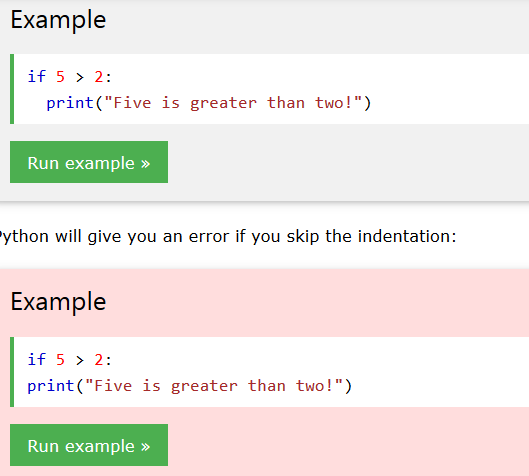
* **Python Indentations**

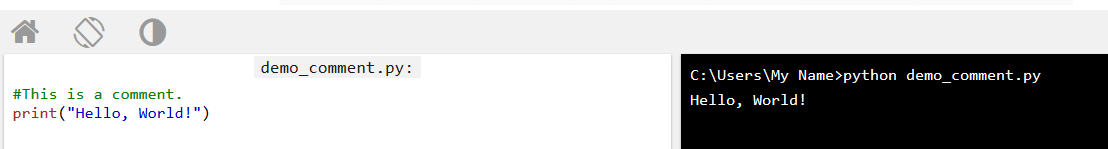
Means condition and print line borabore hoya jbe na, example 1 run korbe but example 2 error dekaba

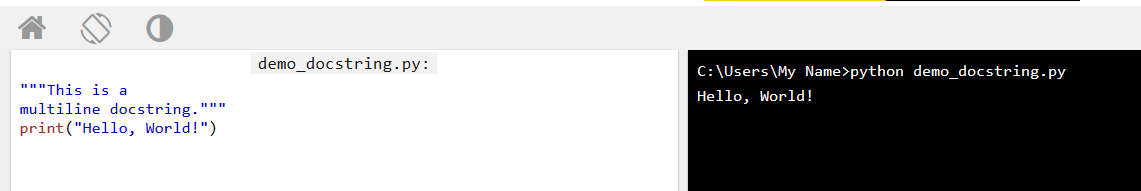
****

## Comment

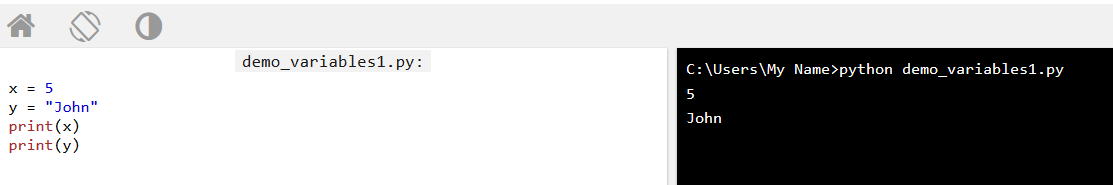
1.# this is comment

2.””” This multi line comment”””

****

****

## Creating Variables

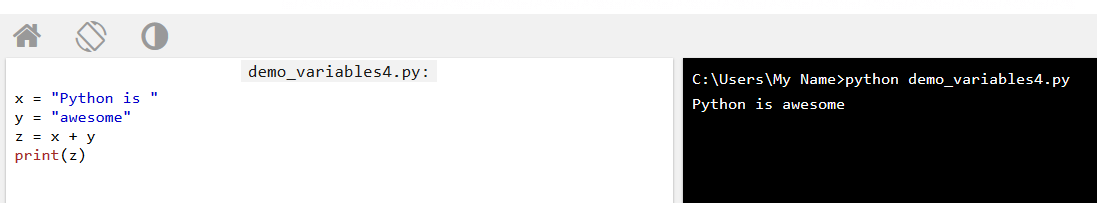


## Output Variables

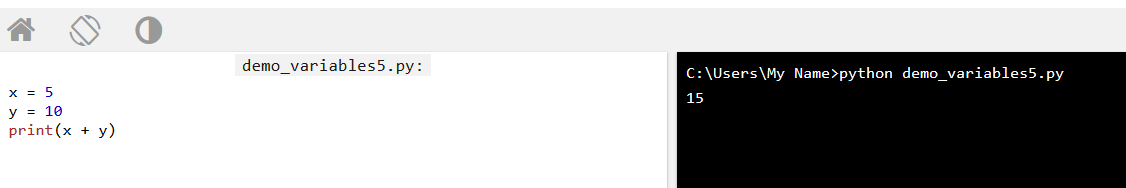
Print(x)

## 

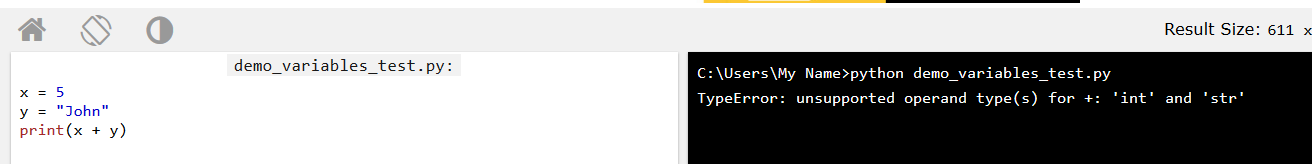
You can also use the + character to add a variable to another variable:



For numbers, the + character works as a mathematical operator:



If you try to combine a string and a number, Python will give you an error



## Python Numbers

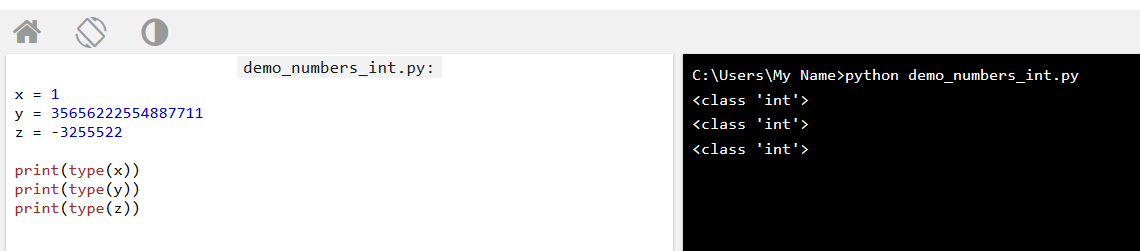
There are three numeric types in Python:

* int
* float
* complex



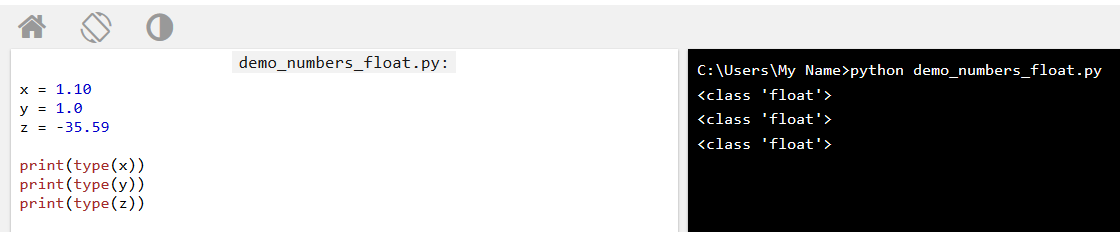
## Int

Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length.

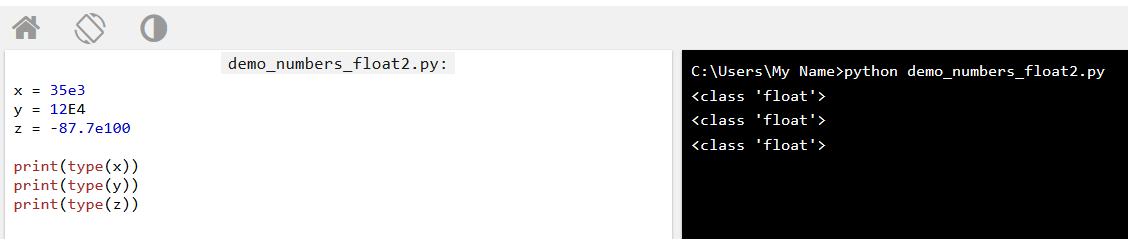


## Float

Float, or "floating point number" is a number, positive or negative, containing one or more decimals.

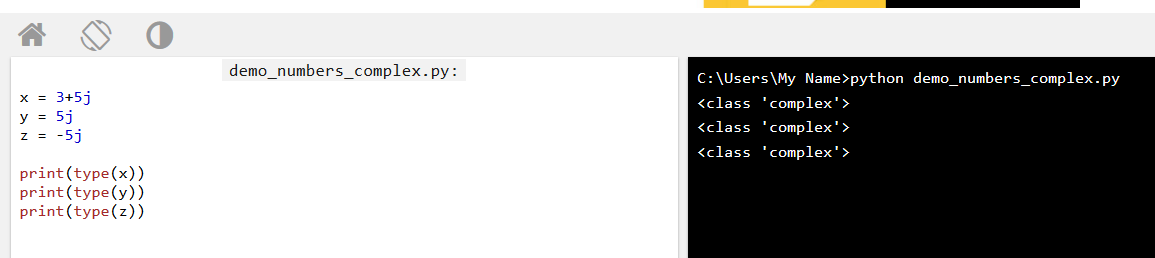


Float can also be scientific numbers with an "e" to indicate the power of 10.



## Complex

Complex numbers are written with a "j" as the imaginary part:

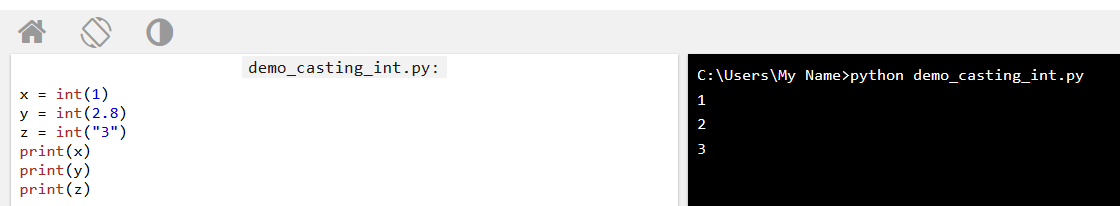


## Specify a Variable Type

Means ami j type define kore debo sey type er output value debe, a k “python casting” bole

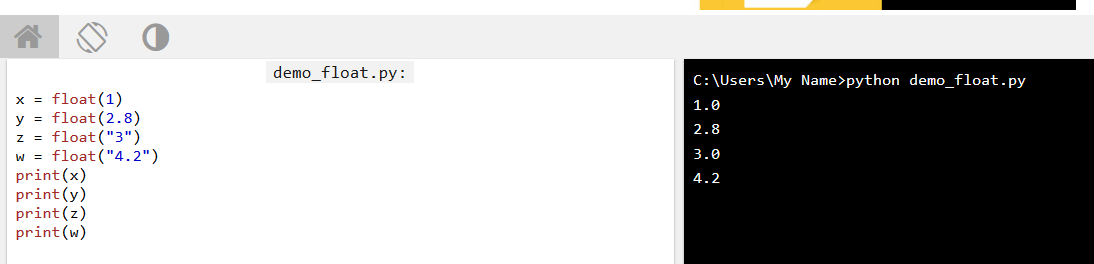
### Example

Integers:



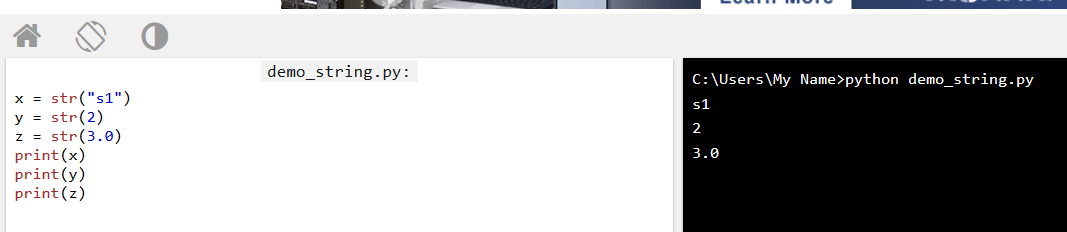
### Example

Floats:



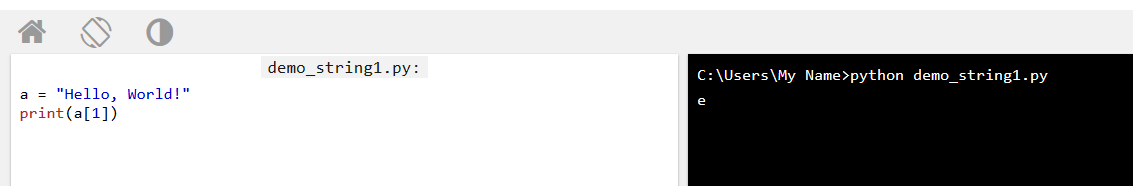
### Example

Strings:

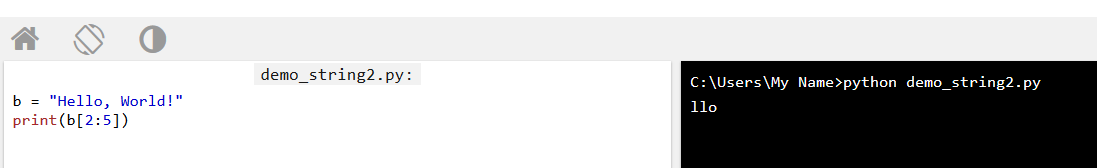


# Python Strings

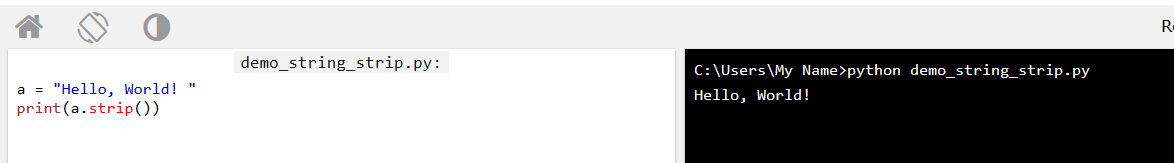
The character at position 1 (remember that the first character has the position 0):



Substring. Get the characters from position 2 to position 5 (not included): means 2 teke 5 number cherecter gulo output a show korbe.



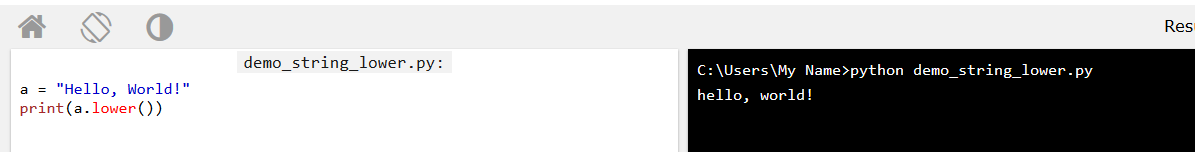
The strip() method removes any whitespace from the beginning or the end:



The len() method returns the length of a string: means string a total koyta charterer ase with zero cherecter .



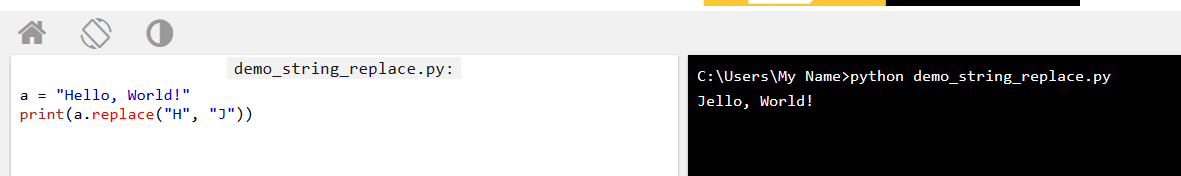
The lower() method returns the string in lower case: means capital latter teke small latter output show



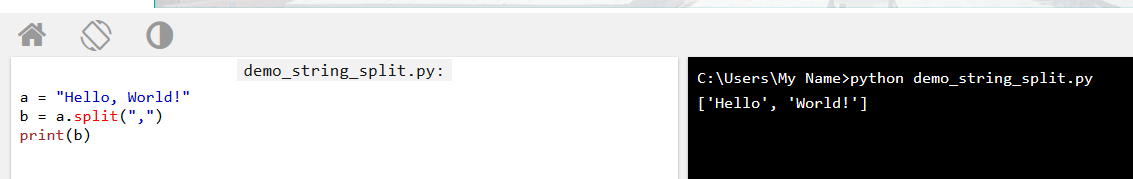
The upper() method returns the string in upper case: means small to capital latter



The replace() method replaces a string with another string: means j latter ti onno latter deya replace korbo



The split() method splits the string into substrings if it finds instances of the separator: means 3rd braket er mode separate kore debe.

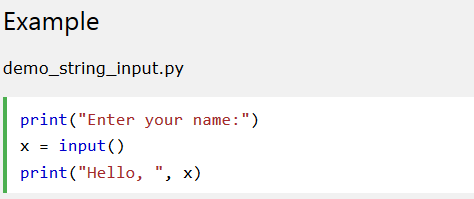


## Command-line String Input

Python allows for command line input.

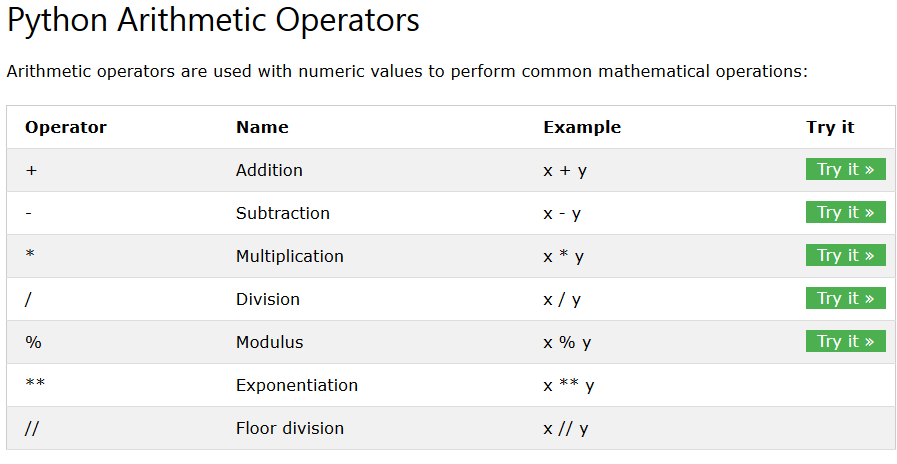
That means we are able to ask the user for input.

