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

```
The Boolean data type represents a logical value that can be either true or
false
Eg: x = 2   if x > 0 then it will return True.
    y = 6   if y < 10 then it will return False.
```

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

```
The three different types of Boolean operators are:
1. AND(&)
2. OR(|)
3. NOT(=!)
```

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).

```
1. AND(&):
a = 1 & b = 0 Output = 0
a = 0 & b = 1 Output = 0
a = 0 & b = 0 Output = 0
a = 1 & b = 1 Output = 1

2. OR(|):
a = 1 | b = 0 Output = 1
a = 0 | b = 1 Output = 1
a = 1 | b = 1 Output = 1
a = 0 | b = 0 Output = 0

3. NOT(=!):
a != 1 Output = 0
a != 0 Output = 1
```

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)

In [1]:
```python
a = (5>4) and (3==5)  # True and False = False
b = not (5>4)     # not(True) = False
c = (5>4) or (3==5)  # True or False = True
d = not((5>4) or (3==5))   # not(True or False) = False
e = (True and True) and (True == False)   # True and False = False
f = (not False) or (not True)   # True or False = True
print(a,b,c,d,e,f)
```

```
False False True False False True
```

5. What are the six comparison operators?

```
The six comparison operators are:

1. Equal to (==): This operator checks if two values are equal and returns
true if they are, and false otherwise.
2. Not equal to (!=): This operator checks if two values are not equal and
returns true if they are not, and false otherwise.
3. Greater than (>): This operator checks if the left value is greater than
the right value and returns true if it is, and false otherwise.
4. Less than (<): This operator checks if the left value is less than the
right value and returns true if it is, and false otherwise.
5. Greater than or equal to (>=): This operator checks if the left value is
greater than or equal to the right value and returns true if it is, and false
otherwise.
6. Less than or equal to (<=): This operator checks if the left value is less
than or equal to the right value and returns true if it is, and false
otherwise.
```

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 (==) is used for comparison. It
checks if two values are equal and returns a Boolean value of true if they
are equal, and false otherwise.
eg: a=5,b=4
a==b
Output will be False.
Assignment (=) operator: The assignment operator (=) is used to assign a
value to a variable.
eg: x = 3
y = x + 5
print(y)
output is 8.
```

```
In short, the main difference is the equal to (==) operator is used for
comparison, while the assignment (=) operator is used to assign values to
variables.
```

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.

In [2]:

```
spam = int(input('Please enter the number:'))
if spam == 1:
    print('Hello')
elif spam == 2:
    print('Howdy')
else:
    print('Greetings!')
```

```
Please enter the number:2
Howdy
```

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

```
If your program is stuck in an endless loop you can press: Ctrl + C
```

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

```
The "break" statement is used to stop the entire loop, while the "continue"
statement is used to skip the rest of the current iteration and move to the
next iteration of the loop.
```

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

```
range(10) generates numbers from 0 to 9 (upper limit 10 excluded) with a
default start of 0 and a default step of 1.
range(0, 10) also generates numbers from 0 to 9 (upper limit 10 excluded)
with an explicit start of 0 and a default step of 1.
range(0, 10, 1) generates numbers from 0 to 9 (upper limit 10 excluded) with
an explicit start of 0 and an explicit step of 1.
All three expressions result in the same sequence of numbers: 0, 1, 2, 3, 4,
5, 6, 7, 8, 9.
```

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.

In [3]:

```python
for i in range (1,11):   # printing the numbers from 1 to 10 using a for loop
    print(i)
```

```
1
2
3
4
5
6
7
8
9
10
```

In [2]:

```python
a= 1
while a <= 10:    # printing the numbers from 1 to 10 using a while loop
    print(a)
    a += 1
```

```
1
2
3
4
5
6
7
8
9
10
```

13. If you had a function named bacon() inside a module named

```
import spam
spam.bacon()
```