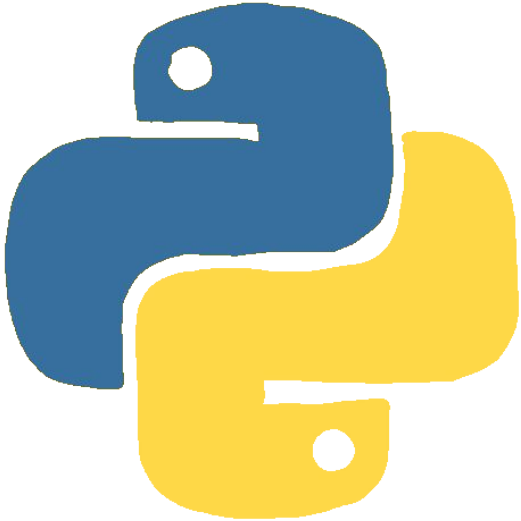
2

Interpreter

Compiled Version

Python: Interpreted Language

Python is an Interpreted Language



Python: Dynamically Typed

For variable declaration, we have seen, these two things are required for memory reservation.

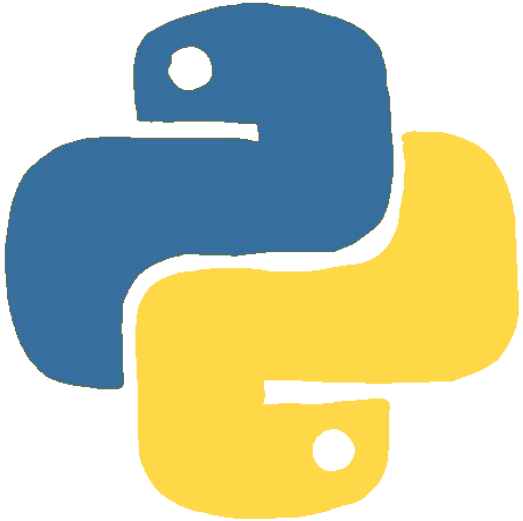
1. Name of the Variable.
2. ~~Type of Data.~~

nameOfTheVariable

Datatype

What is Python?

Python is an interpreted, dynamically typed, high-level and general-purpose programming language.