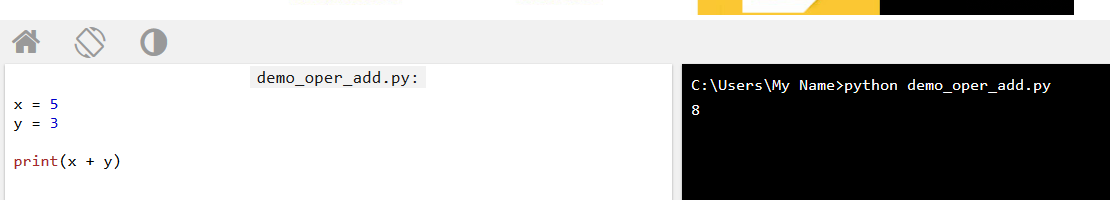
The following example asks for the user's name, then, by using the input() method, the program prints the name to the screen:



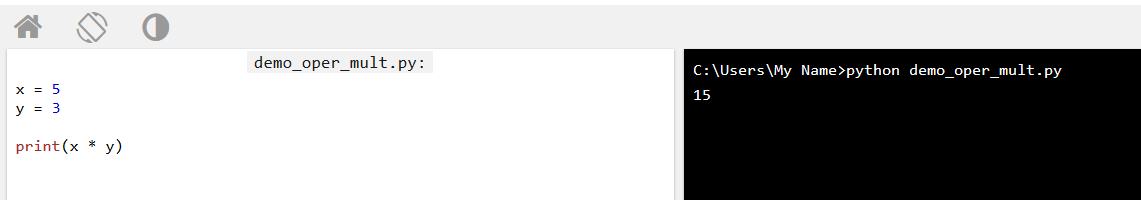
## Python Operators

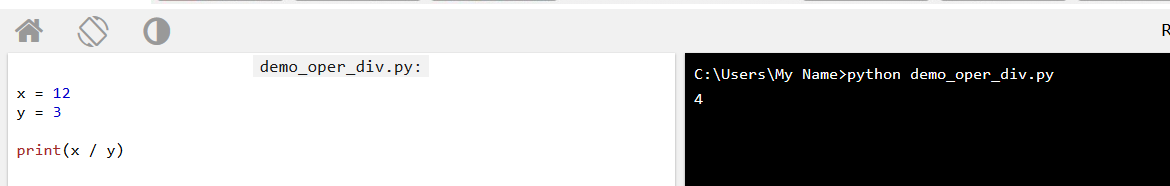
* Arithmetic operators
* Assignment operators
* Comparison operators
* Logical operators
* Identity operators
* Membership operators
* Bitwise operators

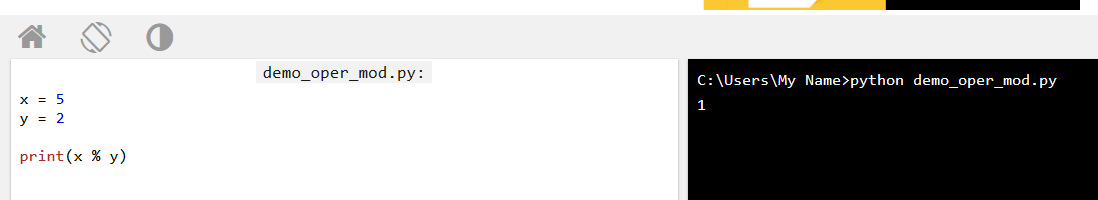


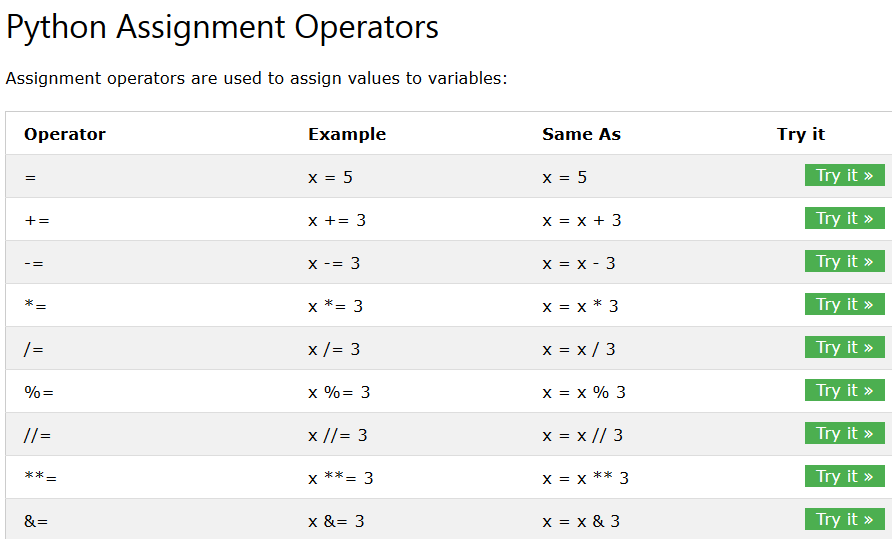


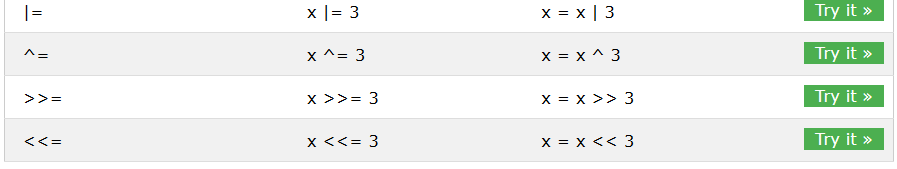


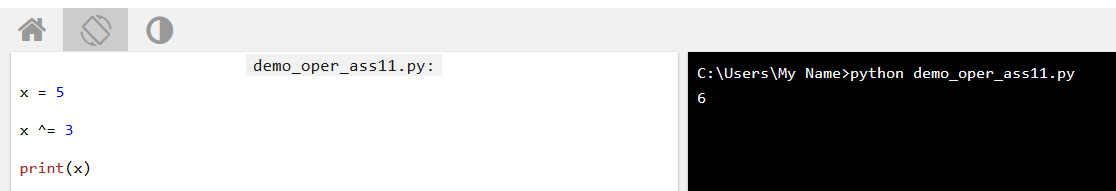
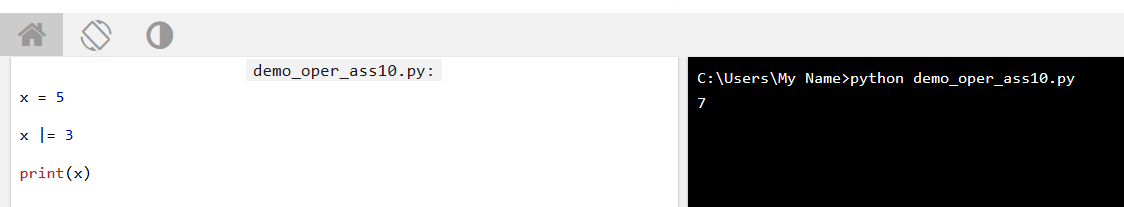
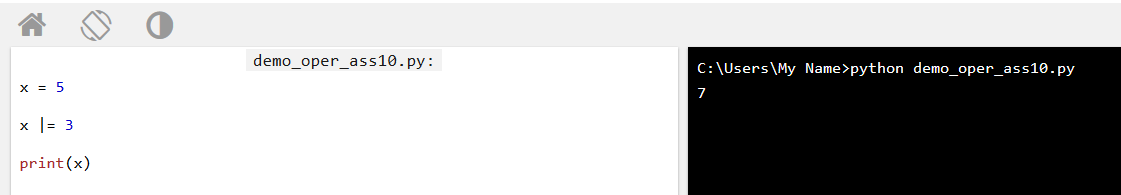
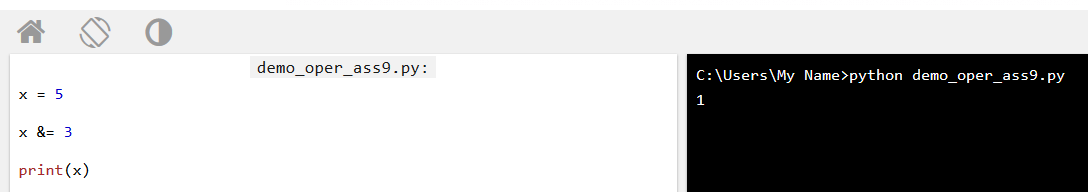
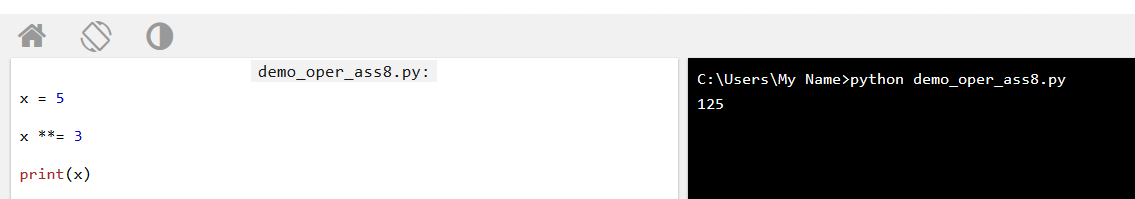
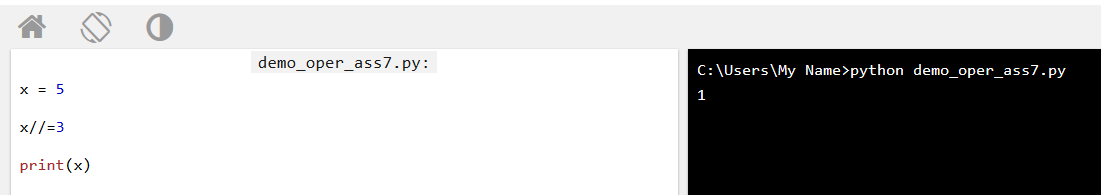
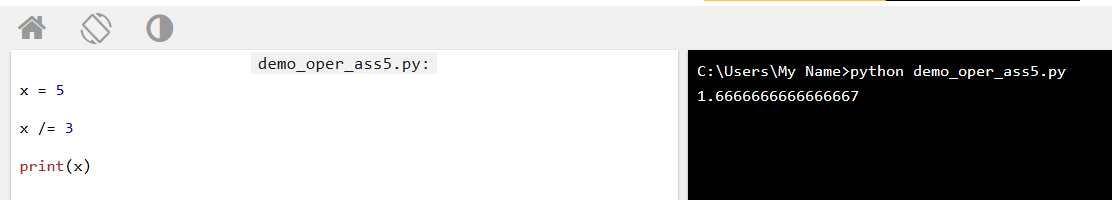
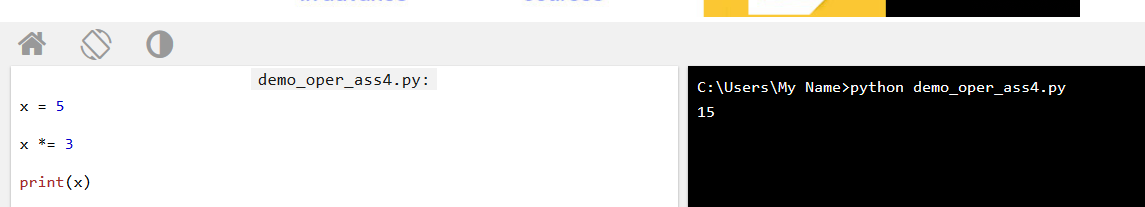
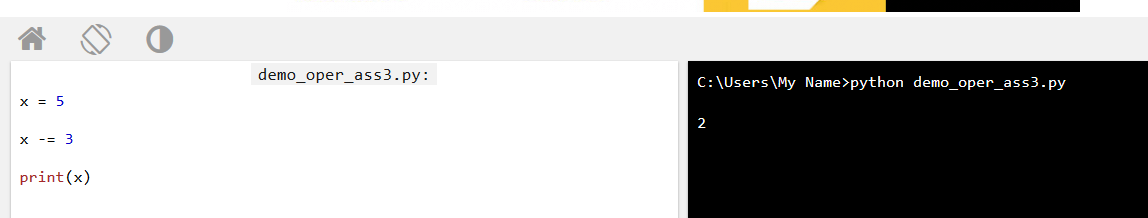


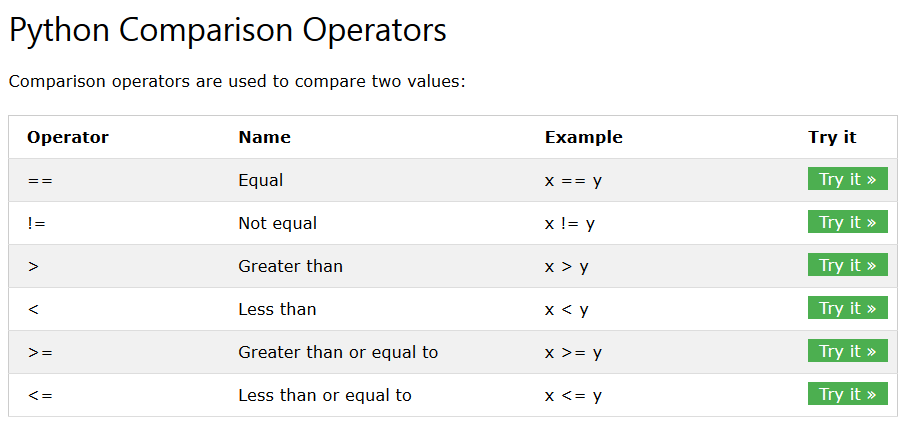


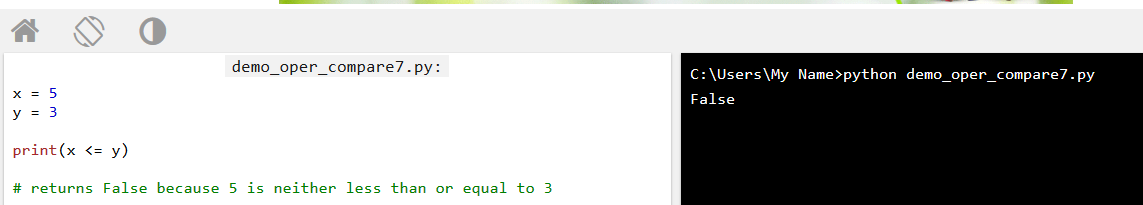
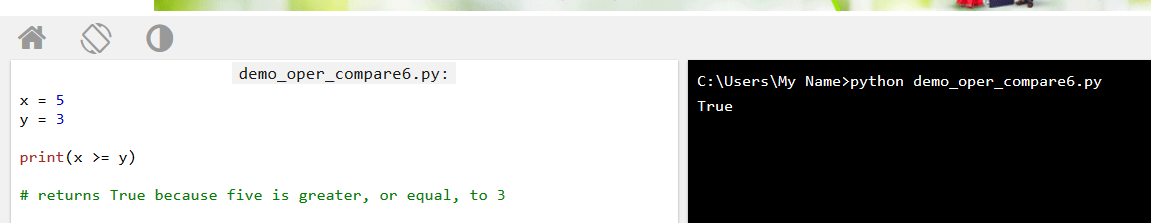
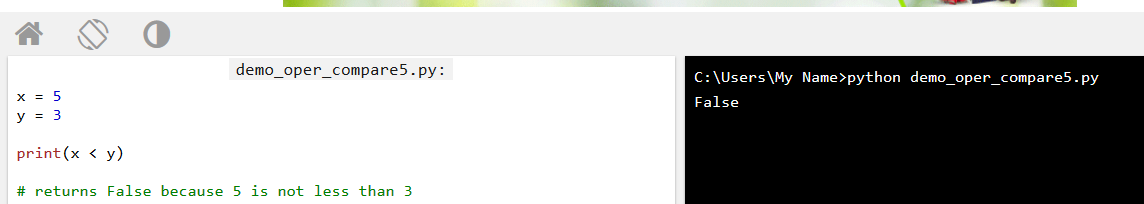
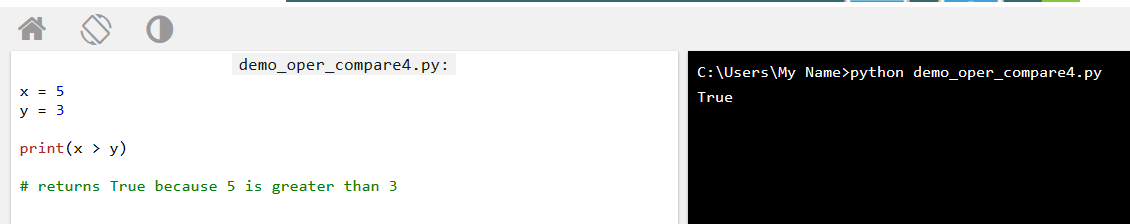
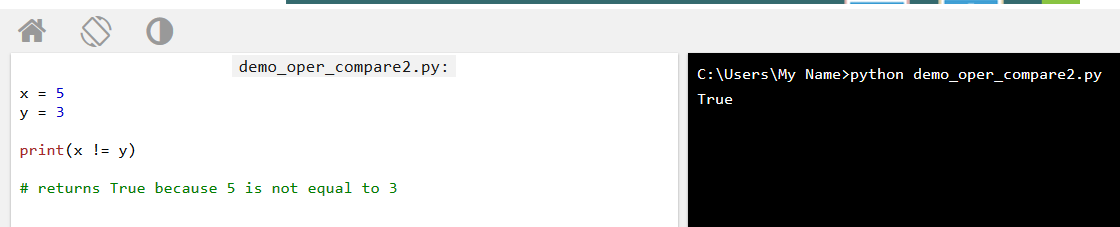
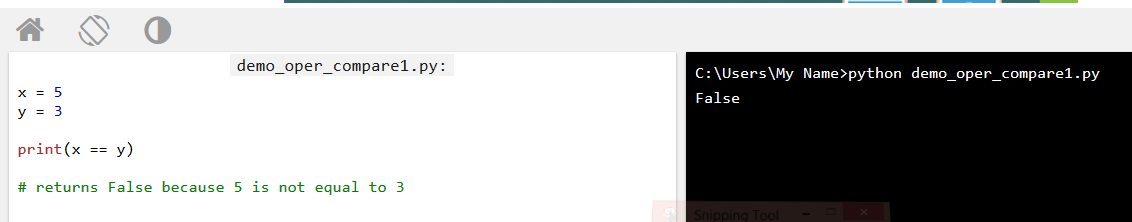


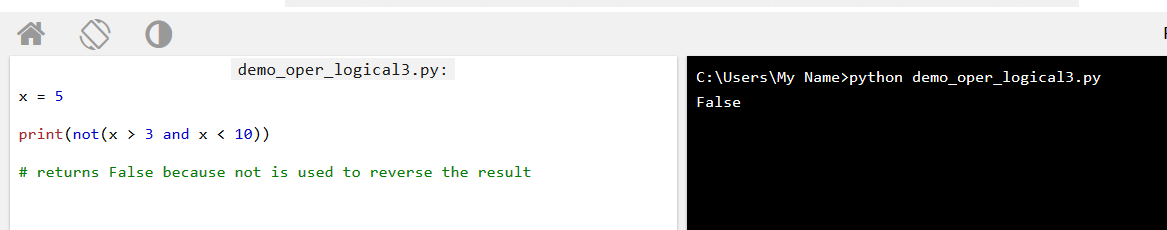
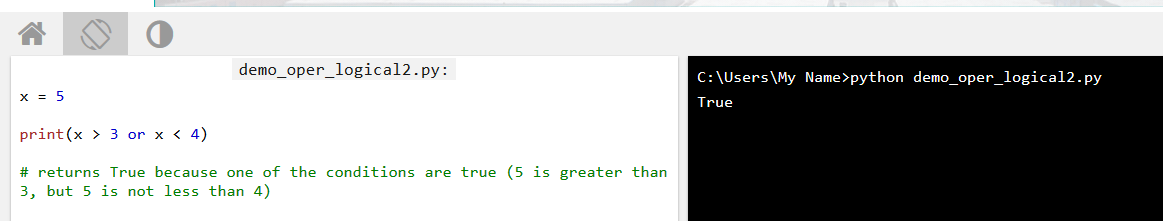
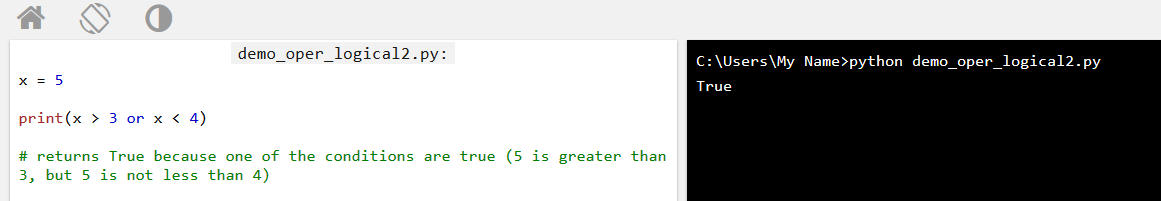
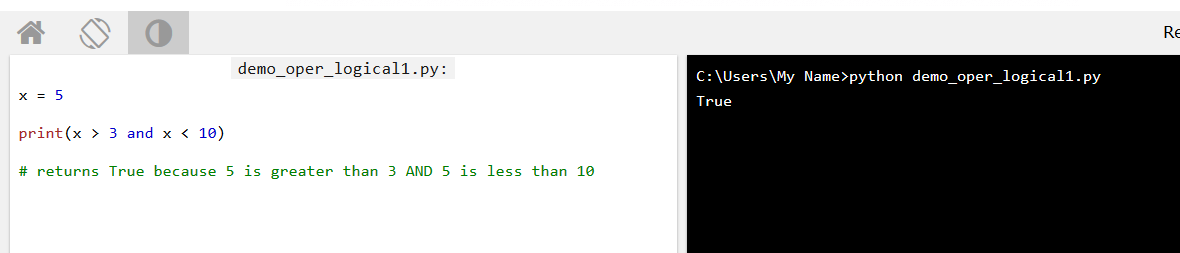


