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

Ans: The two value of Boolean data type are as follows:

a) True(traditional ON) b) False(traditional off)

We can assign the values of **T**rue and **F**alse(assigned keywords) to variables to declare these kinds of values. Type of this data type will return bool(short form of boolean). The first letters of both the keywords is taken in upper case and rest in lower case.
E.g
x=**T**rue
Or
y= **F**alse

Boolean values find tremendous applications in logical operations and decision making part of the code.

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

Ans: The three types of Boolean operators are as follows:

a) and
b) or
c) not

a)“and” takes two inputs and returns true only if both the inputs are True.
b) “or” takes two inputs and returns true if one of the inputs is True
c) “not” takes one input and returns the opposite of the input value

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).

Ans: i) **and** operation truth table

Input1	Input2	Output
True	True	True
True	False	False
False	True	False
False	False	False

“and” takes two inputs and returns true only if both the inputs are True.

ii) or operation truth table

Input 1	Input 2	Output
True	True	True
True	False	True
False	True	True
False	False	False

“**or**” takes two inputs and returns true if one of the inputs is True

iii) “**not**” operation truth table

Input	Output
False	True
True	False

“**not**” takes one input and returns the opposite of the input value

4. What are the values of the following expressions?

(5 > 4) and (3 == 5) Ans: False
not (5 > 4) Ans: False
(5 > 4) or (3 == 5) Ans: True
not ((5 > 4) or (3 == 5)) Ans: False
(True and True) and (True == False) Ans: False
(not False) or (not True) Ans: True

5) What are the six comparison operators?

Ans: The six comparison operators are as follows:

a) > (greater than)
b) < (less than)
c) >= (greater or equal to)
d) <= (less or equal to)
e) == (equal to)
f) =! (not equal)

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

Ans: To assign the value to a variable, symbol “**=**” is used.
For example,
x=7
The above line of code assigns the value of 7 to the variable x.

However, the comparison equal to operator uses “**==**” to check for equality between two values

For example

x=7 ## Assigns the value of 7 to the variable x
If x==7: ## checks whether the x variable has 7 in it(makes comparison
    True

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: The above code puts the value **0** in the variable **spam**

In the first block of code, it checks whether **spam is equal to 10,** as the spam value which is zero(0) is not equal to 10, it does not execute the code in this block(
In the second block of code, it checks whether **spam is greater than 5,** as the spam value which is zero(0) is not greater than 5, it does not execute the code in this block.
In the third block, the else block, all values of spam except the above two blocks return True, so the value of 0 falls in this block,the condition is set as true, the code in this block gets executed.
Output of code:
Ham
spam
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=int(input()) ##takes input for 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 will you press?

Ans: In most python editors, Ctrl+C kills the lop and helps us exit an infinite loop

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

Ans: The **break** keyword terminates the loop when a specific condition is met and the **continue** keyword skips the said part of code if the condition is met

Example:
```
for i in range(0,5):
        if i==3:
                break
        print(f"The number is {i}")
```
The following code iterates the value of i from 0 to less than 5.
For the values of 0,1,2, it executes the line after. However, when i is 3, it encounters the break statement and exits the loop.
Output:
The number is 0
The number is 1
The number is 2

```
for i in range(0,5):
        if i==3:
                continue
        print(f"The number is {i}")
```
The following code iterates the value of i from 0 to less than 5.
For the values of 0,1,2,4, the code executes the print statement. However, when i is 3, it skips the print statement and continues running the loop.

Output:
The number is 0
The number is 1
The number is 2
The number is 4

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

Ans:a) **range(10)**, in a for loop, for i in range(10), this statement initiates the value of i as 0 and iterates the value i<10(10 is excluded, exclusive range). Or we can say it stops the iteration when 10 is encountered(10 exclude) Since, python takes 0 as initial value, it can be said, the loop runs 10 times

b) **range(0,10**): as the first argument in range() is optional, it still initiates the starting value as 0 and runs same as range(10) as explained above

c) **range(0,10,1):** the third argument in the range(), gives the step increment to the iterative value. However, as the value here is 1, the step will be of 1 increment which is same as the iteration of for loop so it would be same as range(10) or range(0,10)

In all the ranges, the loop will be executed 10 times.

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: A)

```
for i in range(1,11):
                    print(i)

i=1
while i in range(11):
                    print(i)
                    i=i+1
```

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

Ans: Following will be the code:

```
import spam
spam.bacon()## will call the function bacon from the module spam
```