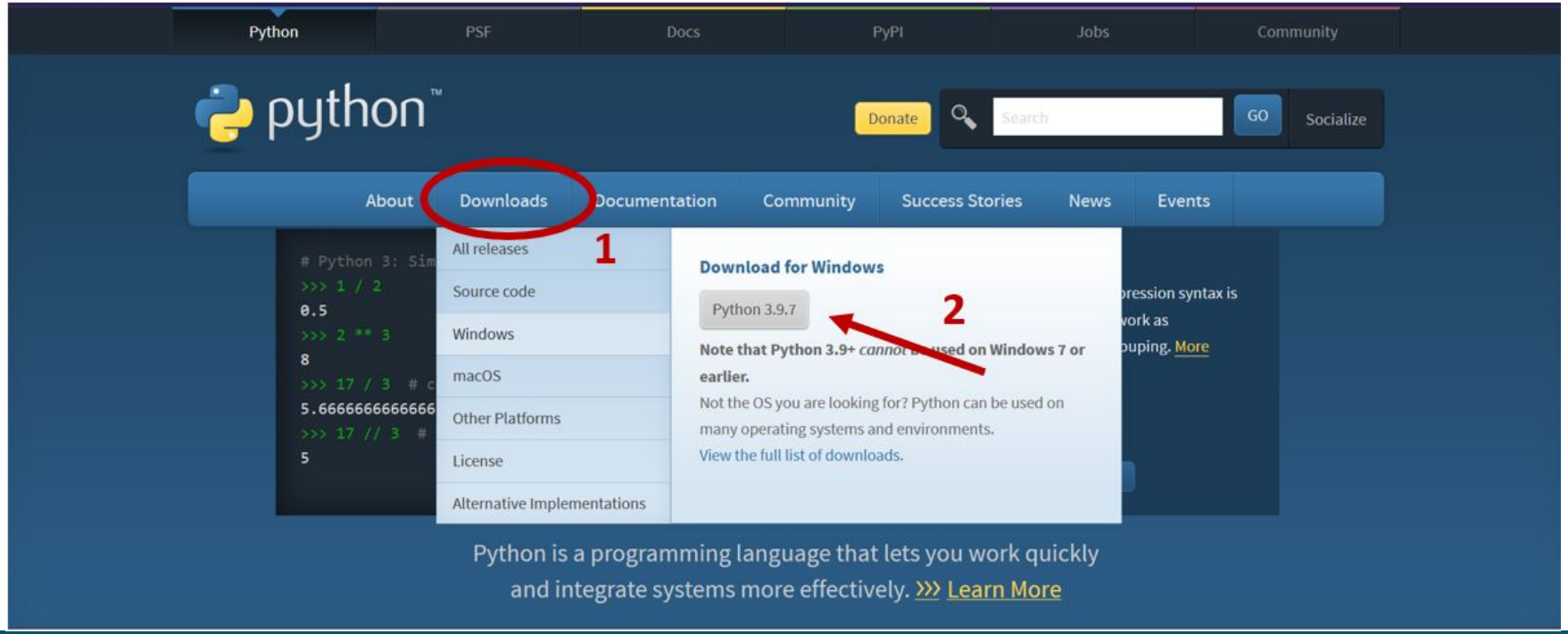
Why Python

Rank	Language	Type	Score
1	Python		100.0
2	Java		95.3
3	C		94.6
4	C++		87.0
5	JavaScript		79.5
6	R		78.6
7	Arduino		73.2
8	Go		73.1
9	Swift		70.5
10	Matlab		68.4
11	Ruby		66.8