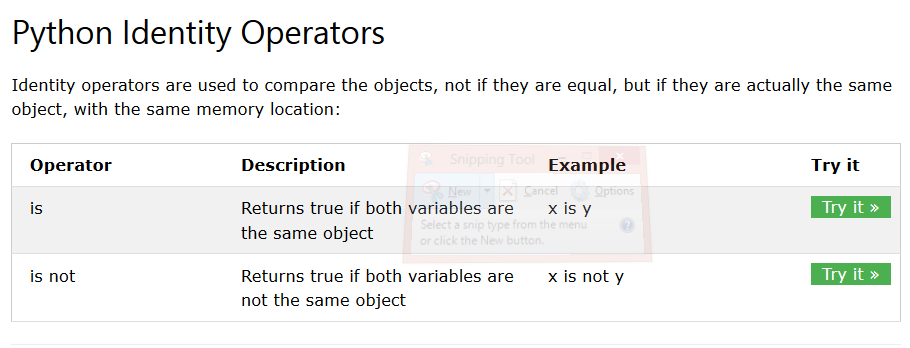


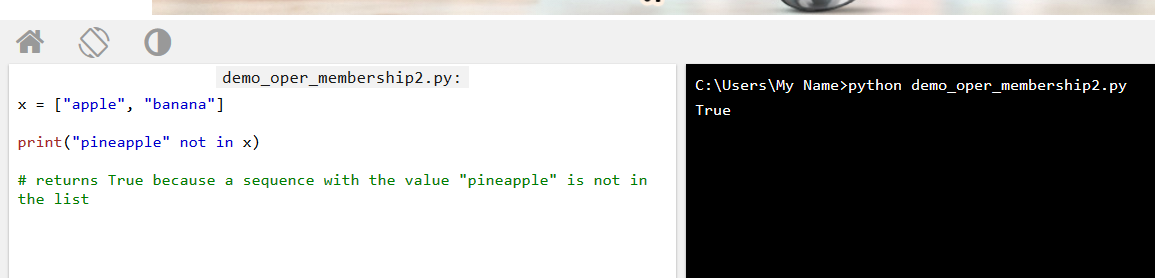
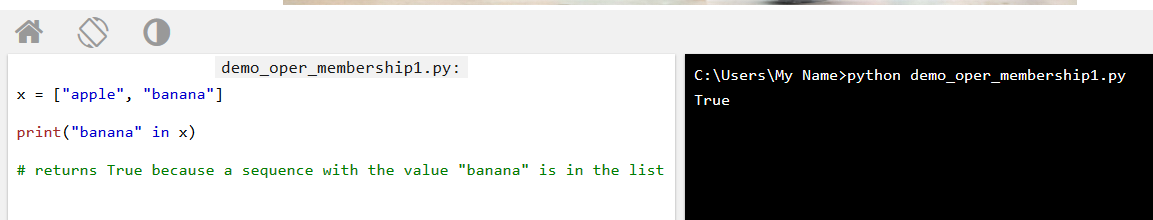
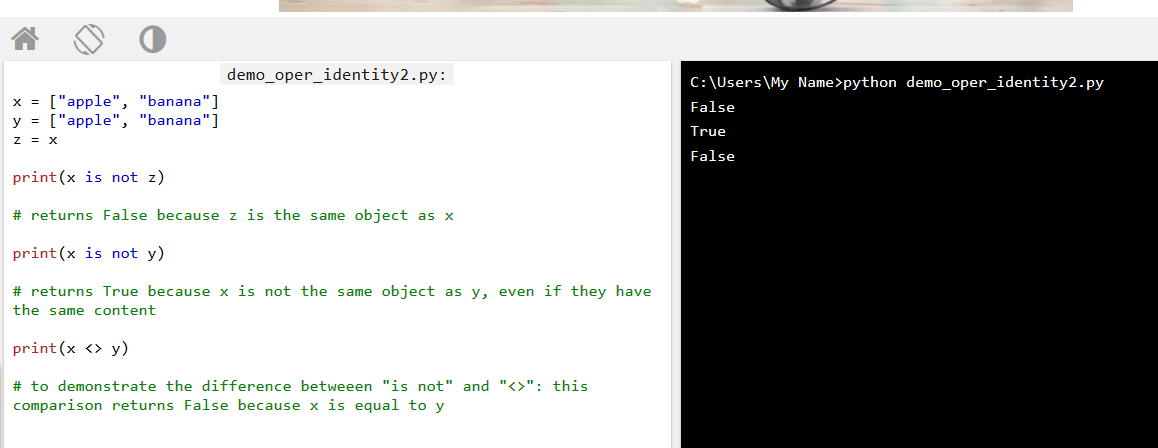
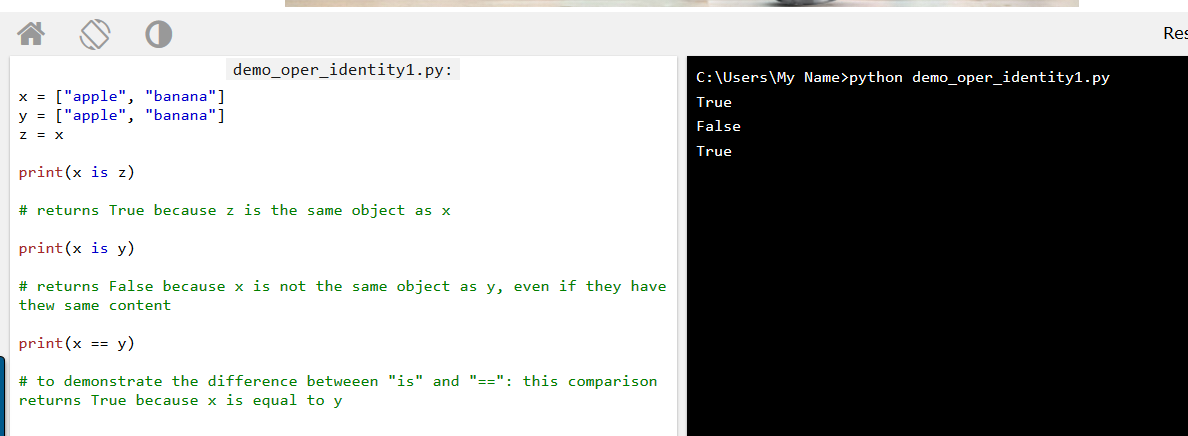


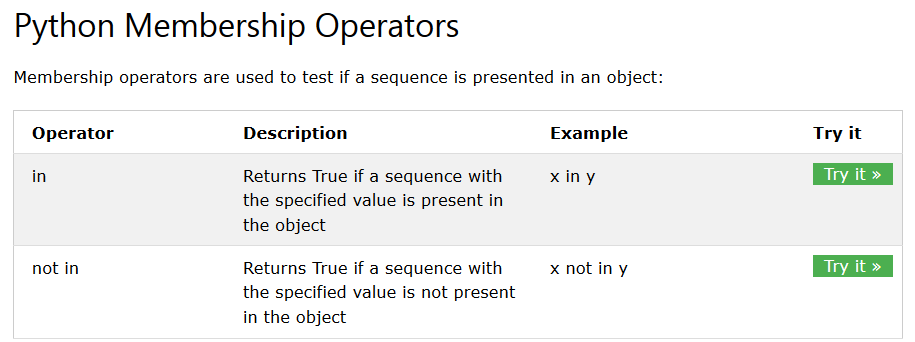


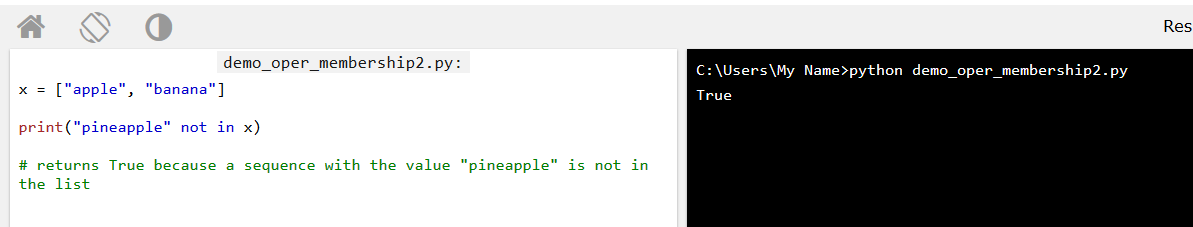
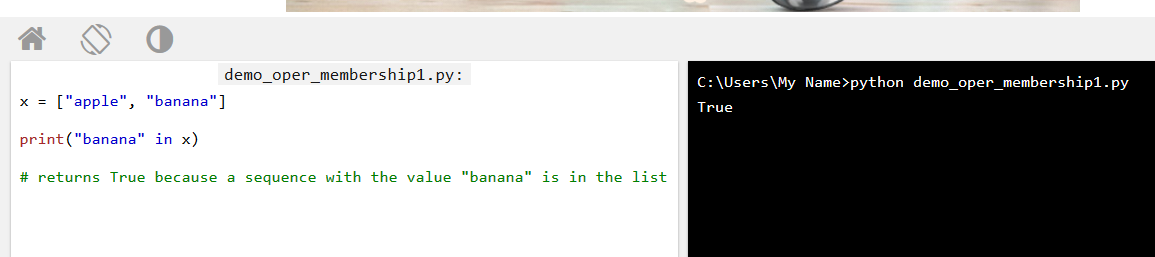


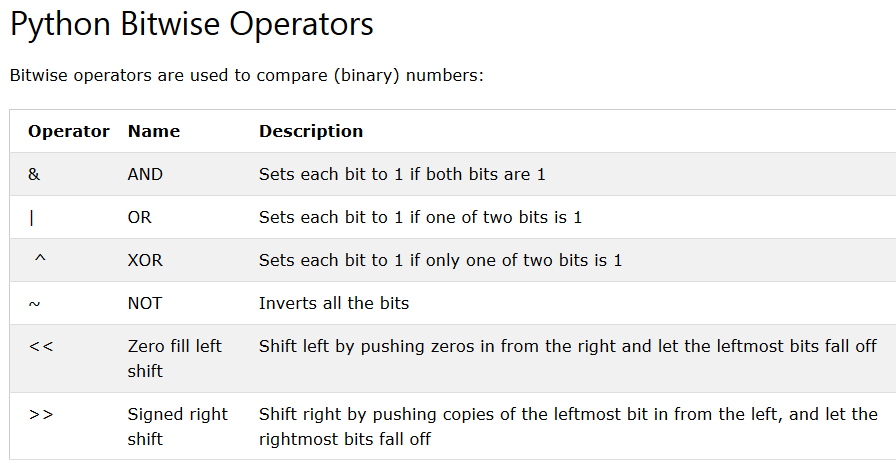






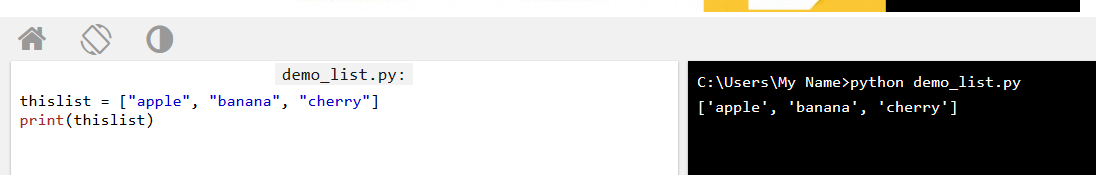






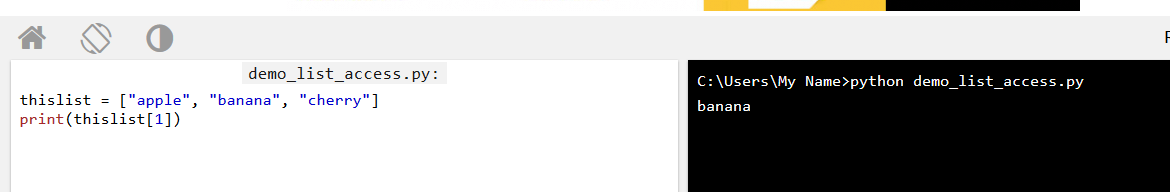
## List

A list is a collection which is ordered and changeable. In Python lists are written with square brackets.



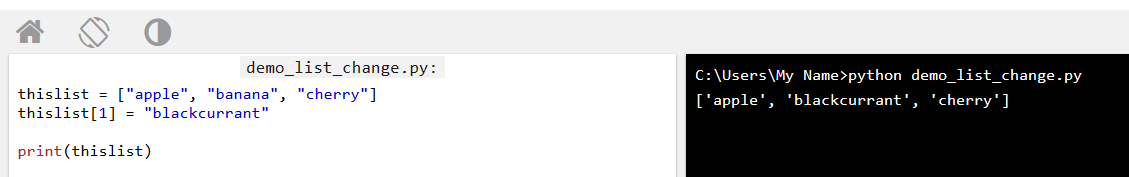
## Access Items

You access the list items by referring to the index number: means index er joto number element select/output a dekbo



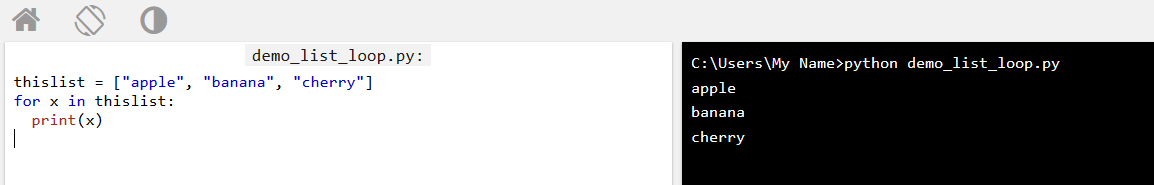
## Change Item Value

To change the value of a specific item, refer to the index number:



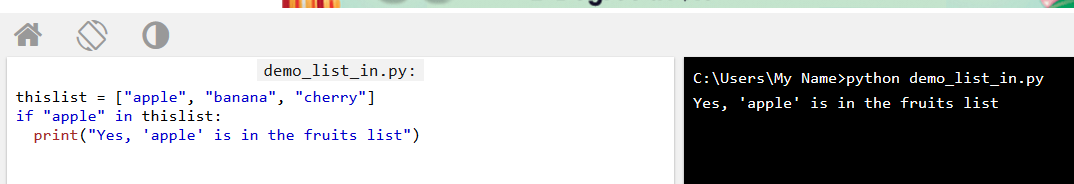
## Loop Through a List

You can loop through the list items by using a for loop: er fole index element gulo list a output kore



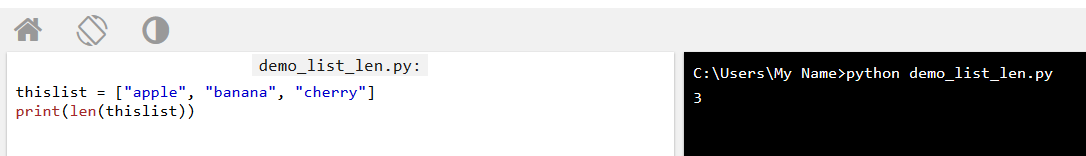
## Check if Item Exists

To determine if a specified item is present in a list use the in keyword: means list a select item ase ki?



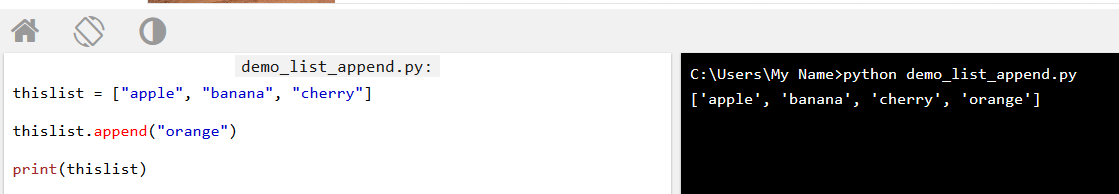
## List Length

To determine how many items a list has, use the len() method. Means index item er total element koto

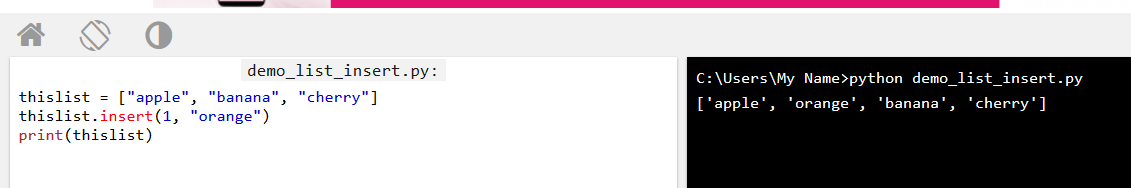


## Add Items

To add an item to the end of the list, use the append() method:

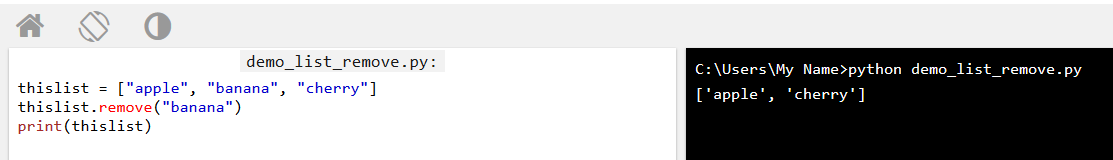


To add an item at the specified index, use the insert() method:

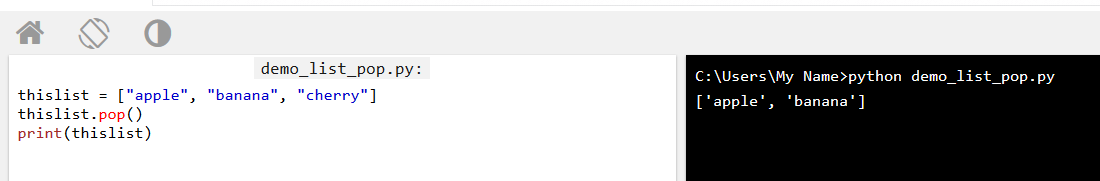


## Remove Item

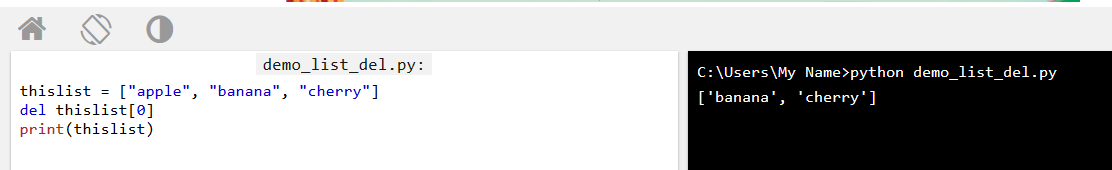
The remove() method removes the specified item:



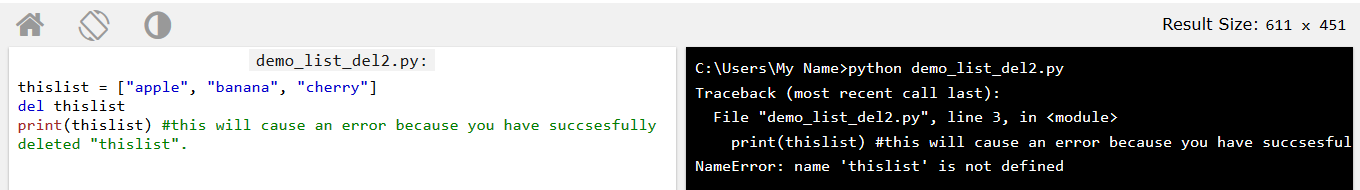
The pop() method removes the specified index, (or the last item if index is not specified):



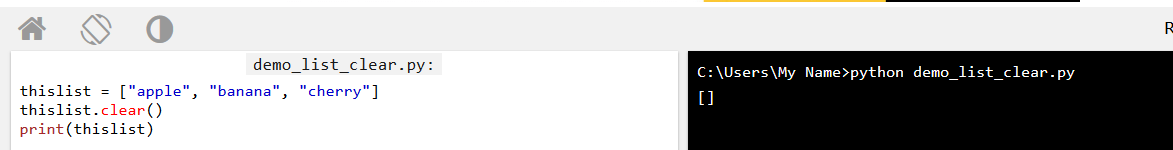
The del keyword removes the specified index:



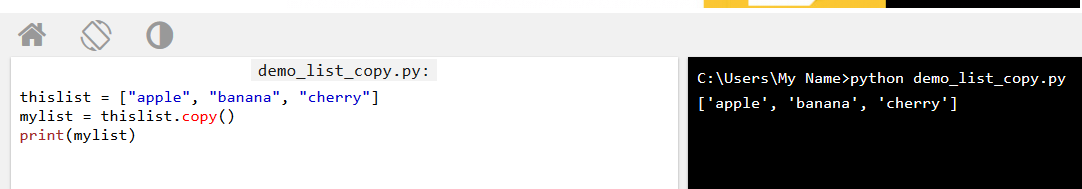
The del keyword can also delete the list completely:



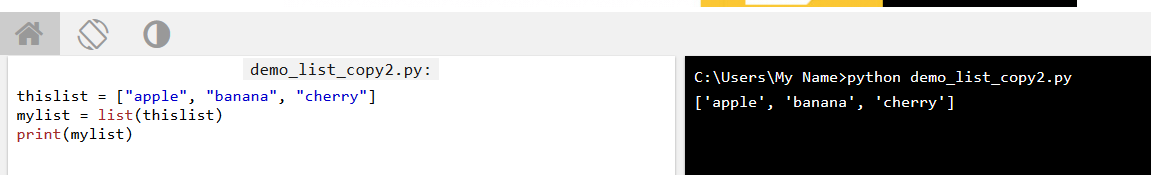
The clear() method empties the list:



## Copy a List

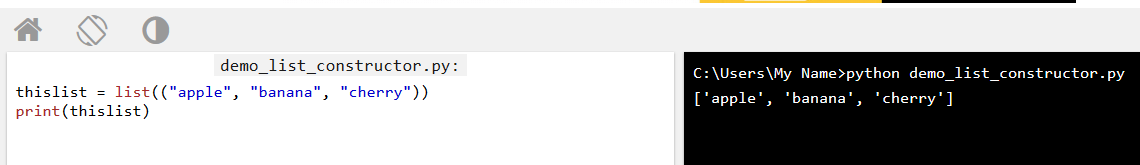


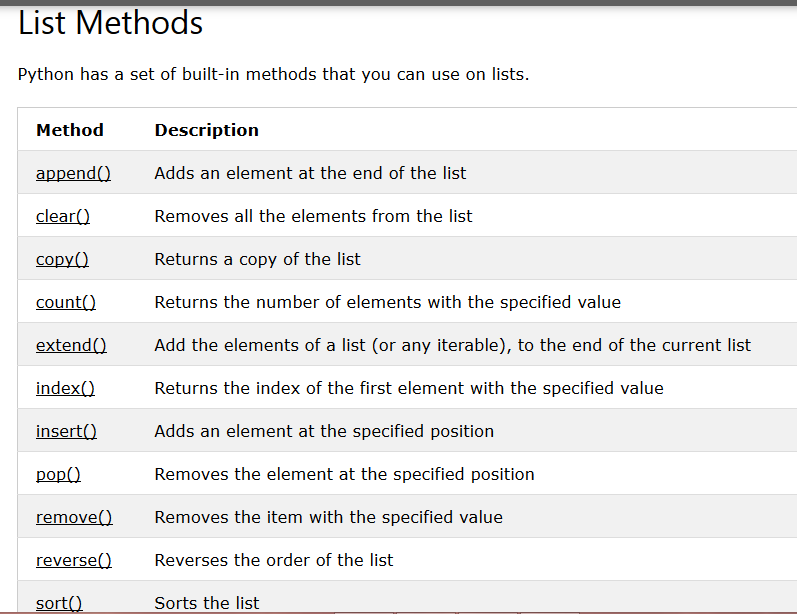
Make a copy of a list with the list() method:



## The list() Constructor

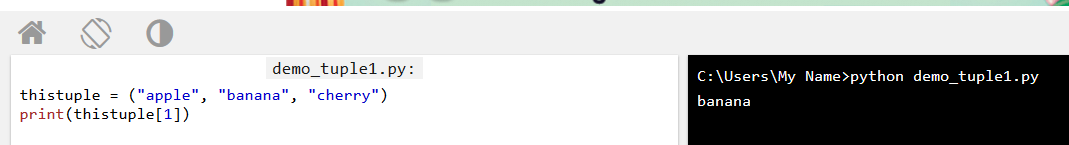
It is also possible to use the list() constructor to make a new list.





