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

Ans – Two types of Boolean data type are:

* True
* False

1. x = 4
2. y = 5
4. a = x < y
5. b = x > y
6. print(a)
7. True
8. print(b)
9. False
10. type(a)
11. type(b)

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

Ans -

* **and**
* **or**
* **not**

These Boolean operators are used to compare Boolean values

**and** or **not** called binary operators because they take two Boolean values

**and** operator will evaluate to True if both Boolean values are true otherwise False

True and True  
True

True and False  
False

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

Ans - Boolean operator’s–

* Truth table for **and** operator’s

|  |  |  |
| --- | --- | --- |
| x | y | x and y |
| False | False | False |
| False | True | False |
| True | False | False |
| True | True | True |

* Truth table for **or** operator’s

|  |  |  |
| --- | --- | --- |
| x | y | x or y |
| False | False | False |
| False | True | True |
| True | False | True |
| True | True | True |

* Truth table for **not** operator’s

|  |  |  |
| --- | --- | --- |
| x | y | not x |
| False | True | True |
| True | False | False |

4. What are the values of the following expressions?

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

not (5 > 4)

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

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

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

(not False) or (not True)

Ans -

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

False

1. not(5 > 4)

False

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

True

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

False

1. (True and True) and (True == False)

False

1. (not False) or (not True)

True

5. What are the six comparison operators?

Ans -

Example

x = 4

y = 5

* Comparison Operator

|  |  |  |
| --- | --- | --- |
| operate | Meaning | Example |
| == | Equal to | x == y |
| != | Not equal to | x != y |
| < | Less than | x < y |
| > | Greater than | x > y |
| <= | Less than equal to | x <= y |
| >= | Greater than equal to | x >= y) |

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

Ans –

== is the equal operator

= is the assignment operator

* Equal to == operator checks whether the two given operands are equal or not. If it is equal then it returns true otherwise it will returns false.

2 == 2

This will return true

* assignment operator  **=** is used for assigning the value to a variable.

a = 10

b = 11

7. Identify the three blocks in this code:

spam = 0

if spam == 10:

print('eggs')

if spam > 5:

print('bacon')

else:

print('ham')

print('spam')

print('spam')

Ans –

1. spam = 0
3. ## from line number 4 to 5 is Block 1
4. if spam == 10:
5. print('eggs')
7. ## from line number 8 to 9 is Block 2
8. if spam > 5:
9. print('bacon')
11. ## from line number 12 to 15 is Block 2
12. else:
13. print('ham')
14. print('spam')
15. print('spam')

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.

Ans –

1. spam = int(input())
2. if spam == 1: ## this condition will prints Hello if 1 is stored in spam
3. print("Hello")
4. elif spam == 2: ## this condition will prints Howdy if 2 is stored in spam
5. print("Howdy")
6. else: ## this condition will prints Greetings!
7. print("Greetings!")



9.If your programme is stuck in an endless loop, what keys you’ll press?

Ans –

Endless loop occurs when a program keeps executing within one loop and it never leaves that block.

To come out of that endless loop use CTRL + C.

10. How can you tell the difference between break and continue?

Ans –

Difference between break and continue –

break – when break keyword is found, it will terminate the current loop and come out of the loop immediately

1. 1
2. num = [10, 20, 30, 40, 50, 25, 35]
3. for i in num:
4. if i > 40:
5. break
6. print(i)
7. 10
8. 20
9. 30
10. 40

continue - current iteration that is running will be stopped and move to the next iteration

1. num = [10, 20, 30, 40, 50, 25, 35]
2. for i in num:
3. # skip below statement if number is greater than 40
4. if i > 40:
5. continue
6. print(i)
8. 10
9. 20
10. 30
11. 40
12. 25
13. 35

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

Ans –

There is no difference value/output of the given range will be same

the range(10) returns the range object with the default start 0, stop 10, and default step 1

1. num\_range\_1 = range(10) #start=0, stop=10, step=1
2. print(num\_range\_1)
3. for i in num\_range\_1:
4. print('Values = ', num\_range\_1[i])
5. range(0, 10)
6. Values = 0
7. Values = 1
8. Values = 2
9. Values = 3
10. Values = 4
11. Values = 5
12. Values = 6
13. Values = 7
14. Values = 8
15. Values = 9

the range(0,10) returns the range object with the default start 0, stop 10, and default step 1

1. num\_range\_2 = range(0,10) #start=0, stop=10, step=1
2. print(num\_range\_2)
3. for i in num\_range\_2:
4. print('Values = ', num\_range\_2[i])
6. range(0, 10)
7. Values = 0
8. Values = 1
9. Values = 2
10. Values = 3
11. Values = 4
12. Values = 5
13. Values = 6
14. Values = 7
15. Values = 8
16. Values = 9

the range(0,10,1) returns the range object with the default start 0, stop 10, and default step 1

1. num\_range\_3 = range(0,10,1) #start=0, stop=10, step=1
2. print(num\_range\_3)
3. for i in num\_range\_3:
4. print('Values = ', num\_range\_3[i])
5. range(0, 10)
6. Values = 0
7. Values = 1
8. Values = 2
9. Values = 3
10. Values = 4
11. Values = 5
12. Values = 6
13. Values = 7
14. Values = 8
15. Values = 9

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.

Ans –

print the numbers from 1 to 10 using a for loop

1. for i in range(1,11):
2. print(i)


6. 1
7. 2
8. 3
9. 4
10. 5
11. 6
12. 7
13. 8
14. 9
15. 10

print the numbers from 1 to 10 using a while loop

1. num = 1
2. while num <= 10:
3. print(num)
4. num = num +1
5. 1
6. 2
7. 3
8. 4
9. 5
10. 6
11. 7
12. 8
13. 9
14. 10

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

Ans –

spam. bacon()

this way function can be called