Python Installation

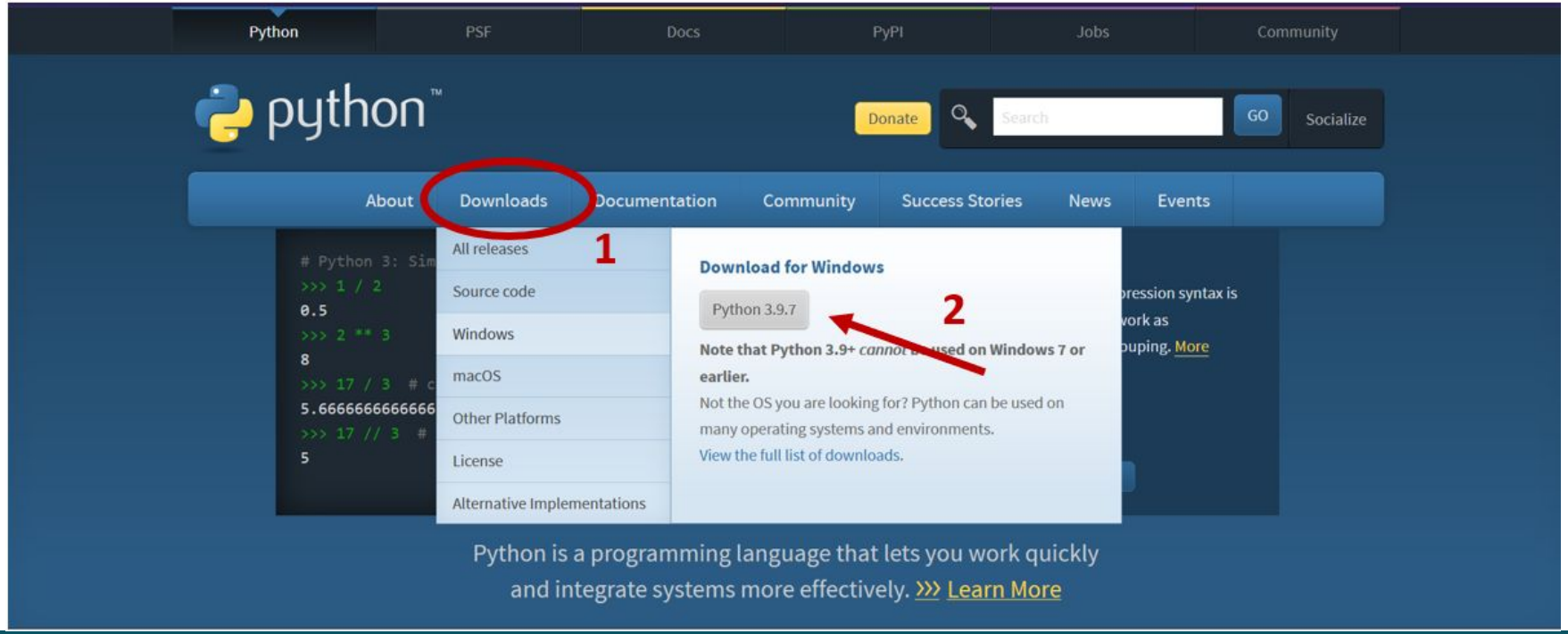
Python Installation

Go to the website: https://www.python.org/



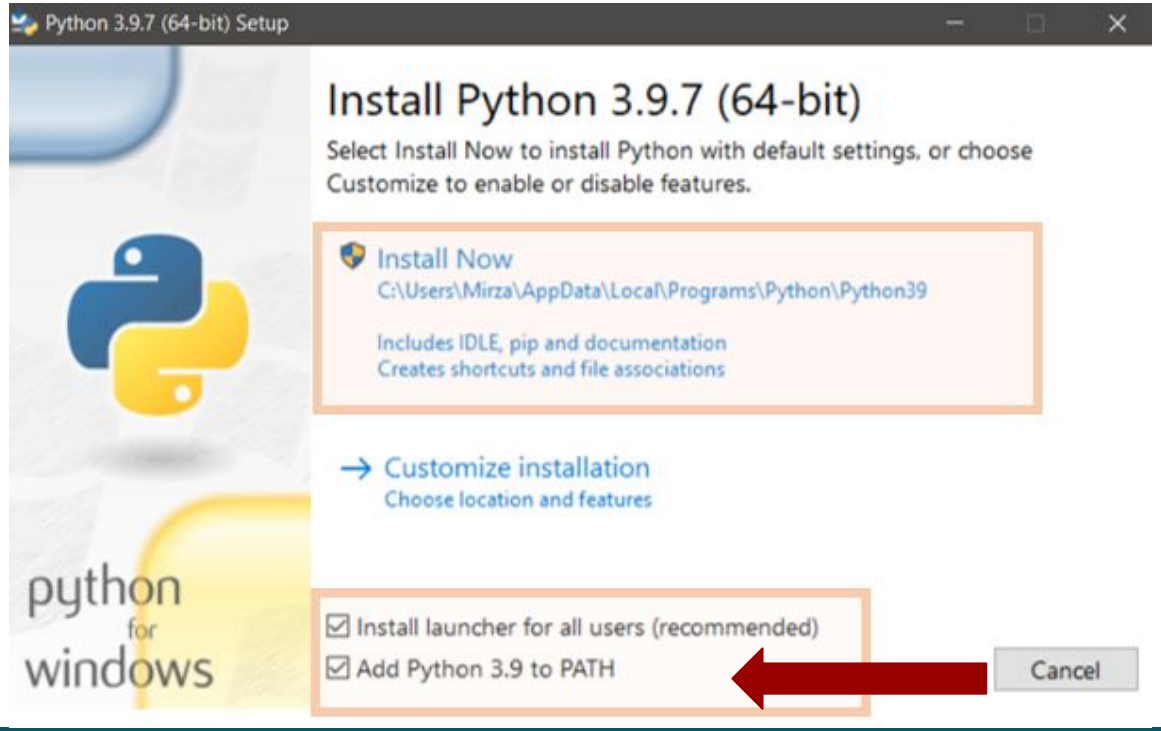
Python Installation

Click on Python 3 and download 'Windows installer (64-bit)'



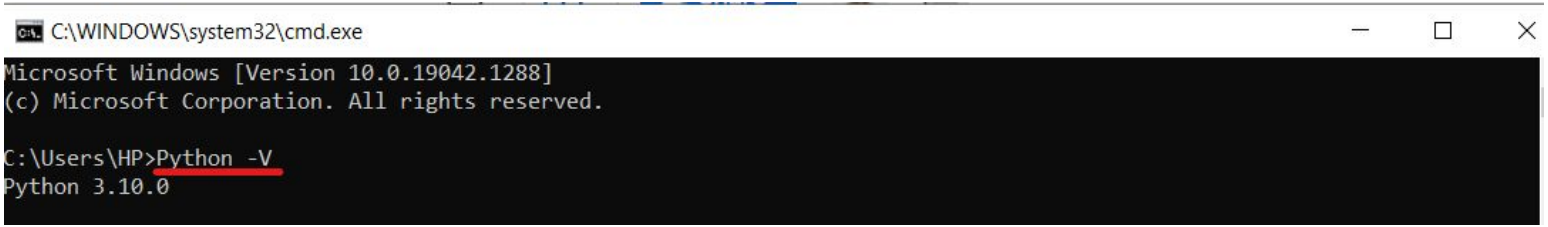
Python Installation

Before clicking the Install Now, make sure that you have added Python Path



Python Installation

To verify that you have successfully installed Python, go to the command prompt and type Python -v



Open a .py File

Make a new text file, change its extension to .py and open it in Visual Studio Code.

Displaying Output in Python

Displaying Output on Console

In C++, we wrote cout(" ") and in C# we wrote Console.WriteLine(" ").
In Python, we write print(" ")

```
print("Hello to Python")
```

Displaying Output on Console

Python does not require semicolons to terminate statements. Although, Semicolons can be used to delimit statements if you wish to put multiple statements on the same line.

```
print("Hello to Python")
```

Displaying Output on Console

Python is an Interpreted language. It means that it's output is generated line by line without any compilation using an interpreter instead of a compiler.