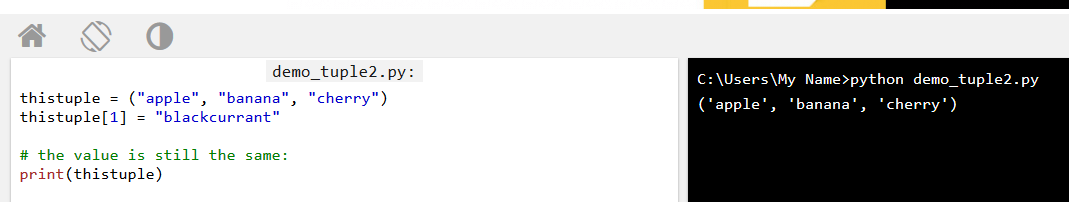
## Access Tuple Items

You can access tuple items by referring to the index number, inside square brackets:



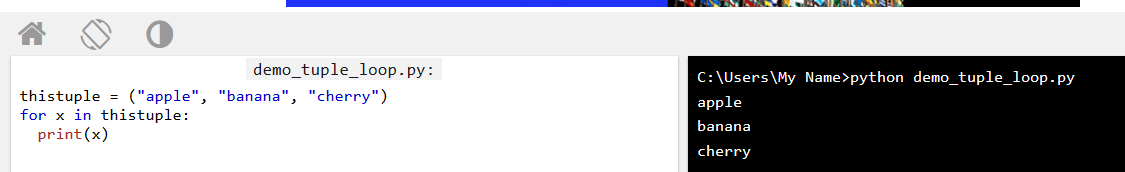
## Change Tuple Values

Once a tuple is created, you cannot change its values. Tuples are **unchangeable**.



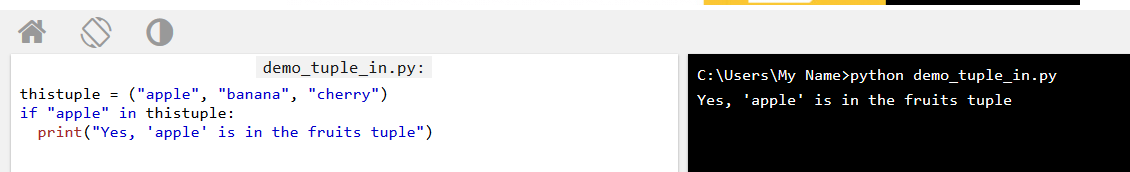
## Loop Through a Tuple

You can loop through the tuple items by using a for loop.



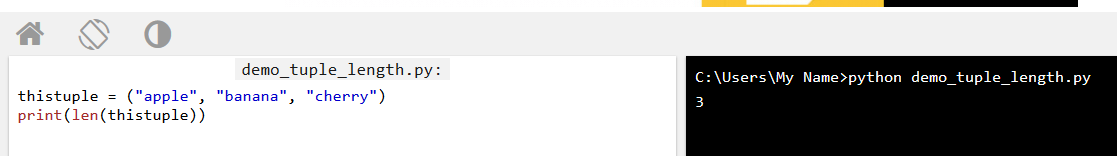
## Check if Item Exists

To determine if a specified item is present in a tuple use the in keyword:



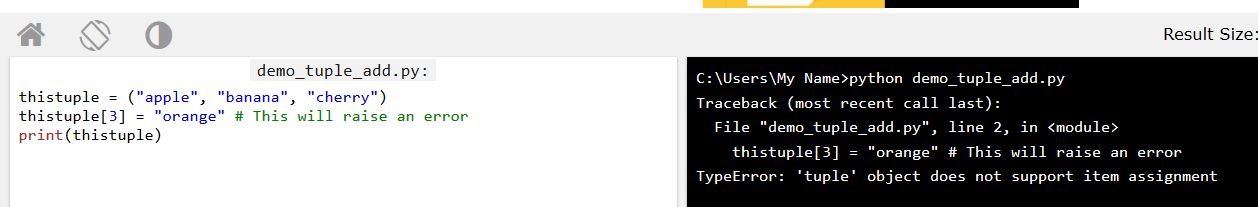
## Tuple Length

To determine how many items a tuple has, use the len() method:



## Add Items

Once a tuple is created, you cannot add items to it. Tuples are **unchangeable**.



## Remove Items

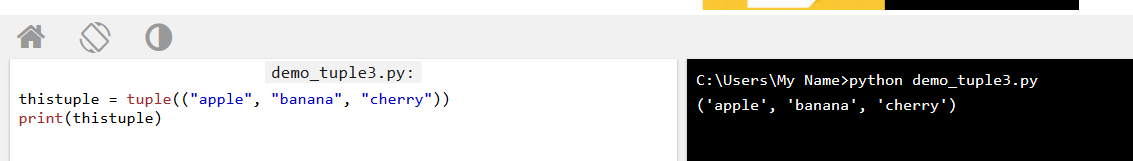
**Note:** You cannot remove items in a tuple.

 Tuples are **unchangeable**, so you cannot remove items from it, but you can delete the tuple completely:



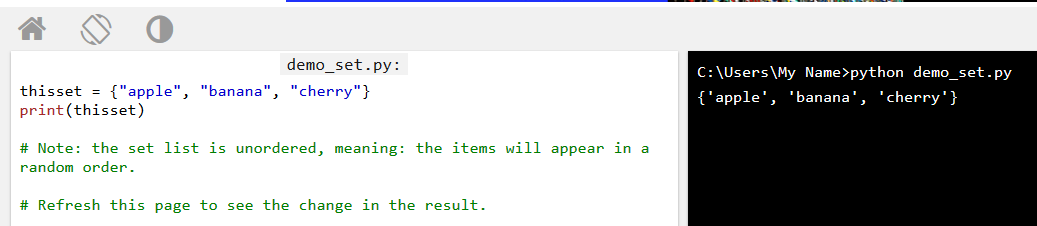
## The tuple() Constructor

It is also possible to use the tuple() constructor to make a tuple.



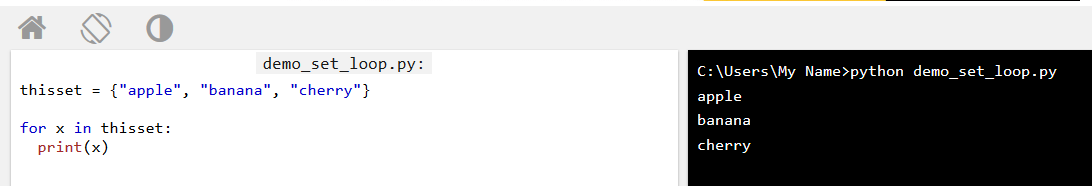
## Set

A set is a collection which is unordered and unindexed. In Python sets are written with curly brackets

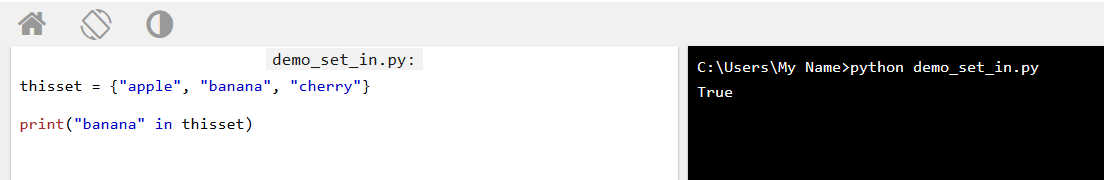
**Note:** Sets are unordered, so the items will appear in a random order.  


## Access Items

You cannot access items in a set by referring to an index, since sets are unordered the items has no index. But you can loop through the set items using a for loop, or ask if a specified value is present in a set, by using the in keyword.



Check if "banana" is present in the set:



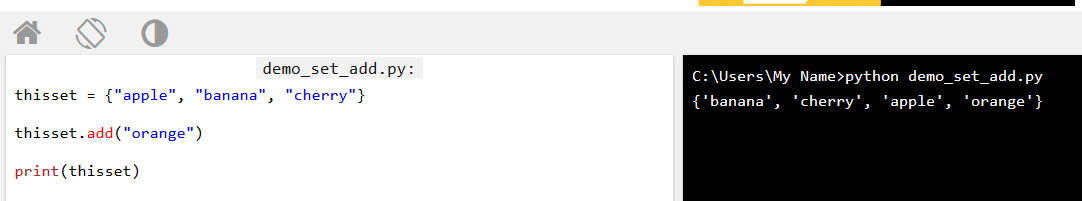
## Change Items

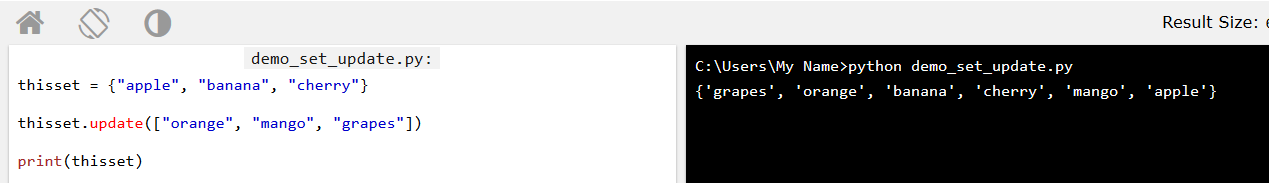
Once a set is created, you cannot change its items, but you can add new items.

## Add Items

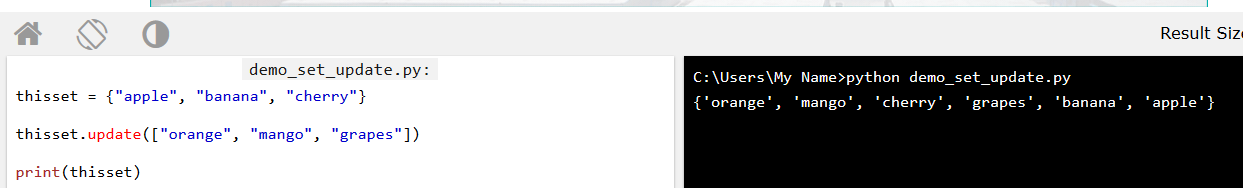
To add one item to a set use the add() method.

To add more than one item to a set use the update() method.



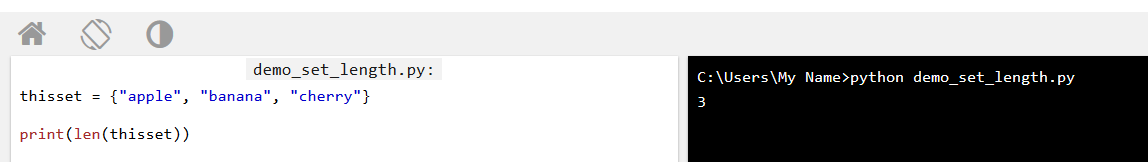


Add multiple items to a set, using the update() method:



## Get the Length of a Set

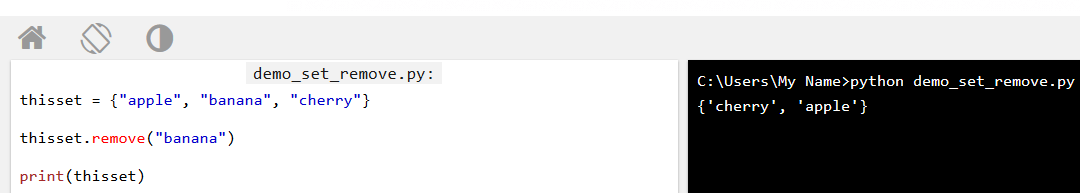
To determine how many items a set has, use the len() method.



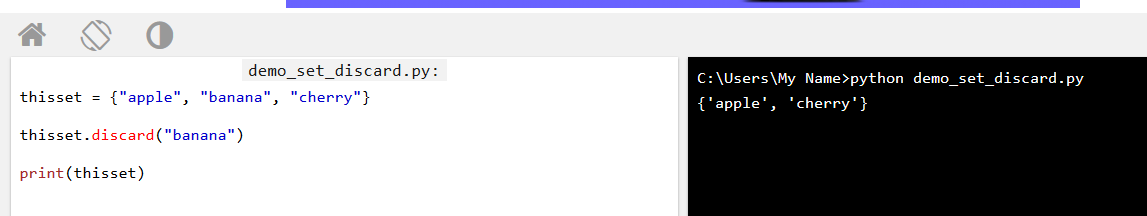
## Remove Item

To remove an item in a set, use the remove(), or the discard() method.

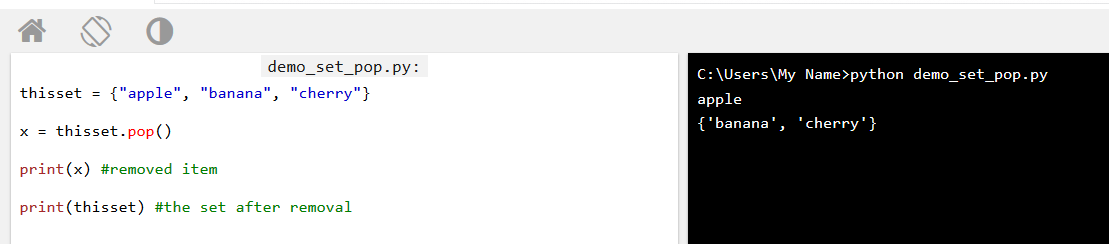
**Note:** If the item to remove does not exist, remove() will raise an error.



Remove "banana" by using the discard() method:



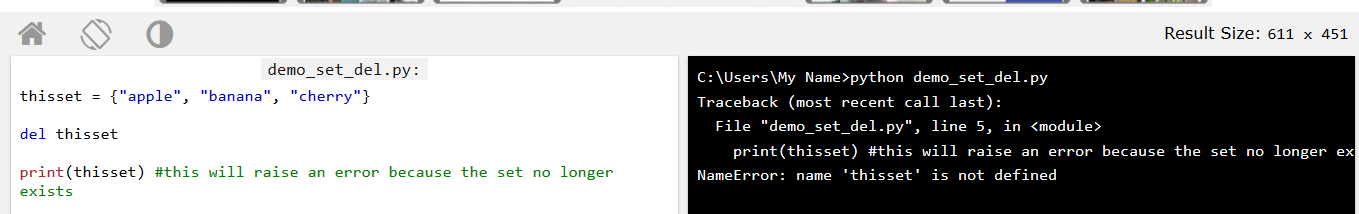
Remove the last item by using the pop() method: **Note:** Sets are unordered, so when using the pop() method, you will not know which item that gets removed.



The clear() method empties the set:

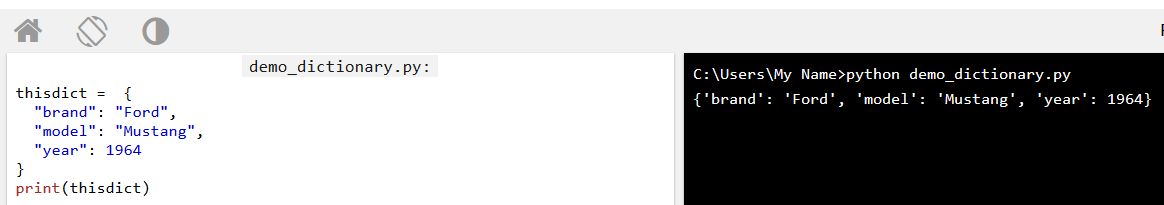


The del keyword will delete the set completely:



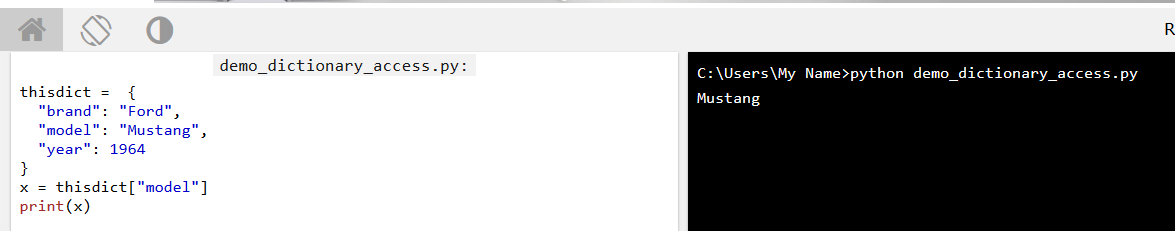
## Dictionary

A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets, and they have keys and values.

