Assignment : 02

Question (1). What are the two values of the Boolean data type? How do you write them?

Solution : The Boolean data type in Python has two possible values: True and False. These values represent the concepts of truth and falsehood, or on and off, in programming logic.

We can write them as True and False. Here capitalization is important because Python treats True and False as keywords and will recognize them only when they are written in all uppercase letters.

Queston (2). What are the three different types of Boolean operators?

Solution : Boolean operators are AND, OR, NOT

Queston (3). Make a list of each Boolean operator&#39;s truth tables (i.e. every possible combination of Boolean values for the operator and what it evaluate).

Solution :

Logical AND (and):

Operand 1	Operand 2	Result
False	False	False
False	True	False
True	False	False
True	True	True

Logical OR (or):

Operand 1	Operand 2	Result
False	False	False
False	True	True
True	False	True
True	True	True

Logical NOT (not):

Operand	Result
False	True
True	False

Question (4). What are the values of the following expressions?

(5 &gt; 4) and (3 == 5)

not (5 &gt; 4)

(5 &gt; 4) or (3 == 5)

not ((5 &gt; 4) or (3 == 5))

(True and True) and (True == False)

(not False) or (not True)

Solution :

1). (5 > 4) and (3 == 5)

Result: False

2). not (5 > 4)

Result: False

3). (5 > 4) or (3 == 5)

Result: True

4). not ((5 > 4) or (3 == 5))

Result: False

5). (True and True) and (True == False)

Result: False

6). (not False) or (not True)

Result: True

Question (5). What are the six comparison operators?

Solution :

Equal (==)

Not Equal (!=)

Greater Than (>)

Less Than (<)

Greater Than or Equal To (>=)

Less Than or Equal To (<=)

Question (6). How do you tell the difference between the equal to and assignment operators?Describe a condition and when you would use one.

Solution :

Equal To Operator (==) : The equal to operator is used to compare two values to determine whether they are equal. It returns a Boolean value True if the values are equal, and False if they are not.

Example: Python code

```
x = 5

y = 7

result = x == y
```

Assignment Operator (=) : The assignment operator is used to assign a value to a variable. It takes the value on the right and assigns it to the variable on the left.

Example: Python code

```
x = 10  # Assigns the value 10 to the variable x
```

Comparison Example:

Suppose you want to check if a user's input is equal to a predefined password. You would use the equal to operator to compare the input with the password:

Python Code

```
user_input = input("Enter the password: ")
```

```
password = "secret"

if user_input == password:
print("Access granted")
else:
print("Access denied")
```

In this example, the equal to operator (==) is used to compare the user's input with the password. If they are equal, the program grants access; otherwise, it denies access.

Question (7). Identify the three blocks in this code:

```
spam = 0
if spam == 10:
print(&#39;eggs&#39;)
if spam &gt; 5:
print(&#39;bacon&#39;)
else:
print(&#39;ham&#39;)
print(&#39;spam&#39;)
print(&#39;spam&#39;)
```

Solution : The blocks are determined by the level of indentation. Block 1 is the code indented under the first if statement, Block 2 is the code indented under the second if statement, and Block 3 is the code indented under the else statement. The remaining print statements are not indented, so they are not part of any specific block.

Question (8). Write code that prints Hello if 1 is stored in spam, prints Howdy if 2 is stored in spam, and prints Greetings! if anything else is stored in spam.

Solution : Python Code :

```
spam = int(input("Enter a value for spam: "))  # Taking user input as an example


if spam == 1:
   print("Hello")
elif spam == 2:
   print("Howdy")
else:
   print("Greetings!")
```

<u>Question (9).</u> If your programme is stuck in an endless loop, what keys you'll press?

<u>Solution :</u> CTRL + C

<u>Question (10)</u>. How can you tell the difference between break and continue?

<u>Solution :</u>

<u>break Statement</u> : The break statement is used to immediately exit the current loop, regardless of whether the loop's condition is still met or not. When a break statement is encountered, the loop is terminated, and the program continues with the next statement after the loop.

Example:

```
for i in range(5):
if i == 3:
break
print(i)
```

In this example, the loop will print 0, 1, and 2, but when i becomes 3, the break statement is executed, and the loop terminates.

<u>continue Statement</u> : The continue statement is used to skip the current iteration of the loop and move to the next iteration. The remaining code within the loop after the continue statement is not executed for that specific iteration.

Example:

```
for i in range(5):
if i == 2:
continue
print(i)
```

Question (11). In a for loop, what is the difference between range(10), range(0, 10), and range(0, 10, 1)?

Solution :

range(10) : This form generates a sequence starting from 0 and ending at 9 (i.e., it includes all numbers from 0 to 9). The increment is implicitly set to 1 by default.

range(0, 10) : This form explicitly specifies both the start and stop values. It starts from 0 and stops before reaching 10 (i.e., it includes numbers from 0 to 9). The increment is implicitly set to 1.

range(0, 10, 1) : This form specifies the start, stop, and step values explicitly. It starts from 0, stops before reaching 10, and increments by 1 at each step.

Question (12). Write a short program that prints the numbers 1 to 10 using a for loop. Then write an equivalent program that prints the numbers 1 to 10 using a while loop.

Solution :

```
for i in range(1, 11):
print(i)
```

Question (13). If you had a function named bacon() inside a module named spam, how would you call it after importing spam?

Solution :

```
import spam

spam.bacon()          # Calling the bacon() function from the spam module
```