

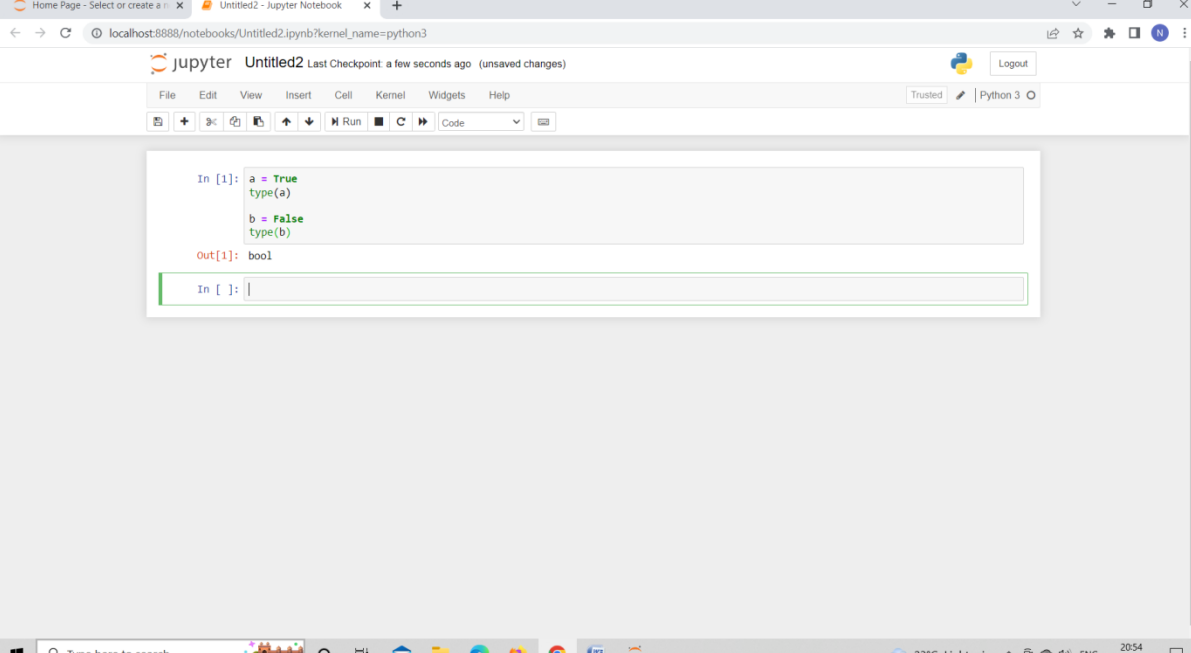
1.What are the two values of the Boolean data type? How do you write them?

SolutionThe two values of the Boolean data types are:

1)True

2)False

We write Boolean data type in python as:



The screenshot shows a Jupyter Notebook interface in a web browser. The browser address bar shows 'localhost:8888/notebooks/Untitled2.ipynb?kernel_name=python3'. The Jupyter Notebook title bar says 'jupyter Untitled2 Last Checkpoint: a few seconds ago (unsaved changes)'. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. The toolbar shows icons for file operations, running, and code execution. The code cell contains the following Python code:

```
In [1]: a = True
        type(a)
        b = False
        type(b)

Out[1]: bool

In [ ]: |
```

The output of the code cell is 'bool'. The Windows taskbar is visible at the bottom, showing the search bar, taskbar icons, and system tray with weather and date information.

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

The three different types of Boolean operators are :

1)And

2)Or

3)not

4)== (equivalent)

5)!=(not equivalent)

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 to).

Solution: Boolean OR Operator

The Boolean or operator returns True if any one of the inputs is True else returns False.

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

ii) Boolean And Operator

The Boolean and operator returns False if any one of the inputs is False else returns True.

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

iii) Boolean Not Operator

The Boolean Not operator only require one argument and returns the negation of the argument i.e. returns the True for False and False for True.

A	Not A
---	-------

True	False
------	-------

False	True
-------	------

iv) Boolean == (equivalent) and != (not equivalent) Operator

Both the operators are used to compared two results. == (equivalent operator returns True if two results are equal and != (not equivalent operator returns True if the two results are not same.

