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

Boolean values have two possible data types: true and false. Generally, it is used to represent the truth values of the expressions. For example, 1==1 is True whereas 2<1 is False. In binary, these are represented by 1 and 0.

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

Three different types of Boolean operators are AND, OR, and NOT which are used as junctions to include or exclude keywords in a search, resulting in more focused and effective results.

1. 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 ).

True and True is True.

False and True is False.

False and False is False.

True or False is True.

not True is False.

not False is True.

False or False is False.

True or True is True.

True and False is False.

False or True is True.

Truth is 1 and False is 0.

Truth Table for AND  
A B output  
0 0 0  
0 1 0  
1 0 0  
1 1 1  
Truth Table for OR  
A B output  
0 0 0  
0 1 1  
1 0 1  
1 1 1  
Truth Table for NOT  
A output  
0 1  
1 0

1. 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)

False

False

True

False

False

True

1. What are the six comparison operators?

Six comparison operators are ==, !=, <, >, <=, and >=

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

== is equal to an operator that compares two values and evaluates to a Boolean, while = is the assignment operator that stores a value in a variable.

*#eg.*

*# Equal To Operator*

**if**(2**==**3):

print("True")

**else**:

print("False")

*#Assignment operator*

c**=**1 *#here we have used assignment operator(=) to assign value of c which is 1*

print("c =",c)

False

c = 1

1. 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')

ham

spam

spam

1. 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.

Input a no.2

Howdy

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

If a program is stuck in an endless loop we will press ctrl+c.

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

**for** i **in** range(10):

**if**(i**==**7):

**break**

print(i)

print('Breaked')

*#use of continue*

**for** i **in** range(10):

**if**(i**==**7):

**continue**

print(i)

0

1

2

3

4

5

6

Breaked

0

1

2

3

4

5

6

8

9

Henceforth, The break statement will move the execution outside and just after a loop. The continue statement will move the execution to the start of the loop.

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

**for** i **in** range(10):

print(i)

print("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx")

**for** i **in** range(0,10):

print(i)

print("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx")

**for** i **in** range(0,10,1):

print(i)

0

1

2

3

4

5

6

7

8

9

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

0

1

2

3

4

5

6

7

8

9

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

0

1

2

3

4

5

6

7

8

9

From the overhead output we can figure that they all do the same thing. The range(10) call ranges from 0 up to (but not including) 10, range(0, 10) tells the loop to start at 0, and range(0, 10, 1) tells the loop to increase the variable by 1 on each iteration.

1. 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.

print("For Loop")

**for** i **in** range(1,11):

print(i)

*#Use of While Loop*

print("While Loop")

a **=**1

**while** a **<=** 10:

print(a)

a**+=**1

For Loop

1

2

3

4

5

6

7

8

9

10

While Loop

1

2

3

4

5

6

7

8

9

10

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

This function can be called with spam.bacon().