INTRODUCTION TO PYTHON

1. Variables and Data types

- Variable: A variable is a container used to store and manipulate values.
- Data types: Data types are the formats in which information is stored.
- They are:
- Integers: (1, 2, 3….)
- Floats (1.1, 2.1, 3.1…)
- Strings ("hello")

```
x = 2
a = 5.1
y = ("hello")
print(x)
print(a)
print(y)

2
5.1
hello
```

2. Lists , Tuples and Dictionaries:

- Lists: are used to store multiple values in a single variable.
  Syntax:
- Lists: a = [1,2,3,4,5,6]
- Tuple: x= (1,2,2,2,2,3,4,5)
- Dictionaries my_dic= {"a": "apple" , "b": "ball", "c": "cat"}

Operations:

List:	Tuple	Dictionaries
a = [1,2,3,4,5,6] print (a[0]) print(a)	x= (1,2 ,3,4,5) print(x[0]) print(x)	y={"model" : "112"} print (y["model"])
1 [1,2,3,4,5]	1 (1,2,3,4,5)	112

Built-in functions:

count(): Used to count how many times a value appears in the list

```
a = [1,2,3,4,5,6]
```

```
print (a.count(1))      # Output : 1

index(): Returns the index of the first occurrence of the value.
a = [1,2,3,4,5,6]
print (a.index(2))       # Output : 1
print(a.index(4))        # Output : 3
```

Dictionaries

```
my_dict = {
   "fruit" : "banana" ,
   "vegetable" : "spinach",
   "dessert" :    "custard",
}

print (my_dict)

Access the value using the .get() function

x = my_dict.get("fruit")

print (x)

Output: banana
```

Note: Dictionaries have other methods for changing, deleting , or updating.

3. Python conditions and if statements

Python supports the usual logical conditions from mathematics:

- Equals: a == b
- Not Equals: a != b
- Less than: a < b
- Less than or equal to: a <= b
- Greater than: a > b
- Greater than or equal to: a >= b

```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```

Compare ages example:

```
name_1 = input("Enter your name : ")

age_1 = int(input("Enter your age : "))       # convert to int

# by default, input() in Python always returns a string. To get input as a numeric value, we use int() function to convert the inputs.

name_2 = input("Enter your name : ")

age_2 = int(input("Enter your age : "))       # convert  to int

if (age_1 > age_2):

   print( name_1, "is older than",name_2 )

else:

   print(name_1 , "is younger than",name_2)
```

Output:

```
Enter your name : nida
Enter your age : 23
Enter your name : minha
Enter your age : 32
nida is younger than minha
```

BMI CALCULATOR

```
name = input("Enter your name : ")

weight = int(input("Enter your weight in kgs : "))

height = float(input("Enter your height in meter : "))

bmi = weight /  (height **  2)

print(bmi)
```

Output:

```
Enter your name : minha

Enter your weight in kgs : 70

Enter your height in meter : 1.75
```

```
22.857142857142858
```

Python while loops

Python has two types of loops:

- while loop
- for loop

4a. while loop

while loop executes a set of statements as long as the condition is true.

```
i = 1
while i < 6:
  print(i)
  i += 1
```

Output :

```
1
2
3
4
5
```

4b. for loop:

for loop is used to iterate over a sequence and execute a block of code for each item.

Example 1: Loop through a list

```
fruits = ["apple", "mango", "orange"]
for fruit in fruits:
   print(fruit)
```

Output

```
apple
mango
orange
```

Example 2: Loop using range()

```
for i in range(8):   # i goes from 0 to 7
     print(i)
```

```
Output:
  0
  1
  2
  3
  4
  5
  6
  7
```

5. Control Statement

a. continue statement.

b. break statement

5a.The continue statement is used to skip the current iteration of a loop and move to the next one.

Example: Continue to the next iteration if i equals 3:

```
i = 0
while i < 6:
  i += 1
  if i == 3: #when this condition is met, it skip printing 3
     continue
   print(i)
Output:
1
2
4
5
6
```

5b.The break statement is used to immediately exit a loop when a certain condition is met.

Example: Break the loop when i equals to 5

```
i = 1
while i < 10:
      if i == 5:
        break
     print(i)
     i += 1
Output:
1
2
3
4
```