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 $(\text{True} == \text{False})$

(not False) or (not True)

Solution:

1)False

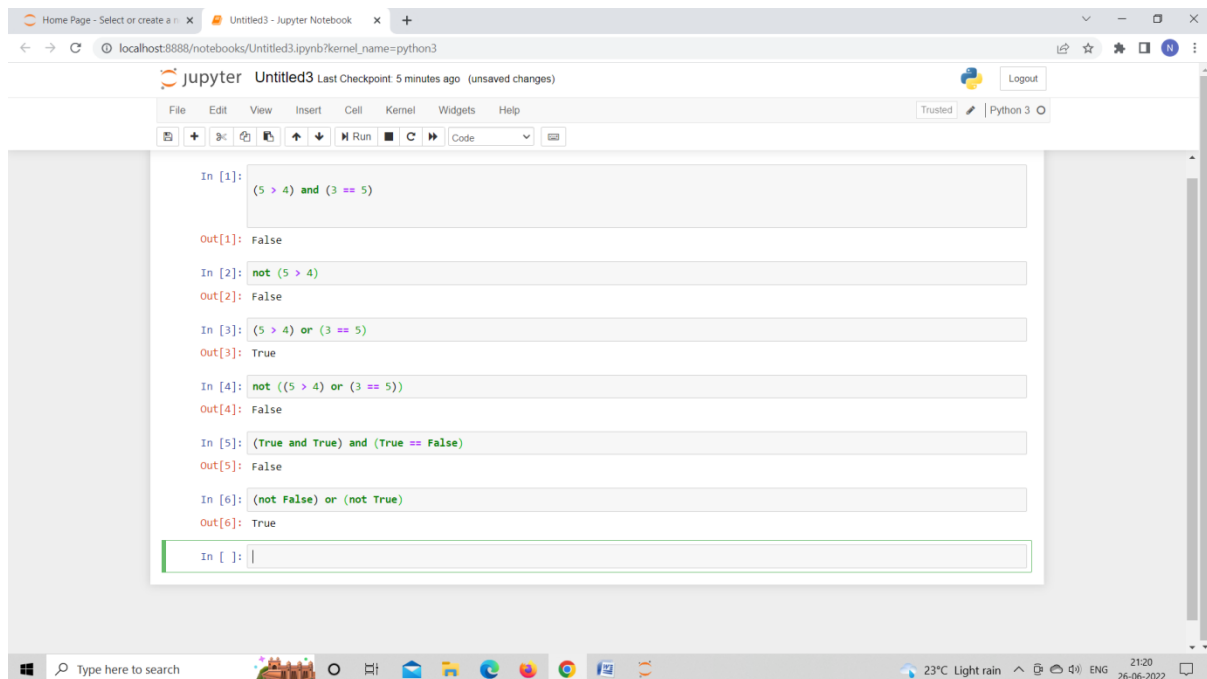
2)False

3)True

4)False

5)False

6)True



```
In [1]: (5 > 4) and (3 == 5)
Out[1]: False

In [2]: not (5 > 4)
Out[2]: False

In [3]: (5 > 4) or (3 == 5)
Out[3]: True

In [4]: not ((5 > 4) or (3 == 5))
Out[4]: False

In [5]: (True and True) and (True == False)
Out[5]: False

In [6]: (not False) or (not True)
Out[6]: True

In [ ]: 
```

5. What are the six comparison operators?

Solution:

The six Comparison operators in python are as follows:

- Less than (<)
- Less than or equal to (<=)
- Greater than (>)
- Greater than or equal to (>=)
- Equal to (==)
- Not 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.

Solution: The main difference between equal to and assignment operators are and when to use them are:

Assignment Operator (=)

= is an Assignment Operator in C, C++, and python and other programming languages, It is Binary Operator which operates on two operands.

= assigns the value of right side expression's or variable's value to the left side variable.

Equal To Operator (==)

== is an Equal To Operator in C and C++ only, It is Binary Operator which operates on two operands.

== compares value of left and side expressions, return 1 if they are equal other will it will return 0.

