### *ANSWERS FOR ASSIGNMENT ONE*

### A1.

### Easy to Code-Python is a very high-level programming language, yet it is effortless to learn.

### Easy to Read-Python code looks like simple English words. There is no use of semicolons or brackets, and the indentations define the code block. You can tell what the code is supposed to do simply by looking at it.

### Free and Open-Source-Python is developed under an OSI-approved open source license. Hence, it is completely free to use, even for commercial purposes. It doesn't cost anything to download Python or to include it in your application.

### Portable-Python is portable in the sense that the same code can be used on different machines.

### Support for GUI-One of the key aspects of any programming language is support for GUI or Graphical User Interface. A user can easily interact with the software using a GUI.

### High-level Language-Python is a high-level programming language because programmers don’t need to remember the system architecture, nor do they have to manage the memory. This makes it super programmer-friendly and is one of the key features of Python.

### A2. **Python** is an object-oriented programming language like Java. Python doesn’t convert its code into machine code, something that hardware can understand. It actually converts it into something called byte code. So within python, compilation happens, but it’s just not into a machine language. It is into byte code (.pyc or .pyo) and this byte code can’t be understood by the CPU. So we need an interpreter called the python virtual machine to execute the byte codes.

### 

A3. Indentation refers to the spaces at the beginning of a code line.

Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important.

Python uses indentation to indicate a block of code.

Example

if 5 > 2:  
  print("Five is greater than two!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_indentation)

Python will give you an error if you skip the indentation.

A4. In Python, we use the hash symbol # to write a single-line comment.

### Example 1: Writing Single-Line Comments

# printing a string

print('Hello world')

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Hello world

Here, the comment is:

# printing a string

This line is ignored by the Python interpreter.

Everything that comes after # is ignored. So, we can also write the above program in a single line as:

print('Hello world') #printing a string

[Run Code](https://www.programiz.com/python-programming/online-compiler)

The output of this program will be the same as in **Example 1**. The interpreter ignores all the text after #.

## Multi-Line Comments in Python

Python doesn't offer a separate way to write multiline comments. However, there are other ways to get around this issue.

We can use # at the beginning of each line of comment on multiple lines.

### Example 2: Using multiple #

# it is a

# multiline

# comment

Here, each line is treated as a single comment and all of them are ignored.

IMPORTANCE OF COMMENTS : Using comments in programs makes our code more understandable. It makes the program more readable which helps us remember why certain blocks of code were written.

Other than that, comments can also be used to ignore some code while testing other blocks of code. This offers a simple way to prevent the execution of some lines or write a quick pseudo-code for the program.

A5.

Unpacking in Python refers to **an operation that consists of assigning an iterable of values to a tuple (or list ) of variables in a single assignment statement**. As a complement, the term packing can be used when we collect several values in a single variable using the iterable unpacking operator, \*

A6.

The process of converting a [Python data type](https://intellipaat.com/blog/tutorial/python-tutorial/python-datatypes/) into another data type is known as type conversion. There are mainly two types of type conversion methods in Python, namely, implicit type conversion and explicit type conversion.

* Type conversion is the process of **converting a data type into another data type**.
* Implicit type conversion is performed by a Python interpreter only.
* Explicit type conversion is performed by the user by explicitly using type conversion functions in the program code.
* Explicit type conversion is also known as **typecasting**.
* Python, when performing implicit typecasting, avoids the loss of data.

A7.

Using loops in Python automates and repeats the tasks in an efficient manner. But sometimes, there may arise a condition where you want to exit the loop completely, skip an iteration or ignore that condition. These can be done by **loop control statements**.

## Break statement

The break statement is used to terminate the loop or statement in which it is present. After that, the control will pass to the statements that are present after the break statement, if available. If the break statement is present in the nested loop, then it terminates only those loops which contains break statement.

EXAMPLE:

for letter in s:

print(letter)

# break the loop as soon it sees 'e'

# or 's'

if letter == 'e' or letter == 's':

break

print("Out of for loop")

print()

i = 0

**Output:**

g

e

Out of for loop

CONTINUE STATEMENT

As the name suggests the continue statement forces the loop to continue or execute the next iteration. When the continue statement is executed in the loop, the code inside the loop following the continue statement will be skipped and the next iteration of the loop will begin.

EXAMPLE:

|  |
| --- |
| # loop from 1 to 10  **for** i **in** range(1, 11):        # If i is equals to 6,      # continue to next iteration      # without printing  **if** i **==** 6:  **continue**  **else**:          # otherwise print the value          # of i          print(i, end **=** " ") |

**Output:**

1 2 3 4 5 7 8 9 10

## Pass statement

As the name suggests pass statement simply does nothing. The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute. It is like null operation, as nothing will happen is it is executed. Pass statement can also be used for writing empty loops. Pass is also used for empty control statement, function and classes.

EXAMPLE

|  |
| --- |
| s **=** "geeks"    # Empty loop  **for** i **in** s:      # No error will be raised  **pass**    # Empty function  **def** fun():  **pass**    # No error will be raised  fun()    # Pass statement  **for** i **in** s:  **if** i **==** 'k':  **print**('Pass executed')  **pass**      print(i) |

**Output:**

G

e

e

Pass executed

k

s

A8.

The Python**range() function**returns the sequence of the given number between the given range. The most common use of it is to iterate sequence type (**Python range() List**, string, etc. ) with for and while loop using [Python](https://www.geeksforgeeks.org/python-programming-language/).

## Example of Python range()

|  |
| --- |
| # printing a number  **for** i **in** range(0,10,2):      print(i, end**=**" ")  print() |

**Output:**

0 2 4 6 8

A9.

while loop repeats the sequence of actions many times until some condition evaluates to False. The condition is given before the loop body and is checked before each execution of the loop body. Typically, the while loop is used when it is impossible to determine the exact number of loop iterations in advance.

The syntax of the while loop in the simplest case looks like this:

while some condition:

a block of statements

ISKE AAGE KA NAHI MILL RHA ITNA HI LIKHLE

A10.

### if-else statement in Python 3

What if you want to execute some code if the given condition is false?

Here is the if-else statement you need to implement.

#### Python syntax for the if-else statement:

if <condtion>:

<your\_if\_code>

else:

<your\_else\_code>

#### Python Program for if-else statement:

**Example:** Write a Python program to print whether if it is a zero or non-zero number.

|  |  |
| --- | --- |
| 1  2  3  4  5 | nNum **=** 1  **if** nNum **==** 0:    print("Number is zero.")  **else**:    print("Number is not zero.") |

**Output:**

Number is not zero.

### Nested if else in Python 3

**Can we add if statement inside another if statement in Python?**

Yes.

#### Python syntax for the nested if-else statement:

if <condtion>:

<your\_code\_block\_1>

if <condition\_1\_1>:

<your\_code\_block\_1\_1>

else:

<your\_code\_block\_1\_2>

elif <condtion>:

<your\_elif\_code\_block>

else:

<your\_else\_code\_block>

#### Python Program for nested if-else in Python:

**Example:** Write a Python program to print if the number is positive, negative or zero, using nested if-else statement.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | nNum **=** 1  **if** nNum !**=** 0:  **if** nNum > 0:      print("Number is a positive.")  **else**:      print("Number is a negative.")  **else**:    print("Number is zero.") |

**Output:**

Number is a positive.

With this, you can check multiple conditions in Python if statement.