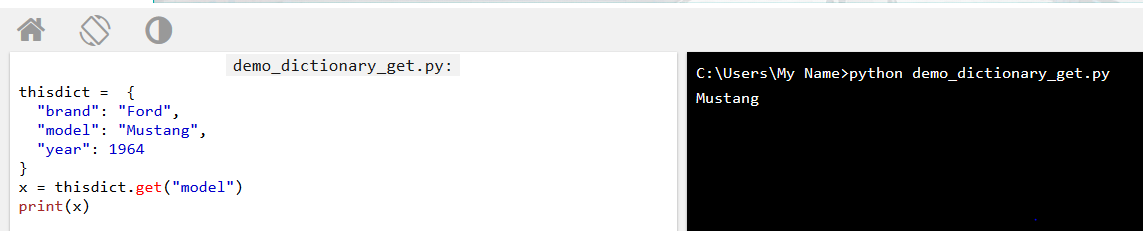


## Accessing Items

You can access the items of a dictionary by referring to its key name, inside square brackets:

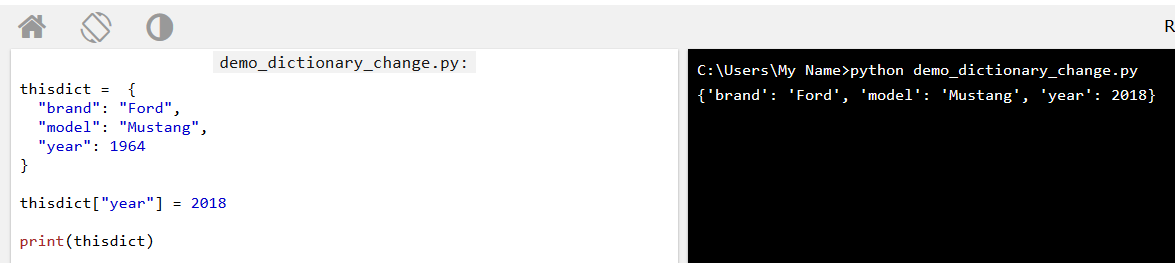


There is also a method called get() that will give you the same result:



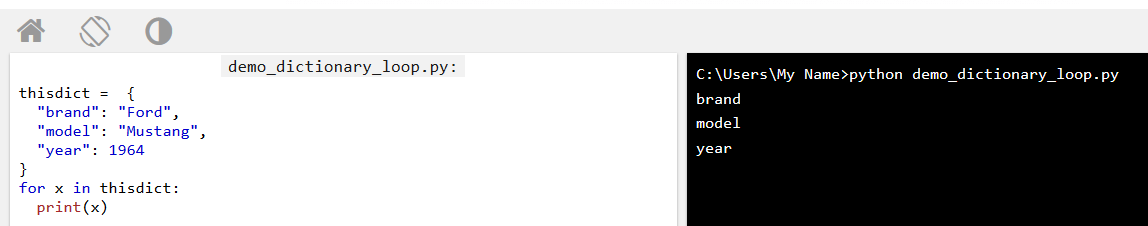
## Change Values

You can change the value of a specific item by referring to its key name:

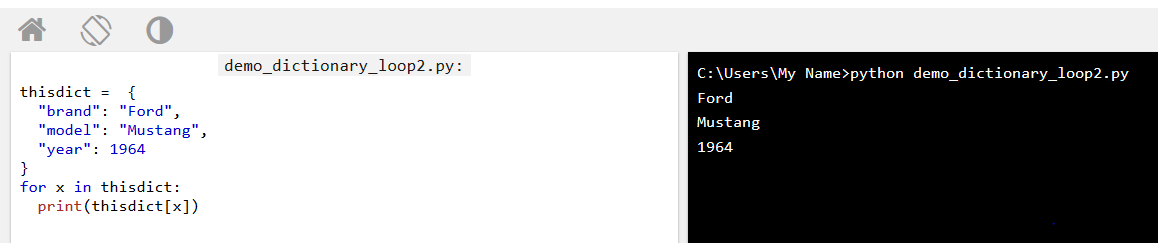


## Loop Through a Dictionary

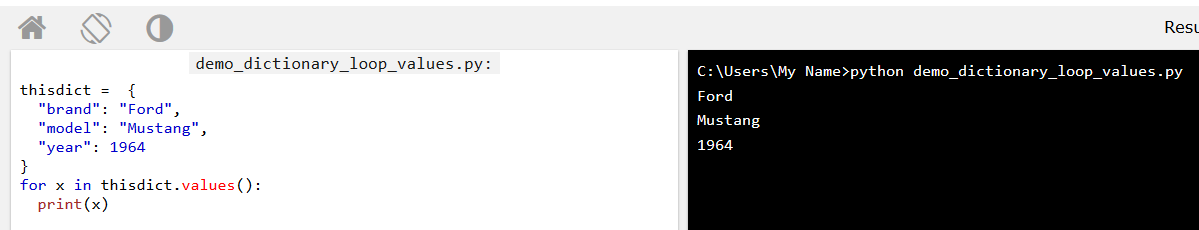
You can loop through a dictionary by using a for loop.



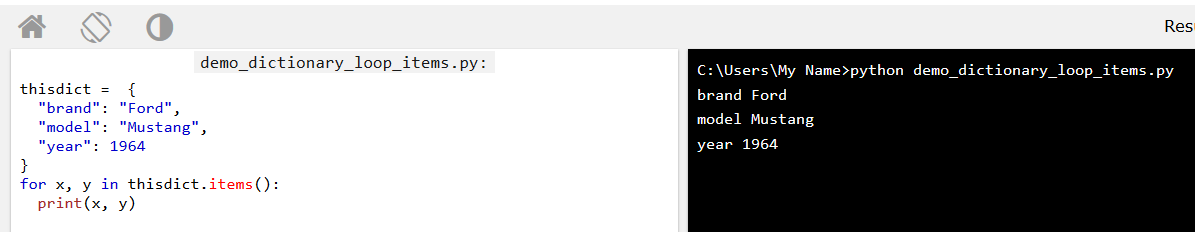
Print all values in the dictionary, one by one:



You can also use the values() function to return values of a dictionary:

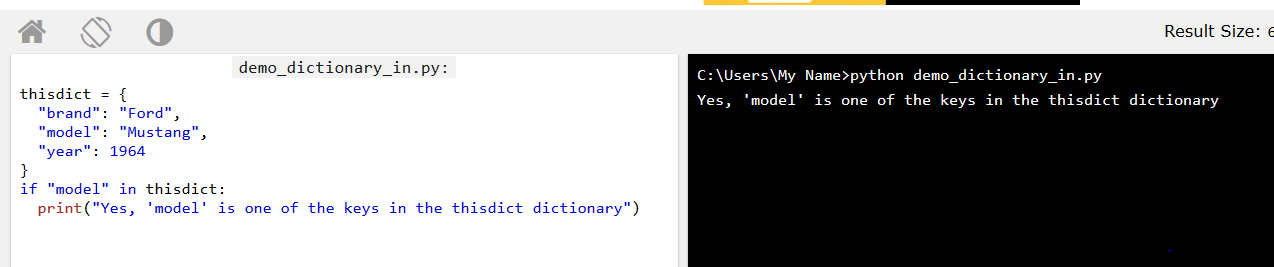


Loop through both keys and values, by using the items() function:



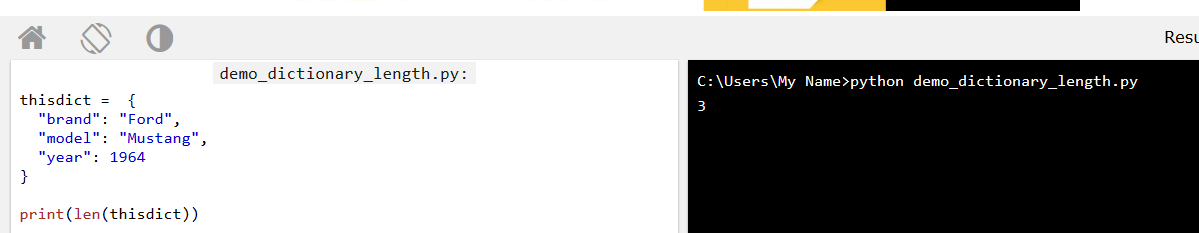
## Check if Key Exists

To determine if a specified key is present in a dictionary use the in keyword:



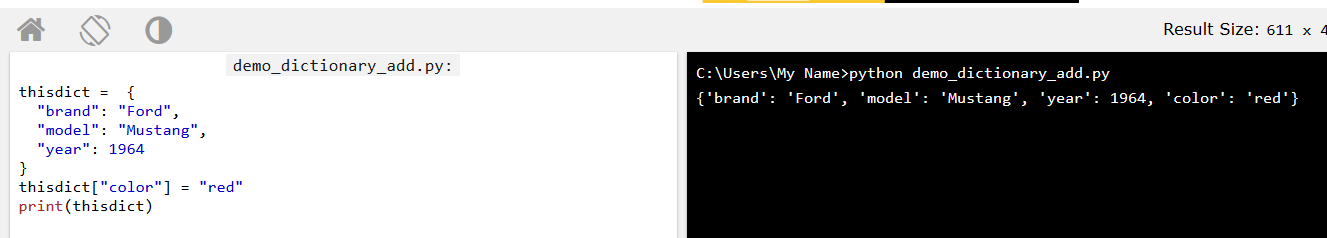
## Dictionary Length

To determine how many items (key-value pairs) a dictionary has, use the len() method.



## Adding Items

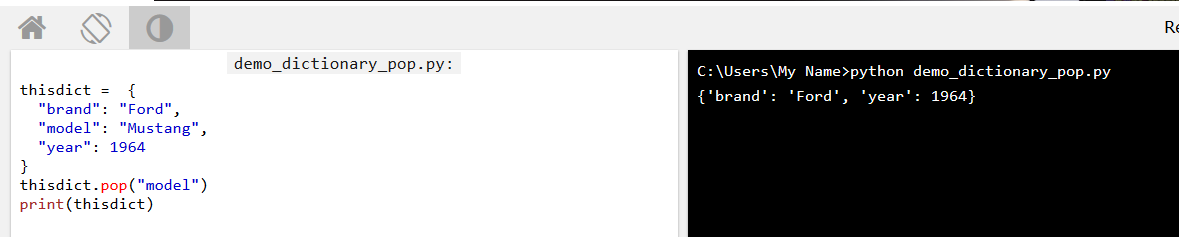
Adding an item to the dictionary is done by using a new index key and assigning a value to it:



## Removing Items

There are several methods to remove items from a dictionary:

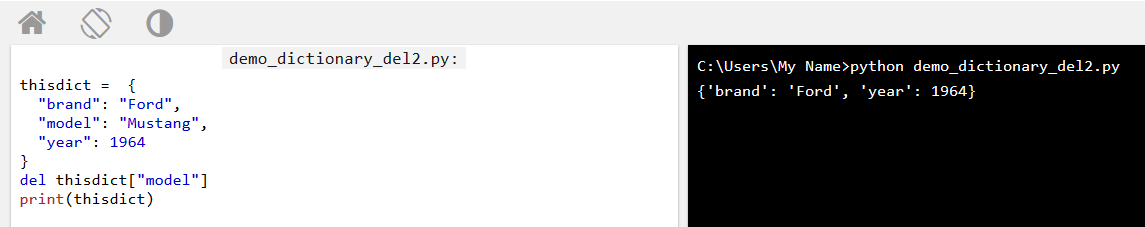
he pop() method removes the item with the specified key name:



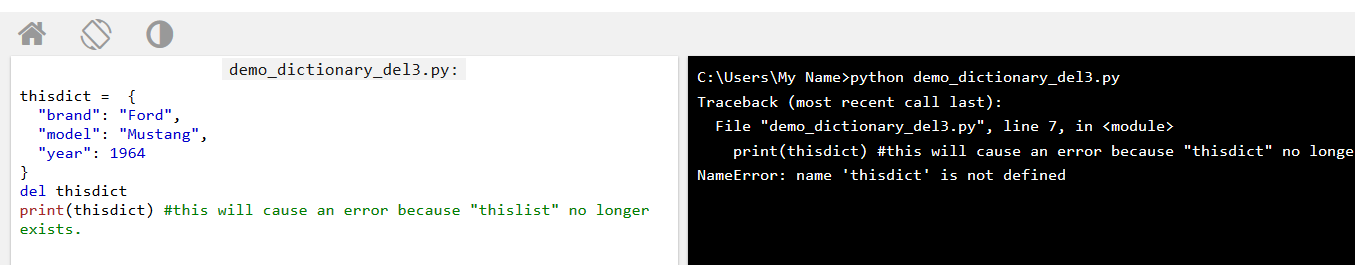
The popitem() method removes the last inserted item (in versions before 3.7, a random item is removed instead):



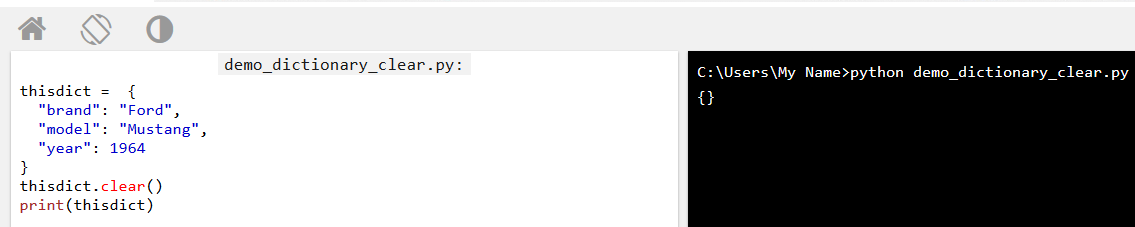
The del keyword removes the item with the specified key name

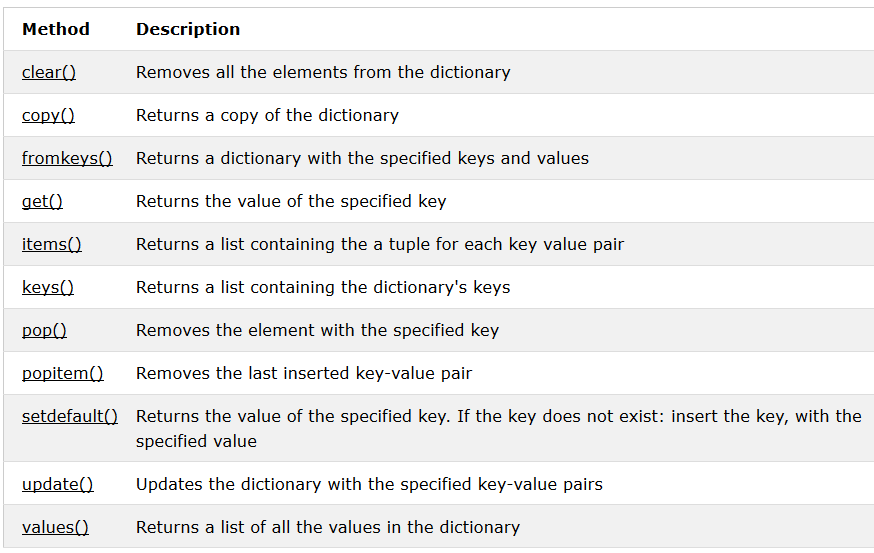


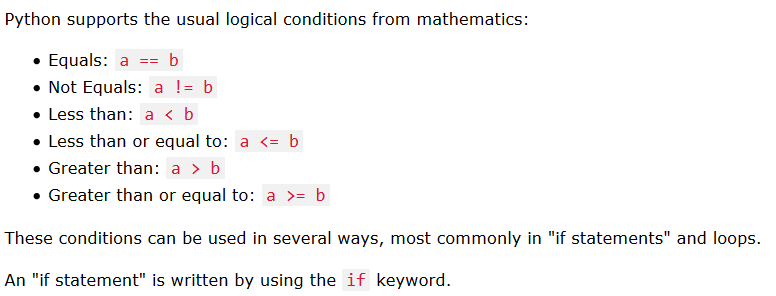
The del keyword can also delete the dictionary completely:

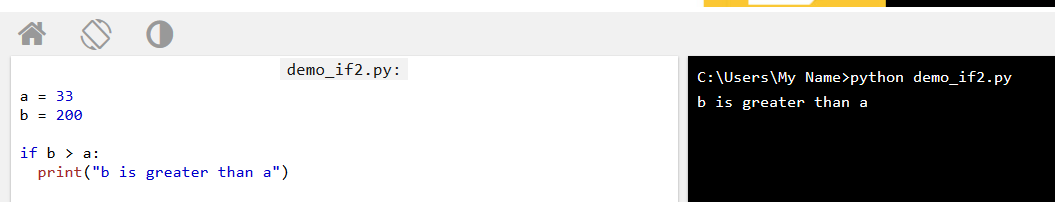


The clear() keyword empties the dictionary:



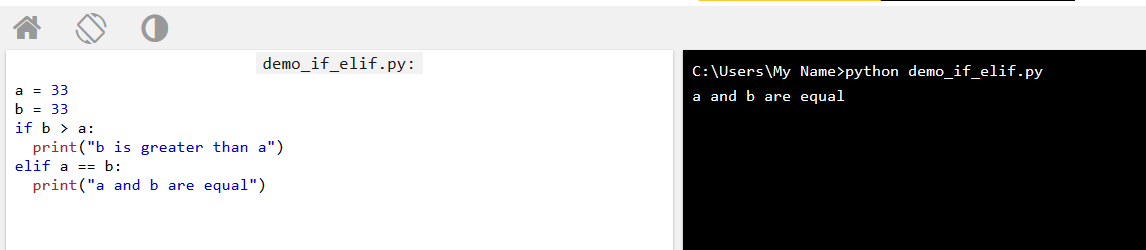


  
If statement:



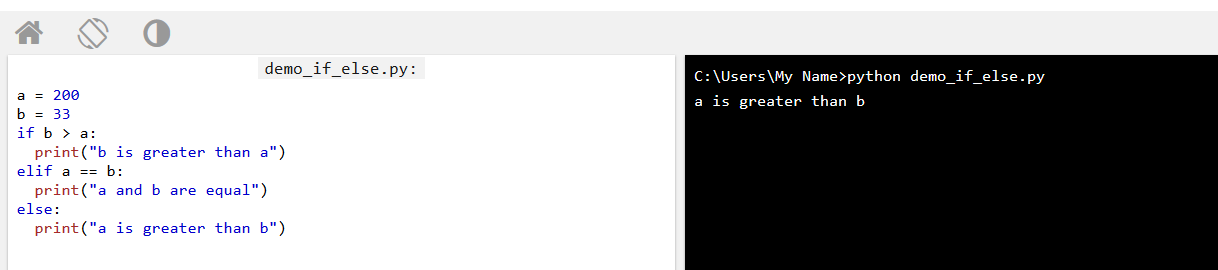
## Elif

The elif keyword is pythons way of saying "if the previous conditions were not true, then try this condition"



## Else

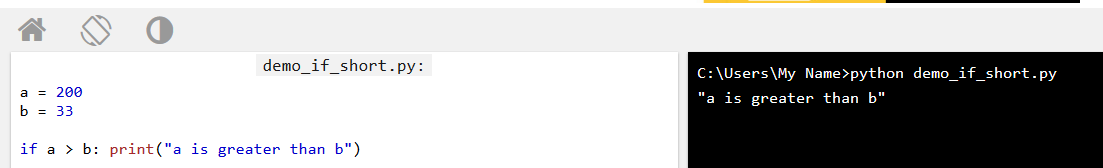
The else keyword catches anything which isn't caught by the preceding conditions.



## Short Hand If

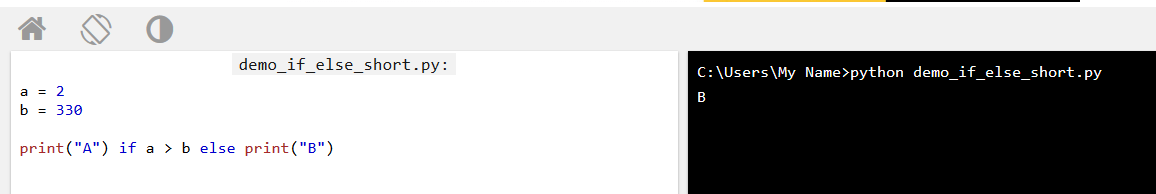
If you have only one statement to execute, you can put it on the same line as the if statement.

One line if statement:

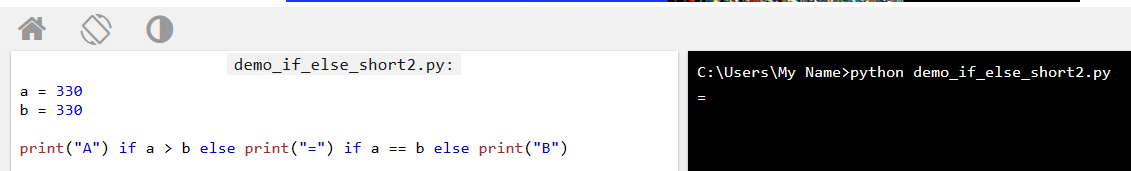


## Short Hand If ... Else

If you have only one statement to execute, one for if, and one for else, you can put it all on the same line: One line if else statement:

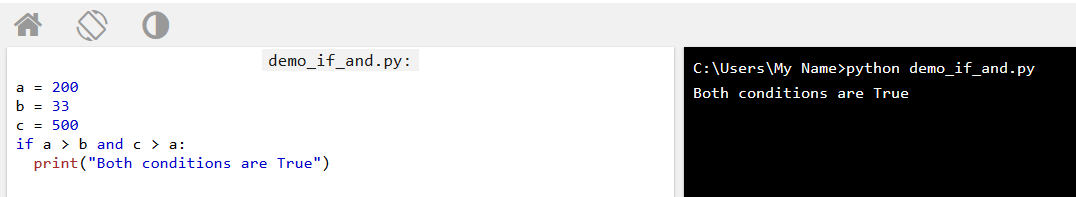


You can also have multiple else statements on the same line: One line if else statement, with 3 conditions:



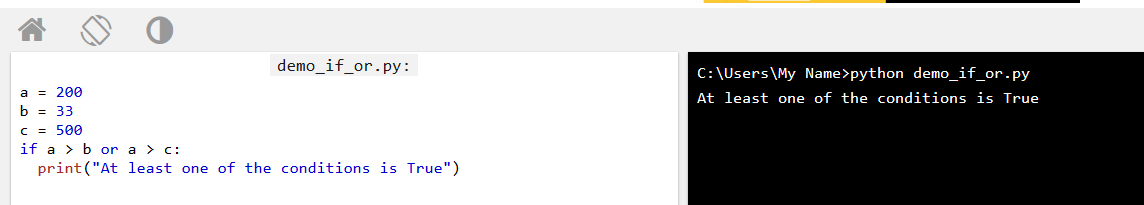
## And

The and keyword is a logical operator, and is used to combine conditional statements: Test if a is greater than b, AND if c is greater than a:



## Or

The or keyword is a logical operator, and is used to combine conditional statements: Test if a is greater than b, OR if a is greater than c:



**Python Loops**

Python has two primitive loop commands:

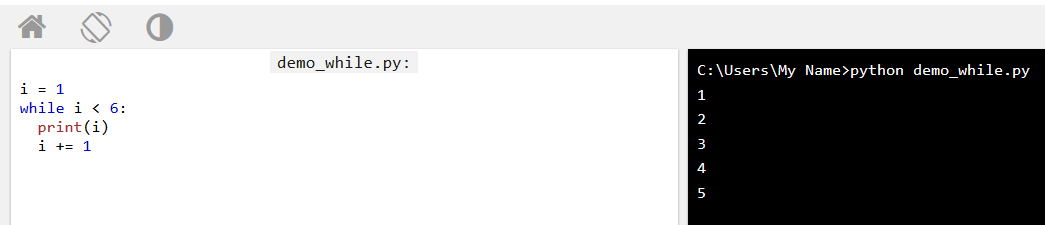
while loops

for loops

**The while Loop**

With the while loop we can execute a set of statements as long as a condition is true.

