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

* True: Represents the logical value of true.
* False: Represents the logical value of false.

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

* ***AND:*** The AND operator returns true if both operands are true; otherwise, it returns false.
* ***OR:*** The OR operator returns true if at least one of the operands is true; otherwise, it returns false.
* ***NOT:*** The NOT operator (also known as the negation operator) negates the truth value of its operand. It returns true if the operand is false and false if the operand is true.

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 ).

* ***AND operator (&&):***

| ***Operand 1*** | ***Operand 2*** | ***Result*** |
| --- | --- | --- |
| true | true | true |
| true | false | false |
| false | true | false |
| false | false | false |

* ***OR operator (||):***

| ***Operand 1*** | ***Operand 2*** | ***Result*** |
| --- | --- | --- |
| true | true | true |
| true | false | true |
| false | true | true |
| false | false | false |

* ***NOT operator (!):***

| ***Operand*** | ***Result*** |
| --- | --- |
| true | false |
| false | true |

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)

* ***(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?

* **Equal to (==)**
* **Not equal to (!=)**
* **Greater than (>)**
* **Less than (<)**
* **Greater than or equal to (>=)**
* **Less than or equal to (<=)**

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

***Equal to operator (==):***

* The equal to operator compares two values to check if they are equal.
* It returns True if the values are equal and False if they are not.
* Example:

x = 5

y = 10

if x == y:

print("x is equal to y")

else:

print("x is not equal to y")

***Assignment operator (=):***

* The assignment operator is used to assign a value to a variable.
* It assigns the value on the right-hand side to the variable on the left-hand side.
* Example:

x = 5

y = x + 3

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')

***Block 1:***

if spam == 10:

print('eggs')

***Block 2:***

if spam > 5:

print('bacon')

***Block 3:***

else:

print('ham')

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 = # Value assigned to spam

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?

Ctrl + C

Pressing Ctrl + C sends an interrupt signal to the running program, causing it to terminate and exit the loop. This key combination is commonly used to break out of infinite loops or stop the execution of a program that is not responding.

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

***break statement:***

* When encountered within a loop (such as a for or while loop), the break statement immediately terminates the loop and moves the execution to the next statement after the loop.
* It allows you to exit the loop prematurely based on a specific condition.
* After encountering a break statement, the remaining iterations of the loop are skipped, and the control flow moves to the statement following the loop.

***continue statement:***

* When encountered within a loop, the continue statement skips the rest of the current iteration and moves the execution to the next iteration of the loop.
* It allows you to bypass the remaining statements within the loop for the current iteration but continues with the next iteration.
* After encountering a continue statement, the loop's control flow jumps to the next iteration, skipping any remaining statements in the current iteration.

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

In a for loop, there is no difference between range(10), range(0, 10), and range(0, 10, 1). They all generate the same sequence of numbers from 0 to 9 (excluding 10) with a step size of 1.

The range() function in Python allows you to create a sequence of numbers, and it can be used with one, two, or three arguments. When only one argument is provided, it represents the stop value, and the start value defaults to 0, while the step value defaults to 1.

Therefore, range(10), range(0, 10), and range(0, 10, 1) all produce the same sequence of numbers: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. The differences in syntax are purely a matter of style or personal preference; functionally, they are equivalent.

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.

***using a for loop:***

for num in range(1, 11):

print(num)

***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?

import spam

spam.bacon()