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

Ans: Two values of Boolean data type are True and False. We have to write in capital letters only. While we are using comparison operators (>, <, <=, >=, ==, and ! =) and logical operators (and, or and not) in the expressions we will get Boolean data type results TRUE or FALSE. In python membership operators also available in and is, these operators also give results in the Boolean values.

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

Ans: Three different types of Boolean operators are the and, or and not. These are the operators will give results in Boolean data types True or 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 evaluates).

1. AND: it evaluates the expression if both expressions are True it will give True otherwise False.

The following is the truth table of AND operator.

|  |  |  |
| --- | --- | --- |
| Boolean Value of Expression one | Boolean value of expression two | Result |
| TRUE | TRUE | TRUE |
| TRUE | FALSE | FALSE |
| FALSE | TRUE | FALSE |
| FALSE | FALSE | FALSE |

1. OR: It evaluates the expressions if both the expressions are FALSE, then it gives results as a FALSE, otherwise it gives results as TRUE. The following is the truth table for the OR operator.

|  |  |  |
| --- | --- | --- |
| Boolean Value of Expression one | Boolean value of expression two | Result |
| TRUE | TRUE | TRUE |
| TRUE | FALSE | TRUE |
| FALSE | TRUE | TRUE |
| FALSE | FALSE | FALSE |

1. NOT: It gives contrary to the statement. If TRUE it gives FALSE or FALSE it gives TRUE. The following is the truth table for the NOT operator.

|  |  |
| --- | --- |
| Expression | Result |
| TRUE | FALSE |
| FALSE | TRUE |

4. What are the values of the following expressions?

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

not (5 > 4) = False

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

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

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

(not False) or (not True) = True

5. What are the six comparison operators?

Ans: The operators which are used to compare two operands are called comparison operators. Those are less than (<), greater than (>), less than or equal ( <=), greater than or equal (>=), equal to(==), and not equal( ! =). These operators generate the binary results TRUE or FALSE.

Ex: In the following examples a and b are operands and the symbols between the operands are operators.

a=9, b=10

print(a==b) False

print(a>b) False

print(a<b) True

print(a<=b) True

print(a>=b) False

print(a!=b) True

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

Ans: An operator which is used to assign a value to the variable or name of memory location is called Assignment operator, it was declared by equal sign (=). The assignment operator will not produce any results it will only assign a value to the variable.

An operator which is used to check the two operands are equal or not is called equal operator. It is declared by double equal sign (==). And it will give results in Binary operators, if both the operands are equal it give TRUE otherwise FALSE.

Example:

a=5

b=a

In this example, a is first assigned with 5 then after a value is assigning to the variable b.

a==b, it will give True because of two variables memory location have the value of 5. And it will give a result True.

7. Identify the three blocks in this code:

spam = 0

if spam == 10:

print('eggs') 🡪 block one

if spam > 5:

print('bacon') 🡪 block two

else:

print('ham') 🡪 block three

print('spam')

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.

Spam= int(input())

if spam==1:

print(“Hello”)

elif spam==2:

print(“Howdy”)

else :

print(“Greetings”)

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

We will use interrupt option in the Kernel menu bar. Or we can click two time “i”, it will interrupt the endless loop.

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

Break and continue are jumping statements, both are used to skip the control or loop of iterations.

While break is used to immediately out to the loop if condition is satisfied and continue is use to skip the present iteration and continue to next iteration.

Example:

n=int(input('enter the input value for n: '))

for i in range(2,n-1):

if n%i==0:

print('it is not a prime number')

break

else:

print('{} is a prime number'.format(n))

In the above program, it write to find out the given number is prime or not. If given number is divisible 2 to n-1 numbers the condition was satisfied and the statement break will execute then immediately the loop end and it will give not a prime number.

n=int(input(“enter the input value for n”))

for i in range(0,n):

if n==0:

continue

print(i\*i)

In the above example it will prints the square of a number except 0, in the if condition mentioned n equals to 0 then pass the continue statement, it will not execute the square of zero and remain numbers n-1 squares will print.

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

Three of range functions give the same result. In this function, must be need to declare ending number of range. Remain starting and step parameters are optional.

range(10) executes the range of numbers from 0 to 10 and 10 is exclusive. It will step by 1. Here we have not mentioned starting point, it automatically take 0 as a starting range. And step here not mentioned it is optional predefine it takes as 1.

range(0,10) executes the range of numbers from 0 to 10 and 10 is exclusive. It will step by 1. Here it was mentioned starting of range zero and it assumes predefined as 0 .

range(0,10,1) executes the range of numbers from 0 to 10 and 10 is exclusive. It will step by 1. But in this statement position variable of step it is optional, predefine it will take step 1.

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:

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

print(i)

1. n,j=11,1

while j<n:

print(j)

j=j+1

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

Ans: Spam.bacon()