Print i as long as i is less than 6:



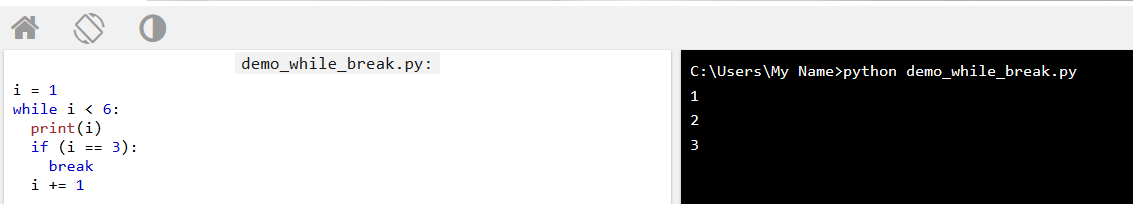
**Note:** remember to increment i, or else the loop will continue forever.

The while loop requires relevant variables to be ready, in this example we need to define an indexing variable, i, which we set to 1.

## The break Statement

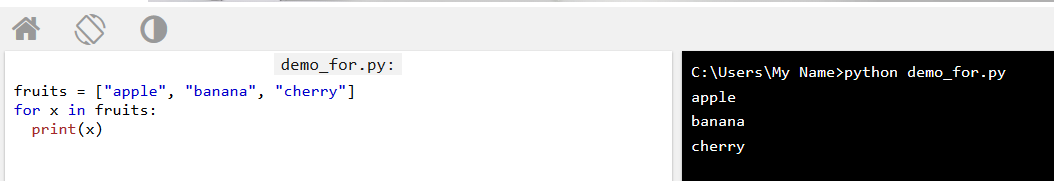
With the break statement we can stop the loop even if the while condition is true:

Exit the loop when i is 3:



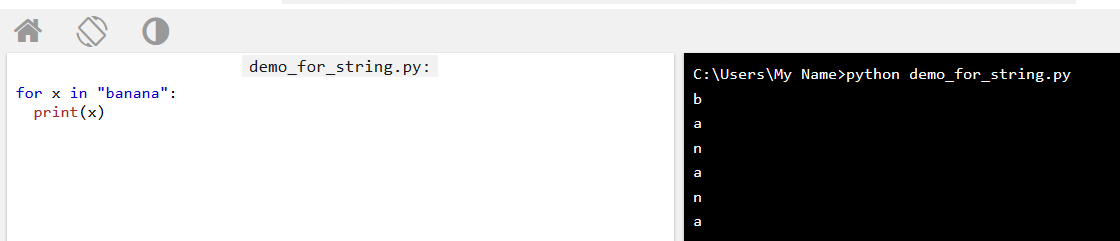
## Python For Loops

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).



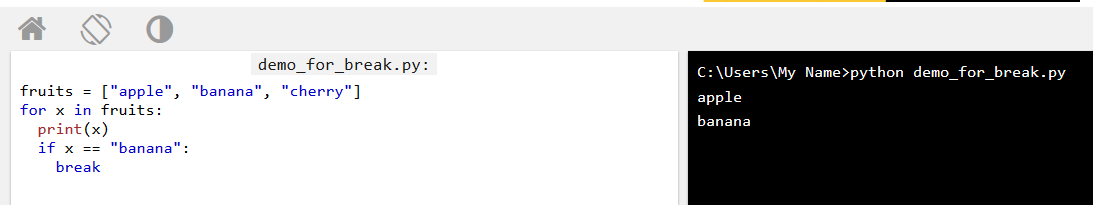
## Looping Through a String

Even strings are iterable objects, they contain a sequence of characters:

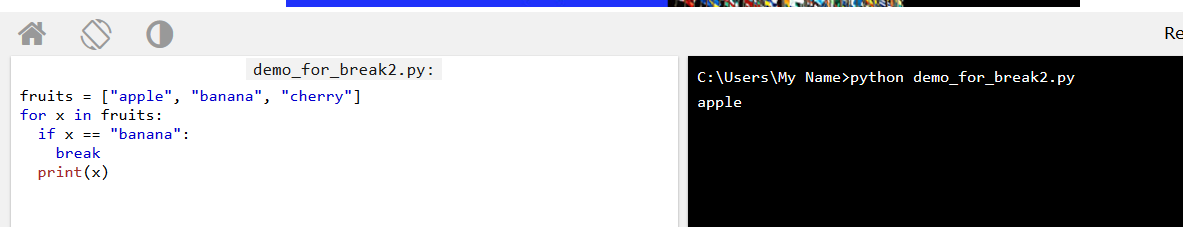


## The break Statement

With the break statement we can stop the loop before it has looped through all the items:

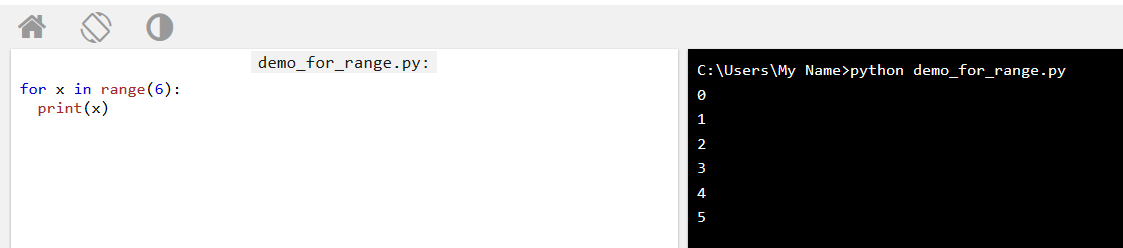


Exit the loop when x is "banana", but this time the break comes before the print:



## The range() Function

The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number. The range() function defaults to 0 as a starting value



## Else in For Loop

The else keyword in a for loop specifies a block of code to be executed when the loop is finished:

Print all numbers from 0 to 5, and print a message when the loop has ended:



## Nested Loops

A nested loop is a loop inside a loop.

The "inner loop" will be executed one time for each iteration of the "outer loop":



# Python Functions

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

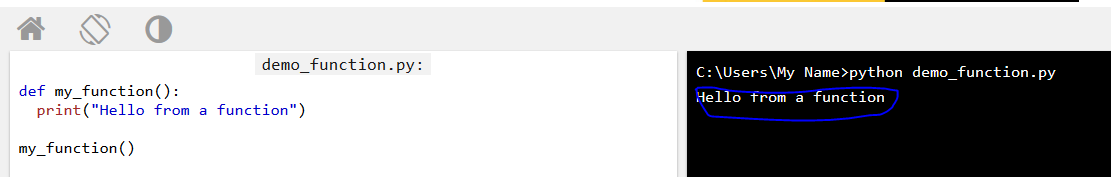
A function can return data as a result.

## Creating a Function

In Python a function is defined using the def keyword:

## Calling a Function

To call a function, use the function name followed by parenthesis:

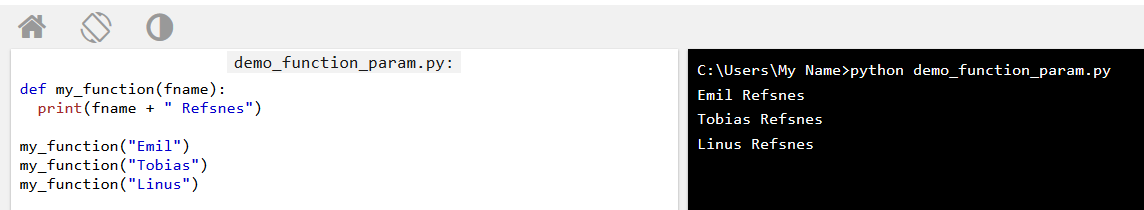


## Parameters

Information can be passed to functions as parameter.

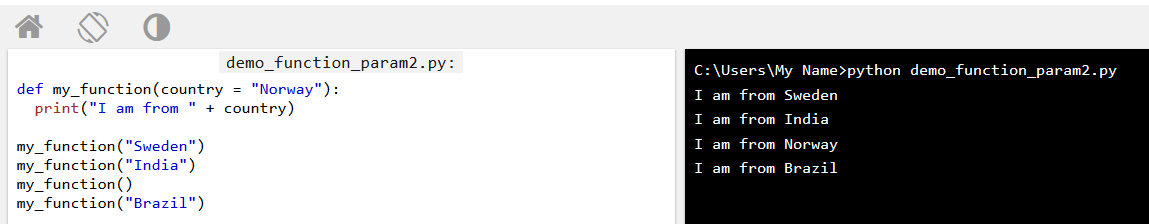
Parameters are specified after the function name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma.

The following example has a function with one parameter (fname). When the function is called, we pass along a first name, which is used inside the function to print the full name:



## Default Parameter Value

If we call the function without parameter, it uses the default value:

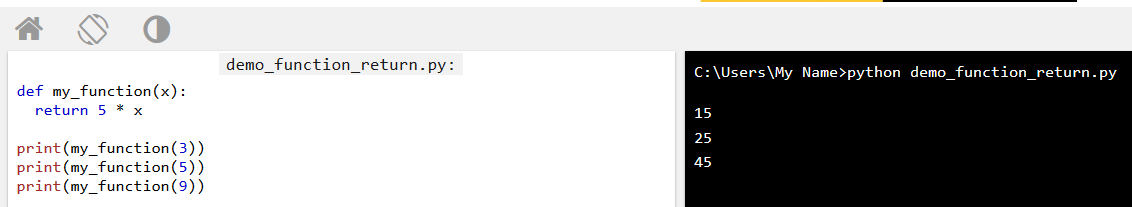


## Passing a List as a Parameter



## Return Values

To let a function return a value, use the return statement:



## Recursion

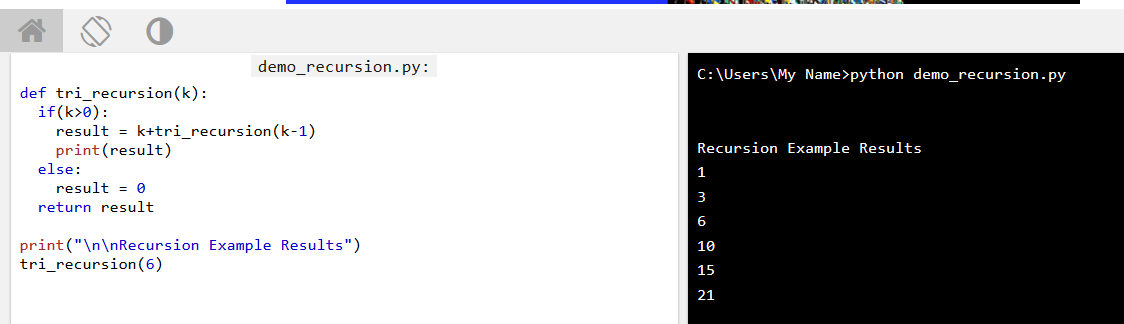
Python also accepts function recursion, which means a defined function can call itself.

Recursion is a common mathematical and programming concept. It means that a function calls itself. This has the benefit of meaning that you can loop through data to reach a result.

The developer should be very careful with recursion as it can be quite easy to slip into writing a function which never terminates, or one that uses excess amounts of memory or processor power. However, when written correctly recursion can be a very efficient and mathematically-elegant approach to programming.

In this example, tri\_recursion() is a function that we have defined to call itself ("recurse"). We use the k variable as the data, which decrements (-1) every time we recurse. The recursion ends when the condition is not greater than 0 (i.e. when it is 0).

To a new developer it can take some time to work out how exactly this works, best way to find out is by testing and modifying it.



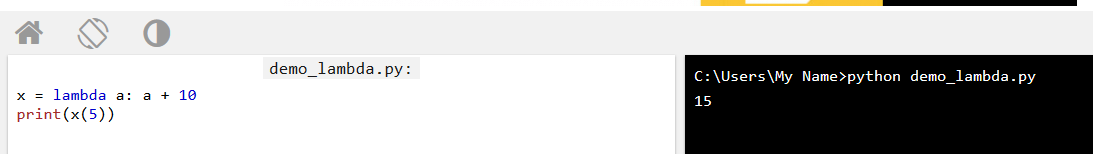
# Python Lambda

A lambda function can take any number of arguments, but can only have one expression.

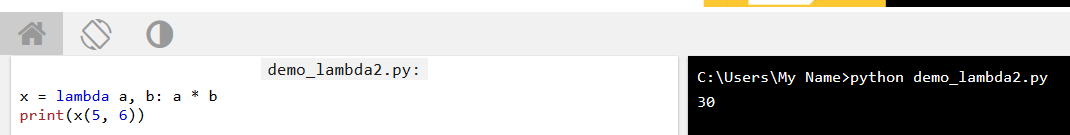
## Syntax

lambda arguments : expression

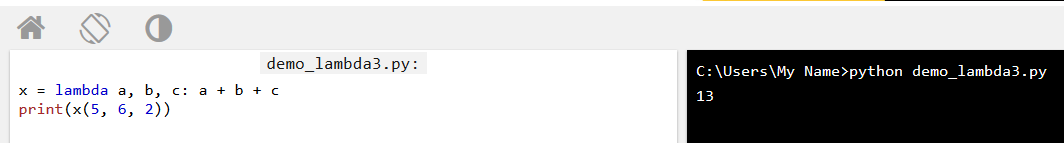
A lambda function that adds 10 to the number passed in as an argument, and print the result:



A lambda function that multiplies argument a with argument b and print the result:



A lambda function that sums argument a, b, and c and print the result:



## Why Use Lambda Functions?

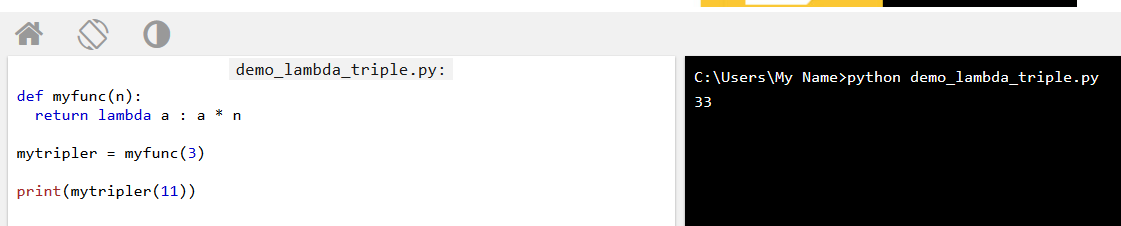
The power of lambda is better shown when you use them as an anonymous function inside another function.

Say you have a function definition that takes one argument, and that argument will be multiplied with an unknown number:



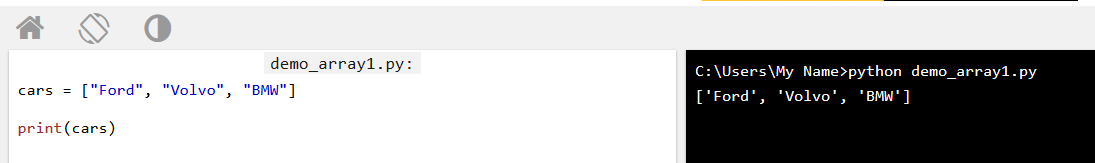
Or, use the same function definition to make a function that always triples the number you send in:

Or, use the same function definition to make both functions, in the same program:



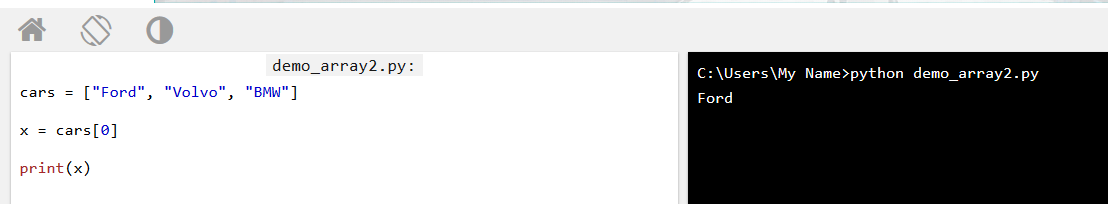
## Arrays

Arrays are used to store multiple values in one single variable:



## Access the Elements of an Array

You refer to an array element by referring to the index number.

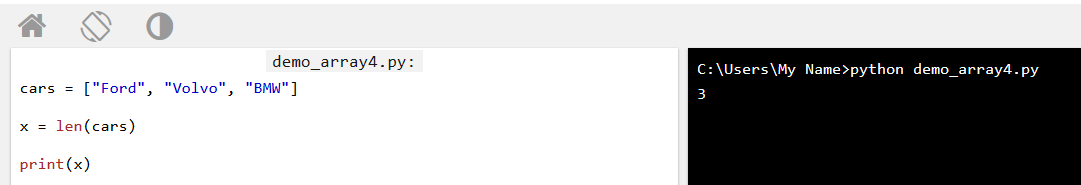


Modify the value of the first array item:

## The Length of an Array

Use the len() method to return the length of an array (the number of elements in an array).

Return the number of elements in the cars array: **Note:** The length of an array is always one more than the highest array index.



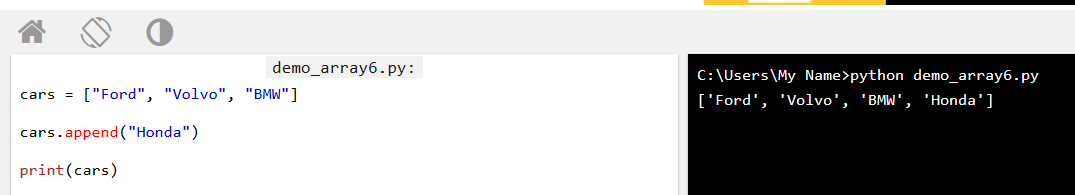
## Looping Array Elements

You can use the for in loop to loop through all the elements of an array.



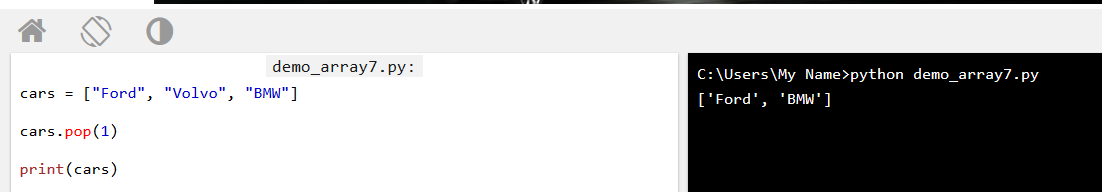
## Adding Array Elements

You can use the append() method to add an element to an array.

