**Assignment 2**

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

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

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 ).

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. What are the six comparison operators?

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

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')

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.

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

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

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

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

**Solutions :**

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

The two values of the Boolean data type are:

* **True**
* **False**

They are written as True and False in Python, with an uppercase first letter.

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

The three Boolean operators are:

* **AND**
* **OR**
* **NOT**

**3. Make a list of each Boolean operator's truth tables.**

**AND Truth Table**

| **A** | **B** | **A AND B** |
| --- | --- | --- |
| True | True | True |
| True | False | False |
| False | True | False |
| False | False | False |

**OR Truth Table**

| **A** | **B** | **A OR B** |
| --- | --- | --- |
| True | True | True |
| True | False | True |
| False | True | True |
| False | False | False |

**NOT Truth Table**

| **A** | **NOT A** |
| --- | --- |
| True | False |
| False | True |

**4. What are the values of the following expressions?**

Let's evaluate each expression one by one:

1. **(5 > 4) and (3 == 5)**
   * (5 > 4) is True
   * (3 == 5) is False
   * True and False evaluates to False
2. **not (5 > 4)**
   * (5 > 4) is True
   * not True evaluates to False
3. **(5 > 4) or (3 == 5)**
   * (5 > 4) is True
   * (3 == 5) is False
   * True or False evaluates to True
4. **not ((5 > 4) or (3 == 5))**
   * (5 > 4) or (3 == 5) is True
   * not True evaluates to False
5. **(True and True) and (True == False)**
   * True and True is True
   * (True == False) is False
   * True and False evaluates to False
6. **(not False) or (not True)**
   * not False is True
   * not True is False
   * True or False evaluates to True

**5. What are the six comparison operators?**

The six comparison operators are:

1. == (Equal to)
2. != (Not equal to)
3. < (Less than)
4. > (Greater than)
5. <= (Less than or equal to)
6. >= (Greater 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.**

* The **equal to** operator (==) checks if two values are the same. It's used in comparisons, like in conditional statements. For example, if x == 10 checks if x is equal to 10.
* The **assignment** operator (=) assigns a value to a variable. For example, x = 10 assigns the value 10 to x.

**Example Condition**:

python

Copy code

x = 5 # Assignment operator sets x to 5

if x == 5: # Equal to operator checks if x is 5

print("x is equal to 5")

**7. Identify the three blocks in this code:**

python

Copy code

spam = 0

if spam == 10:

print('eggs')

if spam > 5:

print('bacon')

else:

print('ham')

print('spam')

print('spam')

The three blocks are:

1. **The if spam == 10: block** (contains print('eggs')).
2. **The if spam > 5: block** (contains print('bacon')).
3. **The else: block** (contains print('ham')).

Each block is defined by indentation in Python.

**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.**

python

Copy code

spam = int(input("Enter the value for spam: ")) # Or set spam to a value, e.g., spam = 1

if spam == 1:

print("Hello")

elif spam == 2:

print("Howdy")

else:

print("Greetings!")

**9. If your program is stuck in an endless loop, what keys will you press?**

To stop an endless loop, you can press **Ctrl + C** on the keyboard. This interrupts the program and stops it from running.

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

* **break**: Immediately exits the loop, ending it completely.
* **continue**: Skips the current iteration and moves to the next iteration of the loop.

**Example**:

python

Copy code

for i in range(5):

if i == 2:

break # Stops the loop entirely when i is 2

print(i)

for i in range(5):

if i == 2:

continue # Skips the current iteration when i is 2

print(i)

**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.
* **range(0, 10)**: Generates numbers from 0 to 9 (same as range(10)).
* **range(0, 10, 1)**: Generates numbers from 0 to 9, incrementing by 1 each time.

All three are effectively the same here, but the third form (range(0, 10, 1)) explicitly specifies the start, stop, and step.

**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**:

python

Copy code

for i in range(1, 11):

print(i)

**Using a while loop**:

python

Copy code

i = 1

while i <= 10:

print(i)

i += 1

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

You would call it like this:

python

Copy code

import spam

spam.bacon()

Alternatively, you could import the specific function directly:

python

Copy code

from spam import bacon

bacon()