C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

G:\OOP 2022\Week 14>FirstPythonCode.py
Hello to Python

G:\OOP 2022\Week 14>
```

Displaying Output on Console

We don't have to tell explicitly to include new line code. print statement prints everything on the new line.

```
print("Hello to Python")
print("First Program in Python")
```

```
G:\OOP 2022\Week 14>FirstPythonCode.py
Hello to Python
First Program in Python

G:\OOP 2022\Week 14>
```

Declaring Variables in Python

Declaring Variables

Python is a dynamically typed language, therefore we do not need to tell the dataType explicitly.

```
number = 5
floatNumber = 5.5
character = 'T'
word = "Hello"
print(number, " ", floatNumber, " ", character, " ", word, " ")
```

```
G:\OOP 2022\Week 14>FirstPythonCode.py
5   5.5   T   Hello
```

Declaring Variables

When we Assign a value to any variable, Python automatically associates a Data Type with it.

```
number = 5
floatNumber = 5.5
character = 'T'
word = "Hello"
print(number, " ", floatNumber, " ", character, " ", word, " ")
```

```
G:\OOP 2022\Week 14>FirstPythonCode.py
5   5.5   T   Hello
```

Declaring Variables

We can Check the Data Type using the type command.

```
number = 5
floatNumber = 5.5
character = 'T'
word = "Hello"
print(type(number), " ", type(floatNumber), " ", type(character), " ", type(word))
```

```
G:\OOP 2022\Week 14>FirstPythonCode.py
<class 'int'>   <class 'float'>   <class 'str'>   <class 'str'>
```

Declaring Variables

Python does not have a character or char type. All single characters are strings with length one.

```
number = 5
floatNumber = 5.5
character = 'T'
word = "Hello"
print(type(number), " ", type(floatNumber), " ", type(character), " ", type(word))
```

```
G:\OOP 2022\Week 14>FirstPythonCode.py
<class 'int'>   <class 'float'>   <class 'str'>   <class 'str'>
```

Taking Input in Python

Taking Input

In C++, we wrote cin and in C# we wrote Console.ReadLine().
In Python, we take Input with input command

```
N1 = input("Enter First Number:")
```

Taking Input

Just like in C#, the input is always in a string. Therefore, we have to type cast the string into integer if we are taking numbers as input.

```
N1 = input("Enter First Number:")
N1 = int(N1)
```

Taking Input

Just like in C#, the input is always in a string. Therefore, we have to type cast the string into float if we are taking decimal numbers as input.

```
N1 = input("Enter First Number:")
N1 = float(N1)
```

Working Example: Vision

Write a program that takes length of side of a square as input and calculate its area using following formula:

Area= Length * Length

```
G:\OOP 2022\Week 14>FirstPythonCode.py
Enter the Length:5
Area is 25.0
```

Working Example: Vision

Write a program that takes length of side of a square as input and calculate its area using following formula:

Area= Length * Length

```
length = float(input("Enter the Length:"))
area = length * length
print("Area is", area)
```

Conditional Statements, Comparison and Logical Operators in Python

Working Example: Vision

We want to print "You are Passed" when Marks are greater than 50 but print "You are Failed" when Marks are less than or equal to 50.

Working Example: Vision

We want to print "You are Passed" when Marks are greater than 50 but print "You are Failed" when Marks are less than or equal to 50.

```
marks = float(input("Enter the Marks:"))
if(marks > 50):
    print("You are Passed")
else:
    print("You are Failed")
```

Body of Conditional Statements

In Python, the indentation (whitespace at the beginning) determines the body of each statement.

```
marks = float(input("Enter the Marks:"))
if(marks > 50):
    print("You are Passed")
else:
    print("You are Failed")
```

Body of Conditional Statements

Instead of else if python uses elif statement.

```
price=float(input("Enter Price"))
quantity=int(input("Enter Quantity"))
amount = price*quantity
if(amount > 200):
    if(amount >1000):
        print("The amount is greater than 1000")
    else:
        if(amount > 800):
            print("The amount is between 800 and 1000")
        elif(amount > 600):
            print("The amount is between 600 and 1000")
        else:
            print("The amount is between 200 and 1000")
elif(amount == 200):
    print("Amount is 200")
else:
    print("Amount is less than 200")
```

Logical Operators in Python

Instead of &&, || and ! symbols, Python uses exact words.

Logical Operator in Python	Explaination	Python Syntax
and	Returns True if both statements are true	if(x > 5 and x < 10):
or	Returns True if one of the statements is true	if(x < 5 or x < 4):
not	Reverse the result, returns False if the result is true	if(not(x > 5 and x < 10)):

Loops in Python

Loops in Python

We have two type of loops

- Counter Loops
- Conditional Loops

Counter Loop

Count the steps and execute until specific number of times.

Condition Loop

Execute the steps until a specific condition is not met.

Working Example: Vision

We want to print "Welcome Jack" 5 times.

Working Example: Vision

We want to print "Welcome Jack" 5 times.