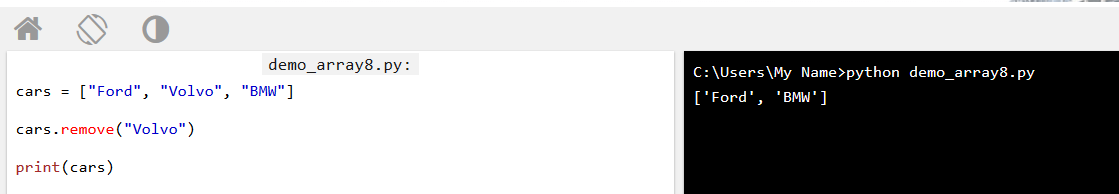


## Removing Array Elements

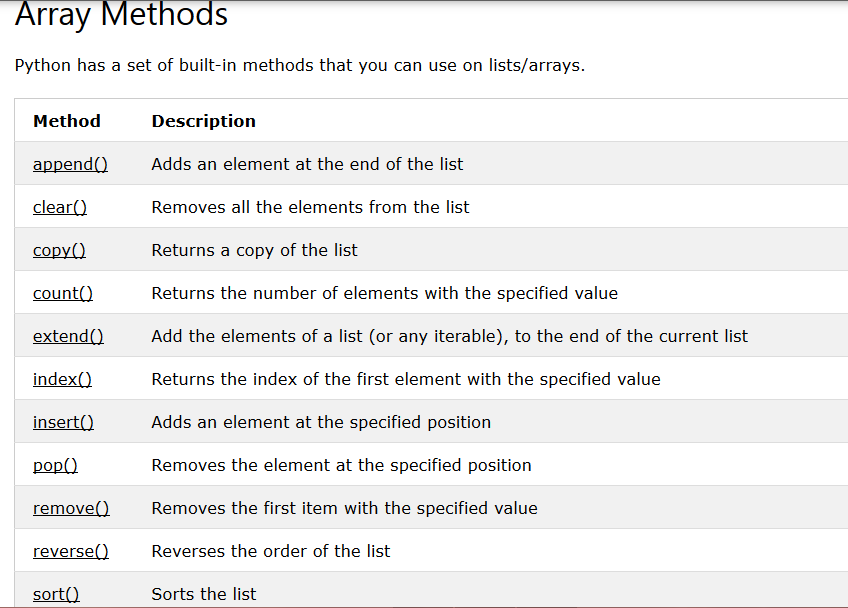
You can use the pop() method to remove an element from the array.



You can also use the remove() method to remove an element from the array.

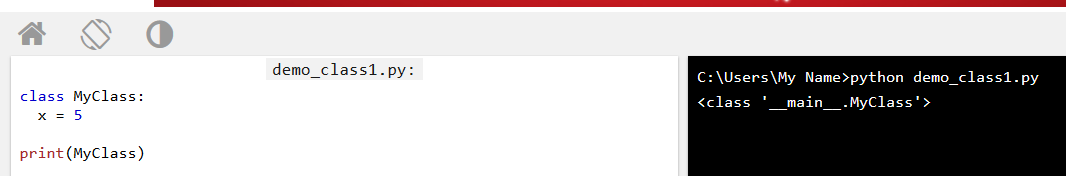


**Note:** The remove() method only removes the first occurrence of the specified value.



## Create a Class

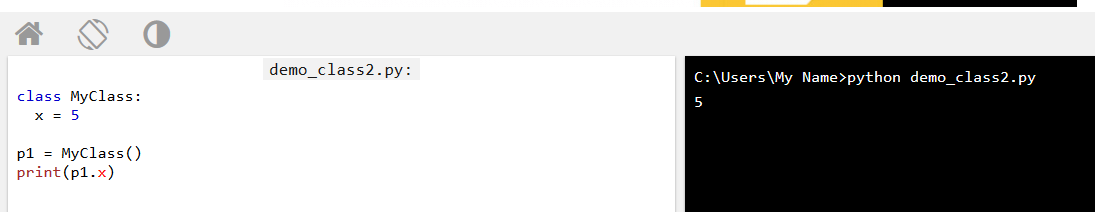
To create a class, use the keyword class:



## Create Object

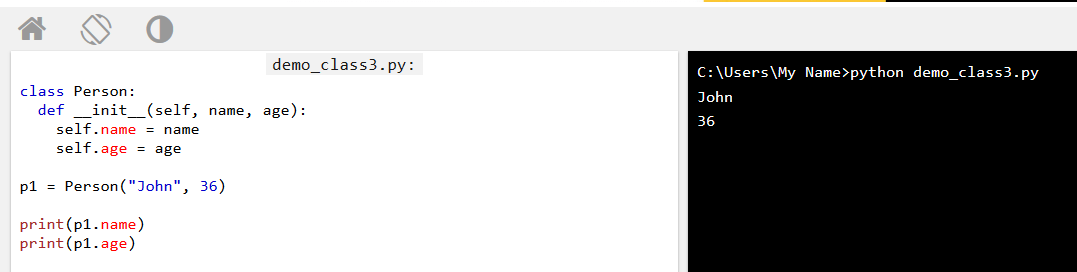
Now we can use the class named myClass to create objects:

Create an object named p1, and print the value of x:



All classes have a function called \_\_init\_\_(), which is always executed when the class is being initiated.

Use the \_\_init\_\_() function to assign values to object properties, or other operations that are necessary to do when the object is being created:



**Note:** The \_\_init\_\_() function is called automatically every time the class is being used to create a new object.

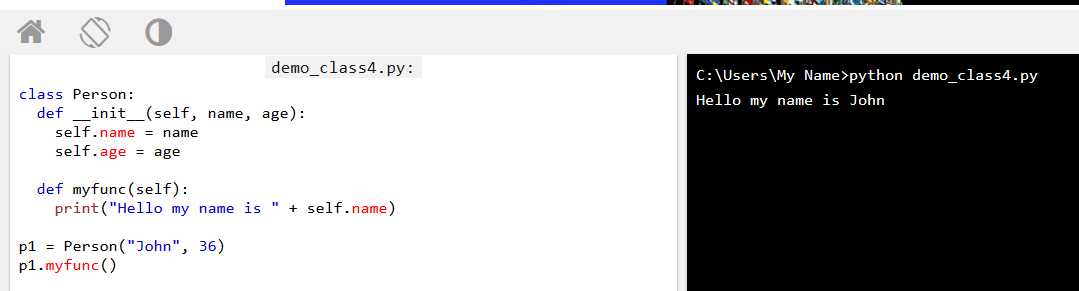
## Object Methods

Objects can also contain methods. Methods in objects are functions that belongs to the object.

Let us create a method in the Person class:

Example

Insert a function that prints a greeting, and execute it on the p1 object:

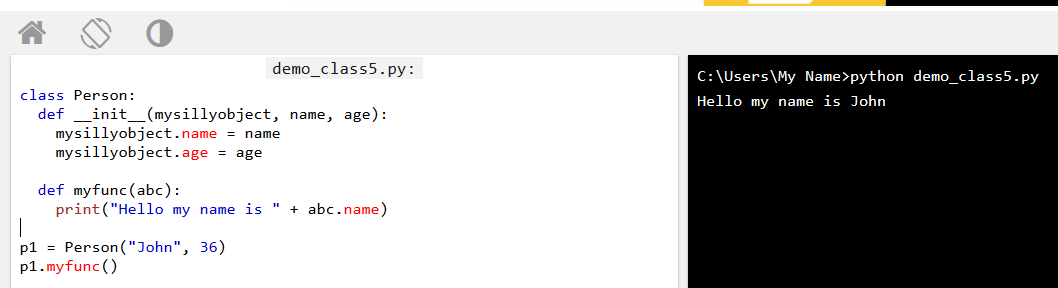


## The self Parameter

**Note:** The self parameter is a reference to the current instance of the class, and is used to access variables that belongs to the class. It does not have to be named self , you can call it whatever you like, but it has to be the first parameter of any function in the class:

Example

Use the words mysillyobject and abc instead of self:



## Modify Object Properties

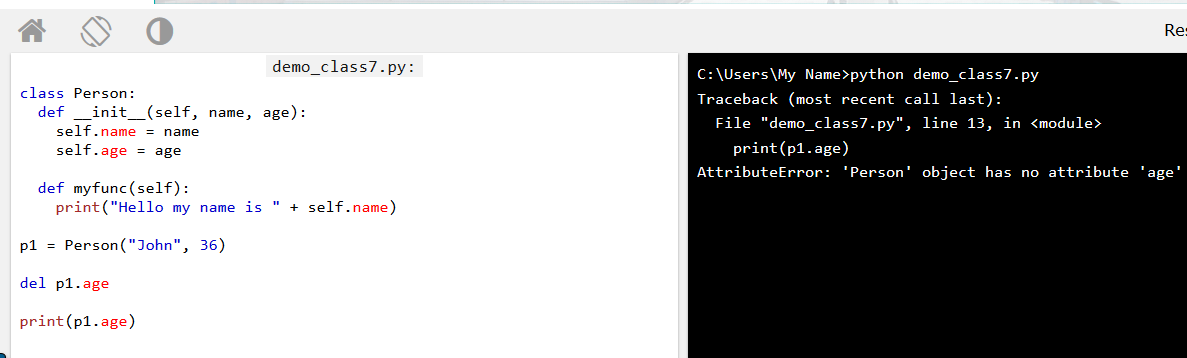


## Delete Object Properties

You can delete properties on objects by using the del keyword:

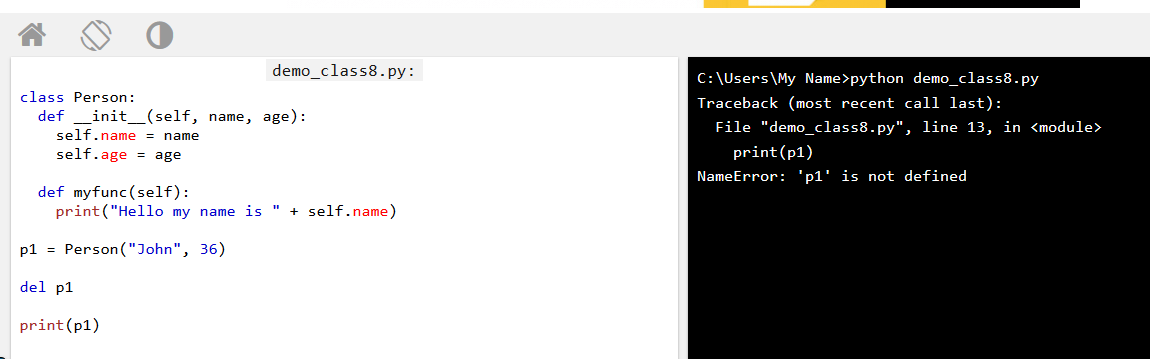
Example

Delete the age property from the p1 object:



## Delete Objects

You can delete objects by using the del keyword:



# Python Modules

Module mens ekta ba ekadik phython program page,ja ekon file location a save kora takbe,jokon nedd hobe tokon oi save kora program tar saving name ta sudo import name”” ta deley add hoye jabe.

Create a Module

To create a module just save the code you want in a file with the file extension .py

### Example

Save this code in a file named mymodule.py

def greeting(name):  
  print("Hello, " + name)

## Use a Module

Now we can use the module we just created, by using the import statement:

import mymodule  
  
mymodule.greeting("Jonathan")



**Note:** When using a function from a module, use the syntax: *module\_name.function\_name.*

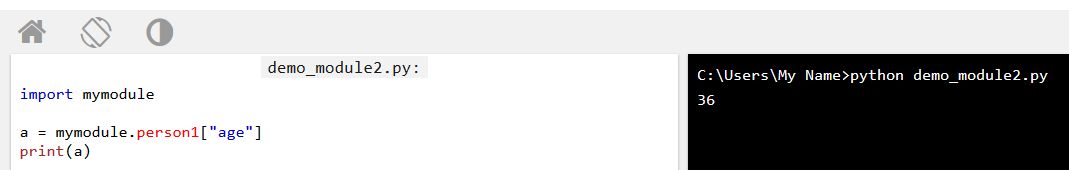
### Example

Save this code in the file mymodule.py

person1 = {  
  "name": "John",  
  "age": 36,  
  "country": "Norway"  
}

Import the module named mymodule, and access the person1 dictionary:

import mymodule  
  
a = mymodule.person1["age"]  
print(a)



## Naming a Module

You can name the module file whatever you like, but it must have the file extension .py

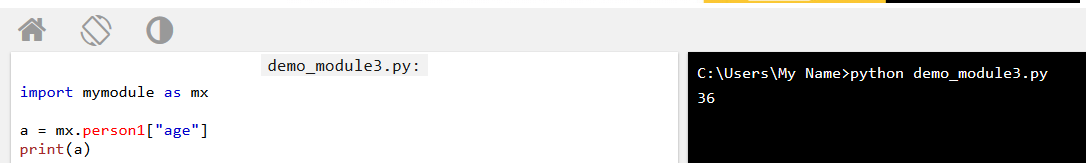
## Re-naming a Module

You can create an alias when you import a module, by using the as keyword:

### Example

Create an alias for mymodule called mx:

import mymodule as mx  
  
a = mx.person1["age"]  
print(a)



## Built-in Modules

There are several built-in modules in Python, which you can import whenever you like.

### Example

Import and use the platform module:

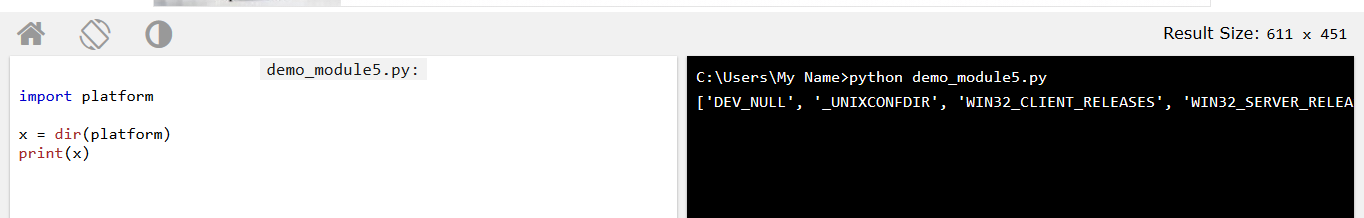
import platform  
  
x = platform.system()  
print(x)



## Using the dir() Function

There is a built-in function to list all the function names (or variable names) in a module. The dir() function:

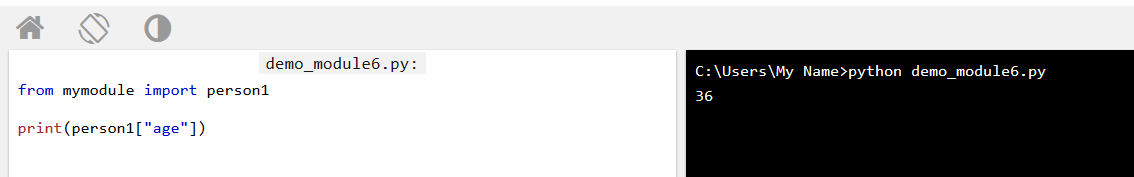
**Note:** The dir() function can be used on all modules, also the ones you create yourself.

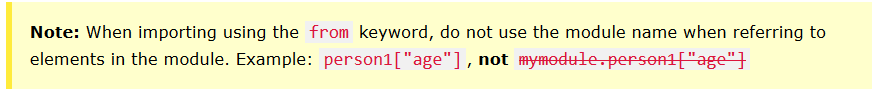


## Import From Module

You can choose to import only parts from a module, by using the from keyword.

def greeting(name):  
  print("Hello, " + name)  
  
person1 = {  
  "name": "John",  
  "age": 36,  
  "country": "Norway"  
}



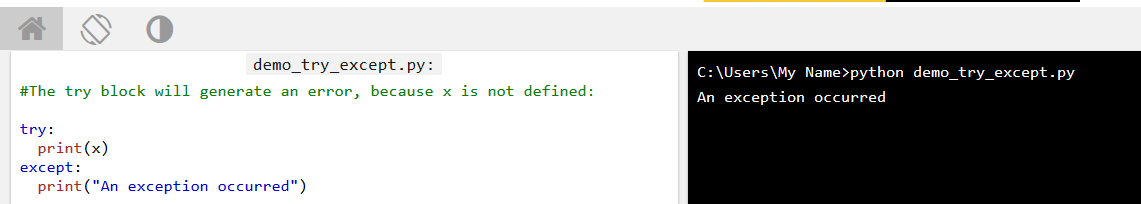


# Python Try, python except, python finally

* Try block error check kore//he try block lets you test a block of code for errors.
* Try block error hole ,Except block er output show korbe//The except block lets you handle the error.
* Try & except block ji hok na keno Finally block output show korbe// The finally block lets you execute code, regardless of the result of the try- and except blocks.
* NT:: try and except block onek ta if and else condition er moto, jemon if true hole else kaj kore na ,abr if false hole else kaj kore
* Kono karone program er j block ta error asar possibility ase seta try block a loikbo and try block er niche Jodi except na like tobe progam exit hoyejabe ba try block error hole next stepa progam ar kaj korbe na,exit hoye jabe,so try likle except liktey hobe

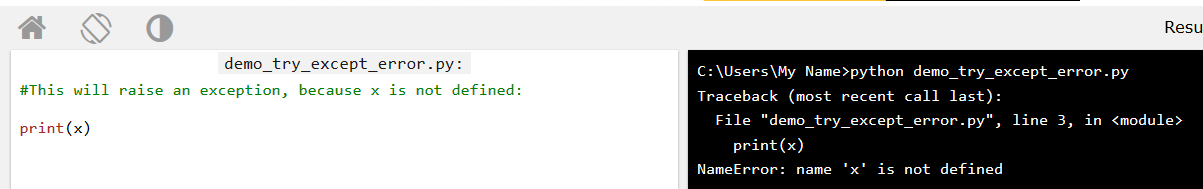
### Example

The try block will generate an exception, because x is not defined:



### Example

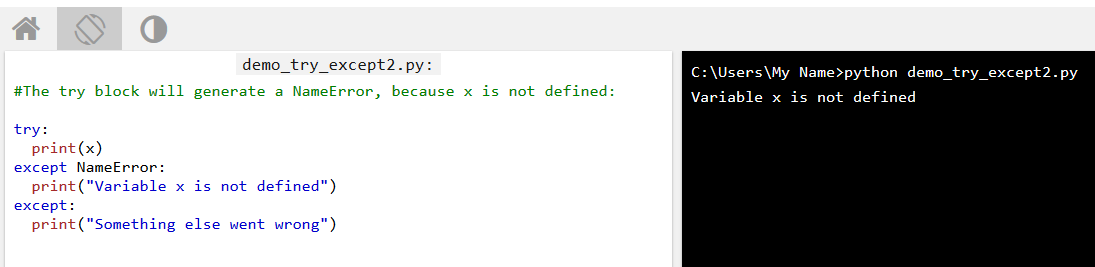
This statement will raise an error, because x is not defined:



## Many Exceptions

You can define as many exception blocks as you want,

Print one message if the try block raises a NameError and another for other errors:

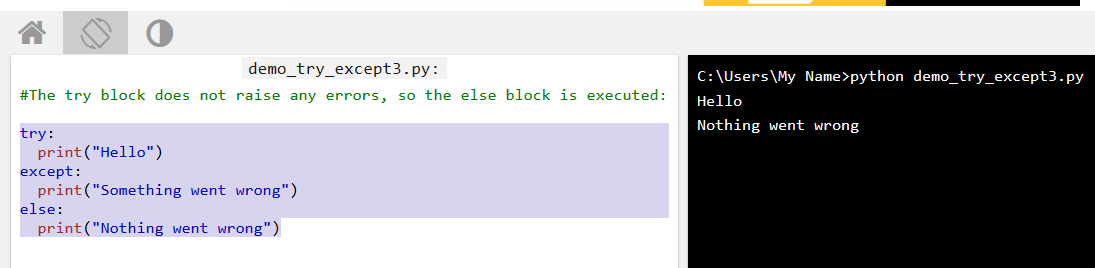


## Else

You can use the else keyword to define a block of code to be executed if no errors were raised:

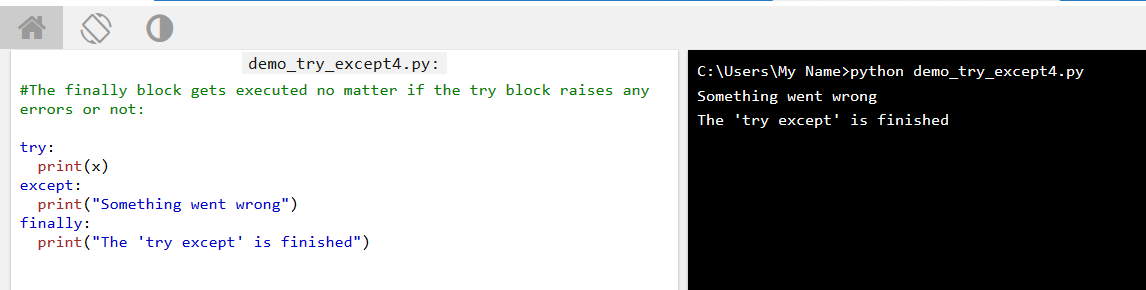
## Else

You can use the else keyword to define a block of code to be executed if no errors were raised: means else block onekta finally block er motoy but else output show korbe ,jokon try block error hobe na.



