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` and `False`. They are written as `True` and `False` (note the capitalization).

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` Operator:\*\*

| A | B | A and B |

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

| True | True | True |

| True | False | False |

| False | True | False |

| False | False | False |

- \*\*`or` Operator:\*\*

| A | B | A or B |

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

| True | True | True |

| True | False | True |

| False | True | True |

| False | False | False |

- \*\*`not` Operator:\*\*

| A | not A |

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

| True | False |

| False | True |

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

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

`True and False`

Result: `False`

- `not (5 > 4)`

`not True`

Result: `False`

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

`True or False`

Result: `True`

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

`not (True or False)`

`not True`

Result: `False`

- `(True and True) and (True == False)`

`True and False`

Result: `False`

- `(not False) or (not True)`

`True or False`

Result: `True`

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

- The six comparison operators are:

- `==` (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.\*\*

- The equal to operator (`==`) is used to compare two values to check if they are equal. The assignment operator (`=`) is used to assign a value to a variable.

- Example of `==` (comparison): `if x == 10:` (checks if `x` is equal to 10)

- Example of `=` (assignment): `x = 10` (assigns the value 10 to the variable `x`)

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

```python

spam = 0

if spam == 10:

print('eggs')

if spam > 5:

print('bacon')

else:

print('ham')

print('spam')

print('spam')

```

- \*\*Block 1:\*\*

```python

spam = 0

```

- \*\*Block 2:\*\*

```python

if spam == 10:

print('eggs')

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`.\*\*

```python

spam = 1 # or any other value to test different cases

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?\*\*

- You can press `Ctrl + C` to interrupt and stop the program in most terminal or command-line interfaces.

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

- `break` terminates the nearest enclosing loop (for or while loop) and exits the loop.

- `continue` skips the remaining code inside the loop for the current iteration and jumps 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 (default start is 0, step is 1).

- `range(0, 10)` generates numbers from 0 to 9 (explicitly specifying the start as 0 and the end as 10).

- `range(0, 10, 1)` generates numbers from 0 to 9 with a step size of 1 (explicitly specifying the start, end, 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.\*\*

\*\*For loop:\*\*

```python

for i in range(1, 11):

print(i)

```

\*\*While loop:\*\*

```python

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`?\*\*

```python

import spam

spam.bacon()

```