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')
```

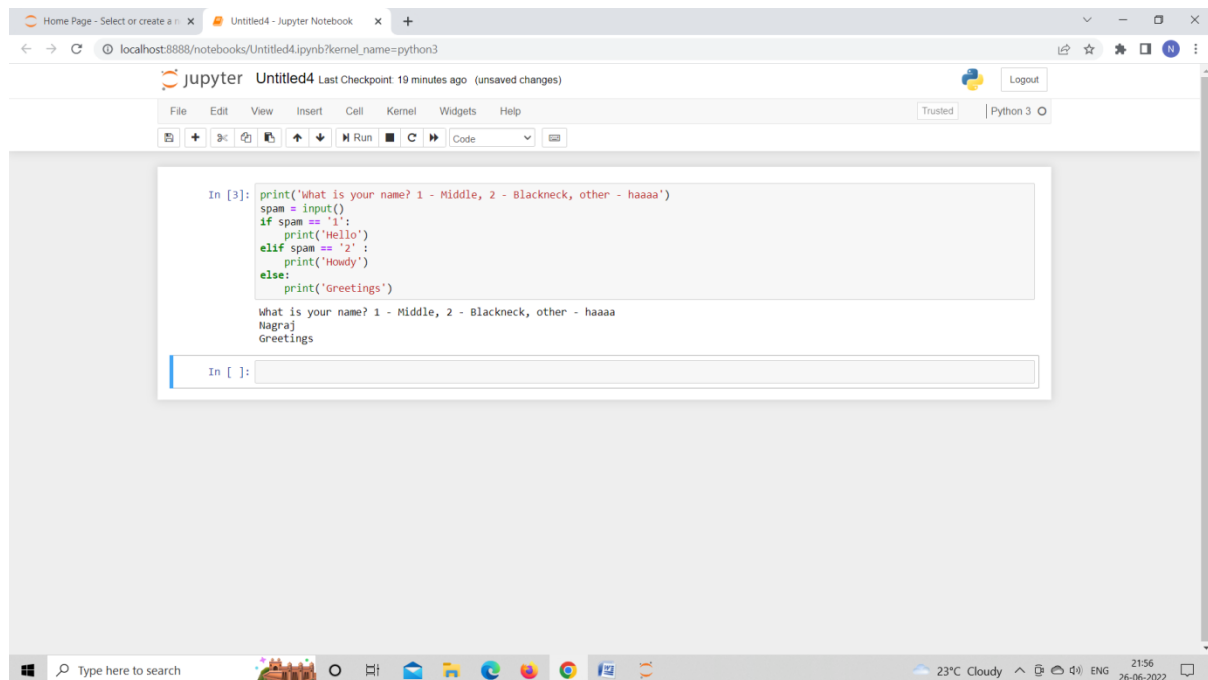
Solution:

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

Solution:



```
In [3]: print('what is your name? 1 - Middle, 2 - Blackneck, other - haaaa')
spam = input()
if spam == '1':
    print('Hello')
elif spam == '2':
    print('Howdy')
else:
    print('Greetings')