```
for i in range(0,5,1):
    print("Jack")
```

Working Example: Vision

Write a program that keeps taking input from the user and sum up all these input until he enters -1. When user enters -1 the program should print sum of all values.

Working Example: Vision

Write a program that keeps taking input from the user and sum up all these input until he enters -1. When user enters -1 the program should print sum of all values.

```
num = int(input("Enter Number:"))
sum = 0
while(num != -1):
    sum = sum + num
    num = int(input("Enter Number:"))
print("Sum is", sum)
```

Working Example: Vision

Write a program that takes 3 numbers as input and print the largest of them.

Solution with variables

Write a program that takes 3 numbers as input and print the largest of them.

```
firstNumber = int(input("Enter First Number:"))
secondNumber = int(input("Enter Second Number:"))
thirdNumber = int(input("Enter Third Number:"))
if(firstNumber > secondNumber and firstNumber > thirdNumber):
    print("First Number is Largest")
elif(secondNumber > firstNumber and secondNumber > thirdNumber):
    print("Second Number is Largest")
else:
    print("Third Number is Largest")
```

Arrays in Python

Arrays: List in Python

You can access variables through same name with different index value [0]

value = [0,0,0,0,0];

Solution with variables

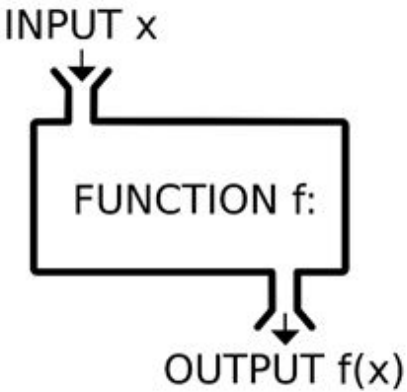
Write a program that takes 3 numbers as input and print the largest of them.

```
num = [0,0,0]
for i in range(0,3):
    num[i] = int(input("Enter Number:"))
largest = num[0]
for i in range(0,3):
    if(num[i] > largest):
        largest = num[i]
print("Largest is", largest)
```

Functions in Python

Working Example: Vision

Write a Function named addition that takes two parameters as input and then returns their sum.



Working Example: Vision

Write a Function named addition that takes two parameters as input and then returns their sum.

```
def add(a, b):
    return a+b

def main():
    num1 = int(input("Enter First Number"))
    num2 = int(input("Enter Second Number"))
    print(add(5,5))

if __name__ == "__main__":
    main()
```

Working Example: Vision

In the solution there are 2 functions.

1. main
2. add

There is also a conditional statement that checks the value of __name__ and compares it to the string "__main__". When the if statement evaluates to True, the Python interpreter executes main().

```
def add(a, b):
    return a+b

def main():
    num1 = int(input("Enter First Number"))
    num2 = int(input("Enter Second Number"))
    print(add(5,5))

if __name__ == "__main__":
    main()
```

Working Example: Vision

Important thing to note here is that we do not have to tell the return type of a function in Python.

```
def add(a, b):
    return a+b

def main():
    num1 = int(input("Enter First Number"))
    num2 = int(input("Enter Second Number"))
    print(add(5,5))

if __name__ == "__main__":
    main()
```

File Handling in Python

Working Example: Vision

Write a program that reads the data from the file if the file exists.

```
import os.path

def main():
    if(os.path.exists("data.txt")):
        f = open("data.txt", 'r')
        lines = f.read()
        f.close()
        print(lines)
    else:
        print("File does not exist")

if __name__ == "__main__":
    main()
```

Opening a File in Python

The following table shows available modes for opening a text file:

Mode	Description
'r'	Open for text file for reading text
'w'	Open a text file for writing text
'a'	Open a text file for appending text

Working Example: Vision

Write a program that Appends the data into the file.

Working Example: Vision

Write a program that Appends the data into the file.

```
import os.path

def main():
    if(os.path.exists("data.txt")):
        f = open("data.txt", 'a')
        f.write("\nNew Line")
        f.close()
    else:
        print("File does not exist")

if __name__ == "__main__":
    main()
```