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

Ans: It has two possible values: True and False , which are special versions of 1 and 0 respectively and behave as such in arithmetic contexts.

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

Ans: Boolean Operators are simple words (AND, OR, NOT or AND NOT) used as conjunctions to combine or exclude keywords in a search, resulting in more focused and productive results.

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: AND Operator (denoted as ∧):

| Input A | Input B | Output |

|---------|---------|--------|

| False | False | False |

| False | True | False |

| True | False | False |

| True | True | True |

OR Operator (denoted as ∨):

| Input A | Input B | Output |

|---------|---------|--------|

| False | False | False |

| False | True | True |

| True | False | True |

| True | True | True |

NOT Operator (denoted as ¬):

| Input | Output |

|-------|--------|

| False | True |

| True | False |

These truth tables represent all possible combinations of Boolean values for the respective operators and their corresponding evaluations.

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**: Let's evaluate the given expressions one by one:

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

(5 > 4) evaluates to True.

(3 == 5) evaluates to False.

True and False evaluates to False.

Therefore, the value of the expression is False.

not (5 > 4)

(5 > 4) evaluates to True.

Applying the not operator to True results in False.

Therefore, the value of the expression is False.

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

(5 > 4) evaluates to True.

(3 == 5) evaluates to False.

True or False evaluates to True.

Therefore, the value of the expression is True.

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

(5 > 4) evaluates to True.

(3 == 5) evaluates to False.

True or False evaluates to True.

Applying the not operator to True results in False.

Therefore, the value of the expression is False.

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

True and True evaluates to True.

True == False evaluates to False.

True and False evaluates to False.

Therefore, the value of the expression is False.

(not False) or (not True)

Applying the not operator to False results in True.

Applying the not operator to True results in False.

True or False evaluates to True.

Therefore, the value of the expression is True.

To summarize, the values of the given expressions are:

False

False

True

False

False

True

5. What are the six comparison operators?

Ans: The six comparison operators in most programming languages are:

Equal to (==): Compares if two values are equal.

Not equal to (!=): Compares if two values are not equal.

Greater than (>): Compares if the left operand is greater than the right operand.

Less than (<): Compares if the left operand is less than the right operand.Greater than or equal to (>=): Compares if the left operand is greater than or equal to the right operand.Less than or equal to (<=): Compares if the left operand is less than or equal to the right operand.

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

Ans: The equal to operator (==) and the assignment operator (=) serve different purposes in programming languages. m:

Equal to Operator (==):

The equal to operator is used for comparison between two values to check if they are equal.

It returns True if the values are equal and False if they are not.

It is commonly used in conditional statements and expressions to make decisions based on equality.

Example:

python

x = 5

y = 7

if x == y:

print("x and y are equal")

else:

print("x and y are not equal")

In the above example, the equal to operator is used to compare the values of x and y. Since they are not equal (5 != 7), the output will be "x and y are not equal."

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.

It does not compare values but rather performs the assignment operation.

Example:

python

x = 5

y = x

print("The value of y is:", y)

In the above example, the assignment operator is used to assign the value of x (which is 5) to the variable y. The value of y will be 5 as well.

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:

Block 1:

python

Copy code

spam = 0

if spam == 10:

print('eggs')

Block 2:

python

Copy code

if spam > 5:

print('bacon')

else:

print('ham')

Block 3:

Python

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.

Ans:

spam = # Assuming a value is stored in spam

if spam == 1:

print('Hello')

elif spam == 2:

print('Howdy')

else:

print('Greetings!')

In the code above, we use the if-elif-else statement to check the value stored in the variable spam and perform different actions based on the value. If spam is equal to 1, it will print "Hello". If spam is equal to 2, it will print "Howdy". Otherwise, if any other value is stored in spam, it will print "Greetings!".

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

Ans: can stop an infinite loop with CTRL + C

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

Ans: The primary difference between break and continue statement in C is that the break statement leads to an immediate exit of the innermost switch or enclosing loop. On the other hand, the continue statement begins the next iteration of the while, enclosing for, or do loop.

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

Ans: range(10):

This variation specifies a single argument, which represents the end value of the range (exclusive).

It generates a sequence of numbers starting from 0 (default start value) up to, but not including, 10.

The step value is implicitly set to 1 (default step value).

range(0, 10):

This variation specifies two arguments: start and end values of the range.

It generates a sequence of numbers starting from 0 (specified start value) up to, but not including, 10 (specified end value).

The step value is implicitly set to 1 (default step value).

range(0, 10, 1):

This variation specifies three arguments: start, end, and step values of the range.

It generates a sequence of numbers starting from 0 (specified start value) up to, but not including, 10 (specified end value), incrementing by 1 (specified step value) at each iteration.

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: Certainly! Here's a short program that prints the numbers 1 to 10 using a for loop:

python

# Using a for loop

for i in range(1, 11):

print(i)

And here's an equivalent program that prints the numbers 1 to 10 using a while loop:

python

# Using a while loop

num = 1

while num <= 10:

print(num)

num += 1

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

Ans: If you have a function named bacon() inside a module named spam, you can call it after importing the spam module using the dot notation. Here's how you can do it:

python

import spam

spam.bacon()