what is your name? 1 - Middle, 2 - Blackneck, other - haaaa
Nagraj
Greetings

In [ ]:
```

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

Solution:

If your Program is stuck in an endless loop. We are going to press Ctrl + C where C stands for cancel

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

Solution:

Break Statement

Continue Statement

The Break statement is used to exit from the The continue statement is not used to exit

Break Statement

loop constructs.

The break statement is usually used with the switch statement, and it can also use it within the while loop, do-while loop, or the for-loop.

When a break statement is encountered then the control is exited from the loop construct immediately.

Syntax:

break;

Continue Statement

from the loop constructs.

The continue statement is not used with the switch statement, but it can be used within the while loop, do-while loop, or for-loop.

When the continue statement is encountered then the control automatically passed from the beginning of the loop statement.

Syntax:

continue;

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

Solution:

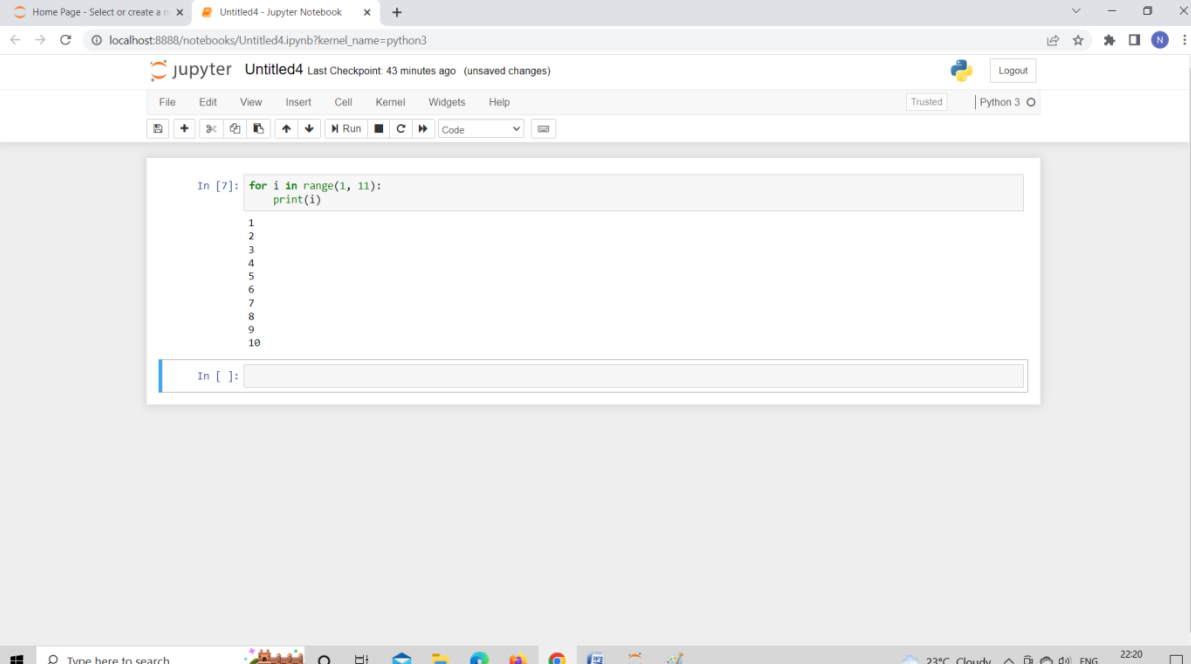
range(10)[range(stop) takes only one argument

range(0,10)[range(start,stop) takes two arguments

range(0,10,1)[range(start,stop,step) takes three arguments

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.

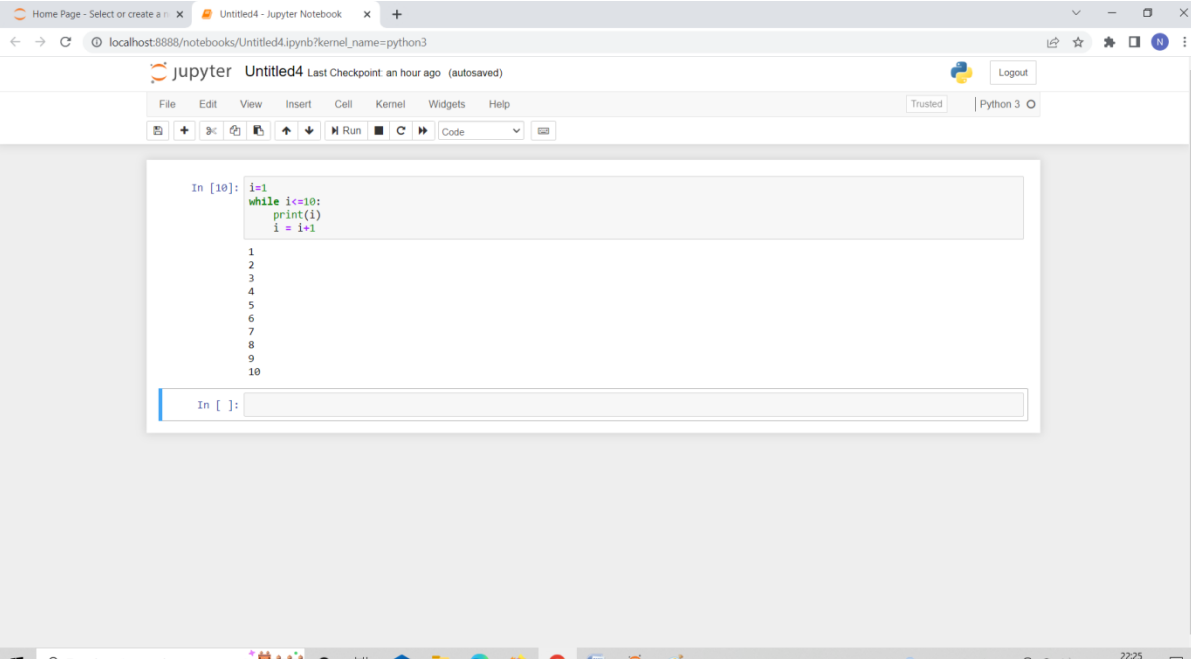
Solution:



The screenshot shows a Jupyter Notebook interface with a single code cell. The code cell contains a for loop that prints numbers from 1 to 10. The output of the code cell shows the numbers 1 through 10, each on a new line. The Jupyter Notebook interface includes a menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The status bar at the bottom shows the temperature as 23°C, the weather as Cloudy, and the time as 22:20 on 26-06-2022.

```
In [7]: for i in range(1, 11):  
        print(i)  
  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

Using while loop:



The screenshot shows a Jupyter Notebook interface with a single code cell. The code cell contains a while loop that prints numbers from 1 to 10. The output of the code cell shows the numbers 1 through 10, each on a new line. The Jupyter Notebook interface includes a menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The status bar at the bottom shows the temperature as 22°C, the weather as Cloudy, and the time as 22:25 on 26-06-2022.

```
In [10]: i=1  
         while i<=10:  
             print(i)  
             i = i+1  
  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

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

Solution: This function can be called with `spam.bacon()`.