## Finally

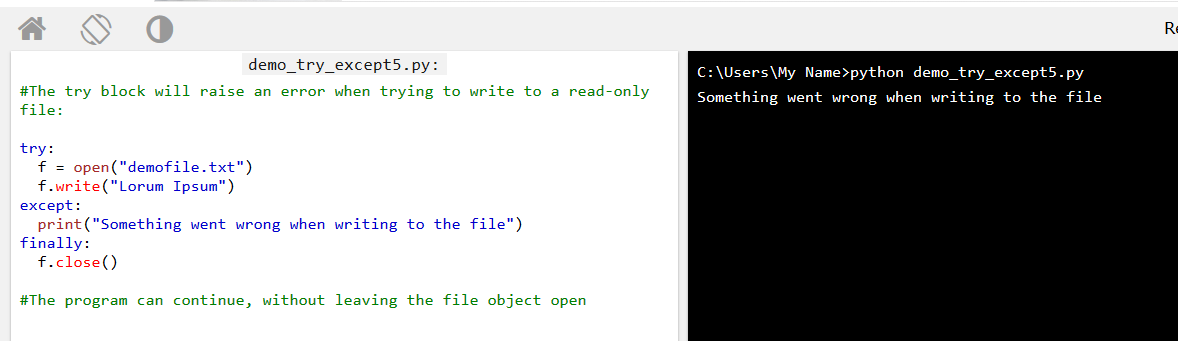
The finally block, if specified, will be executed regardless if the try block raises an error or not.



This can be useful to close objects and clean up resources:

### Example

Try to open and write to a file that is not writable:



# Python RegEx

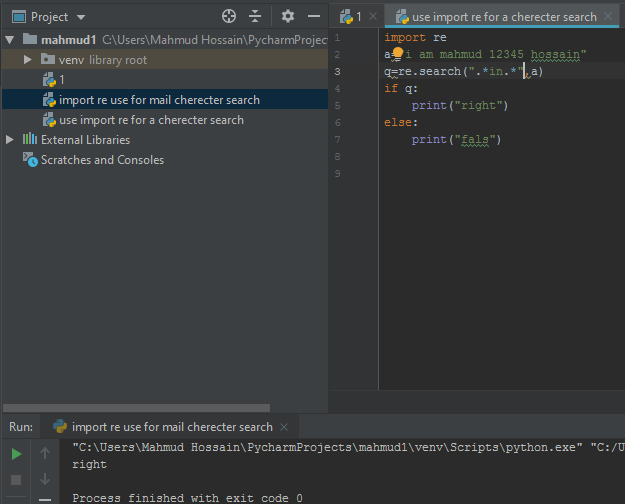
RegEx can be used to check if a string contains the specified search pattern.

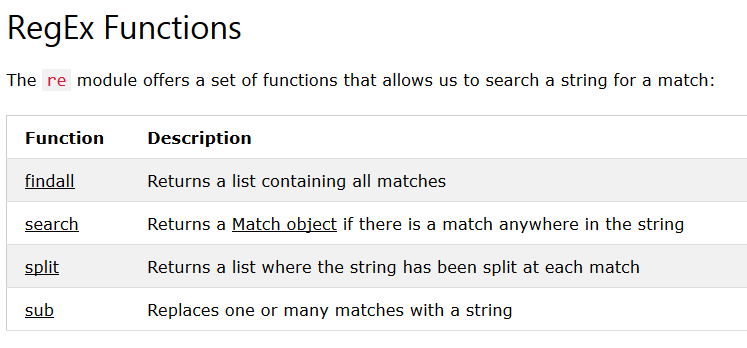
Python has a built-in package called re, which can be used to work with Regular Expressions.

Import the re module:

import re

means import re kore kono sentence or string teke cherecter search kora ji.like as “Mahmud hossain” ey string a Mahmud/h/a ase kin i ta buja ji.



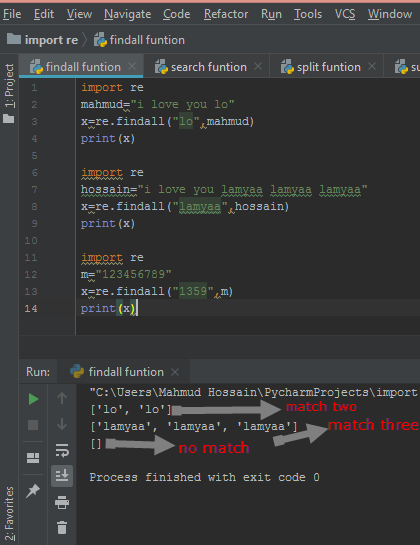


## The findall() Function

The findall() function returns a list containing all matches.

### Example

Print a list of all matches:

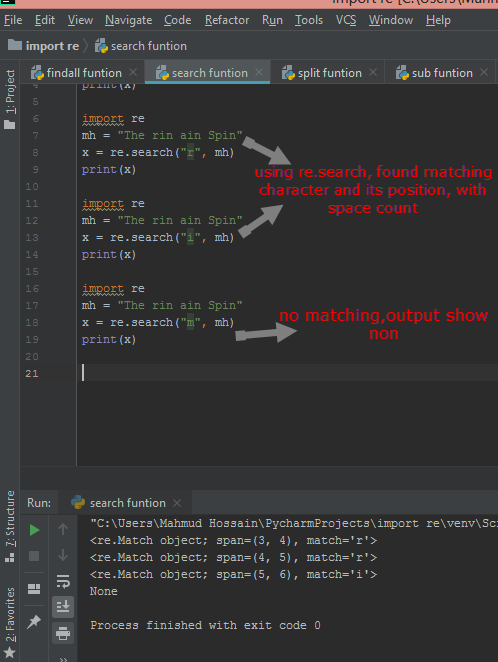


If no matches are found, an empty list is returned:

## The search() Function

The search() function searches the string for a match, and returns a [Match object](https://www.w3schools.com/python/python_regex.asp#matchobject) if there is a match.

If there is more than one match, only the first occurrence of the match will be returned:



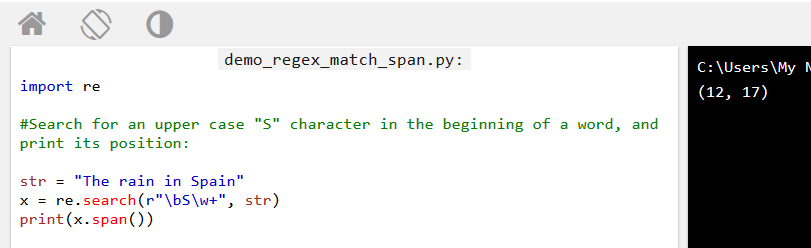
The Match object has properties and methods used to retrieve information about the search, and the result:

.span() returns a tuple containing the start-, and end positions of the match.  
.string() returns the string passed into the function  
.group() returns the part of the string where there was a match

### Example

Print the position (start- and end-position) of the first match occurrence.

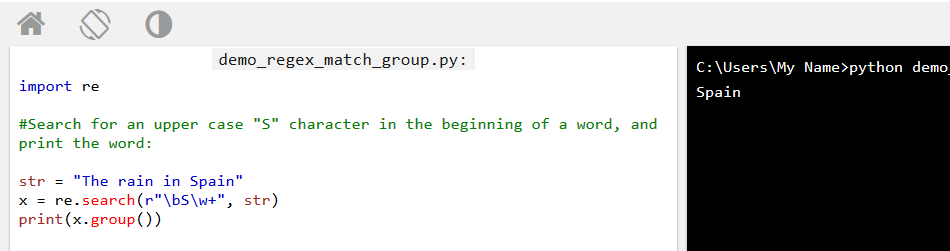
The regular expression looks for any words that starts with an upper case "S":

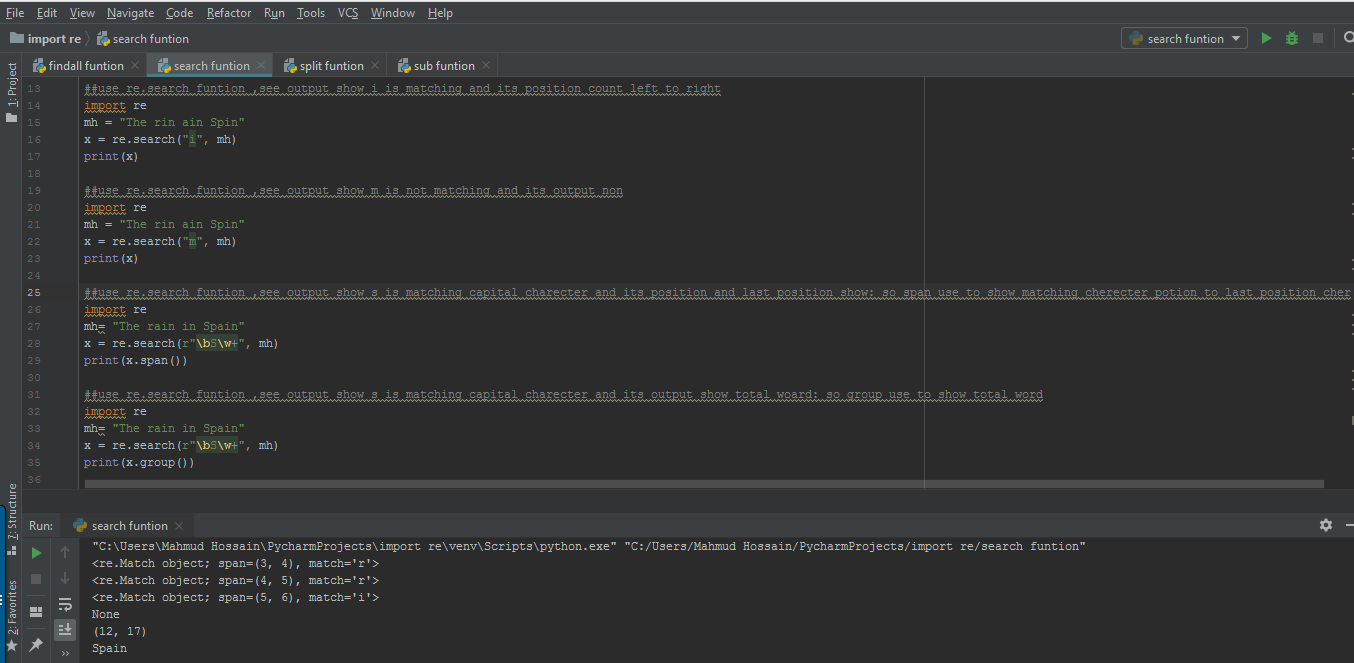


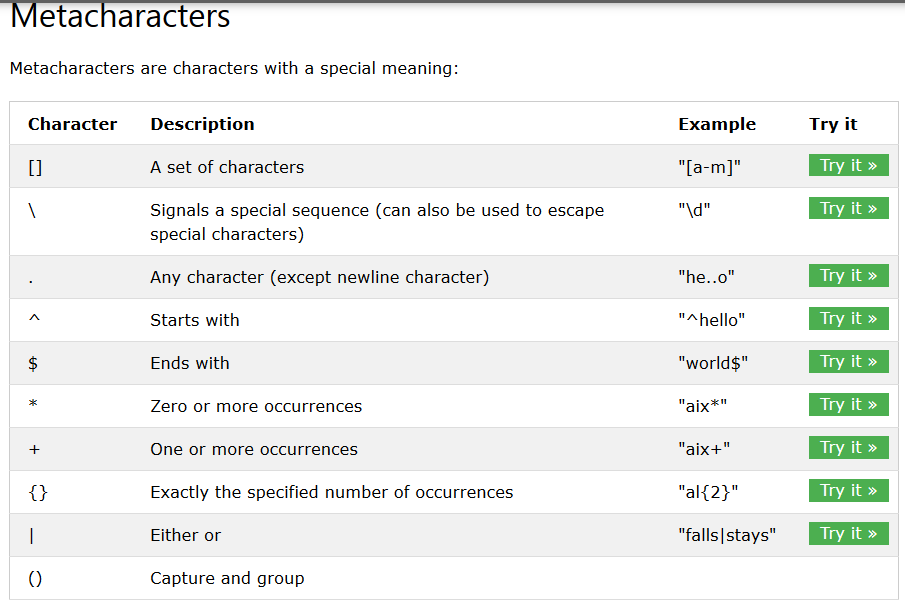
### Example

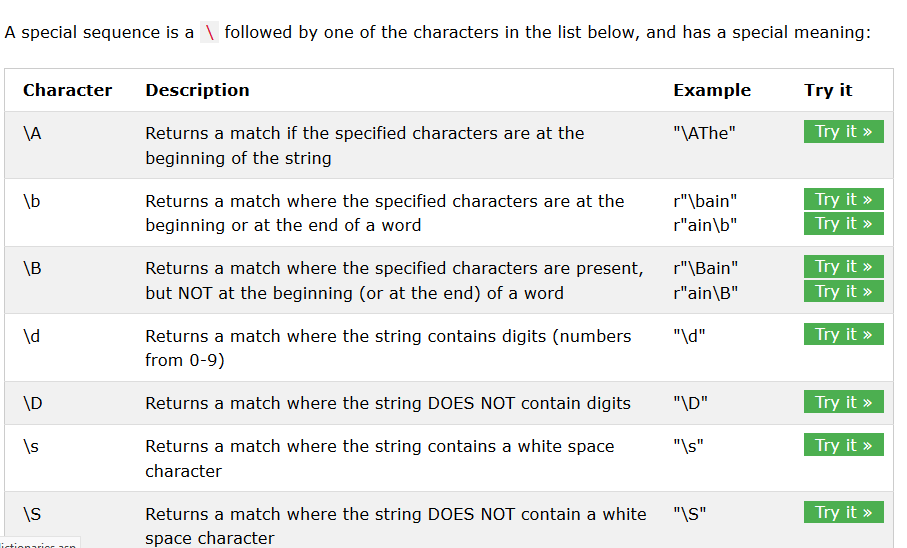
Print the part of the string where there was a match.

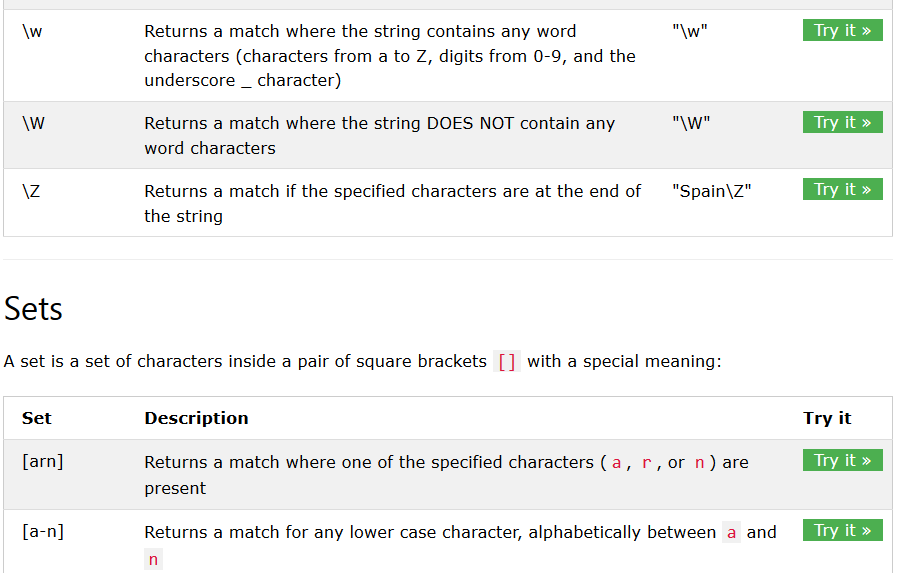
The regular expression looks for any words that starts with an upper case "S":

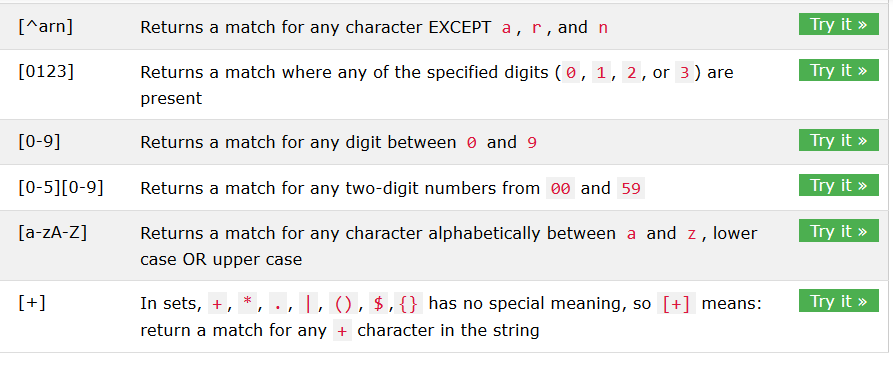


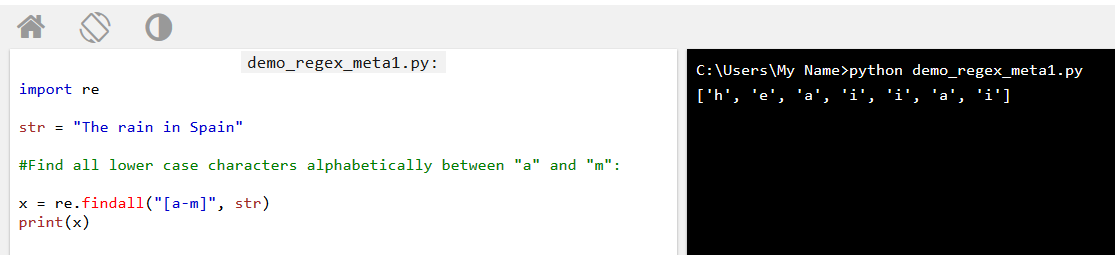




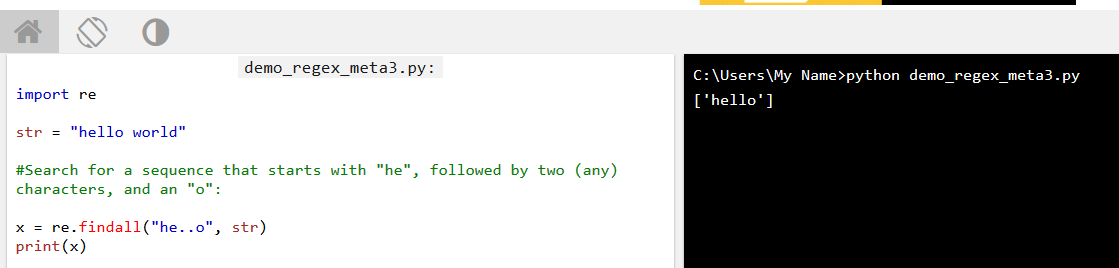


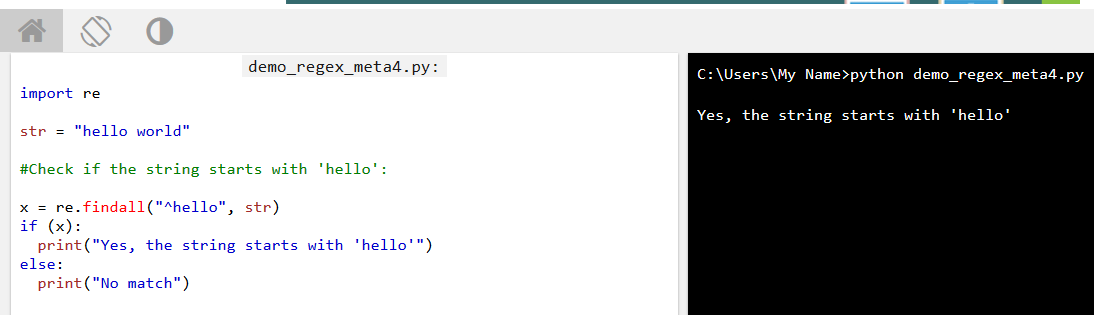


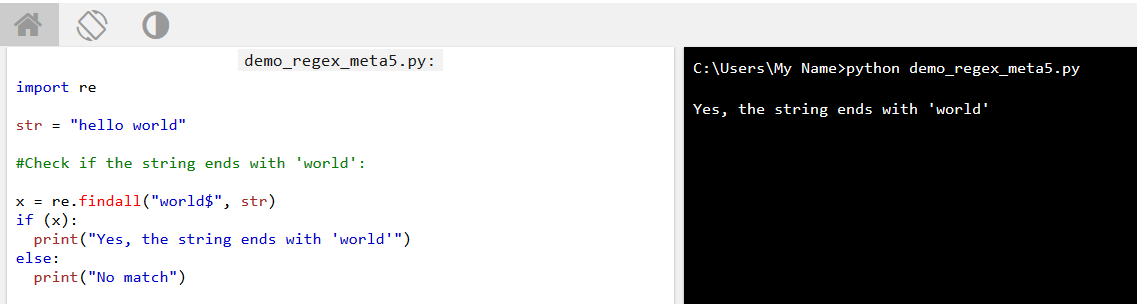


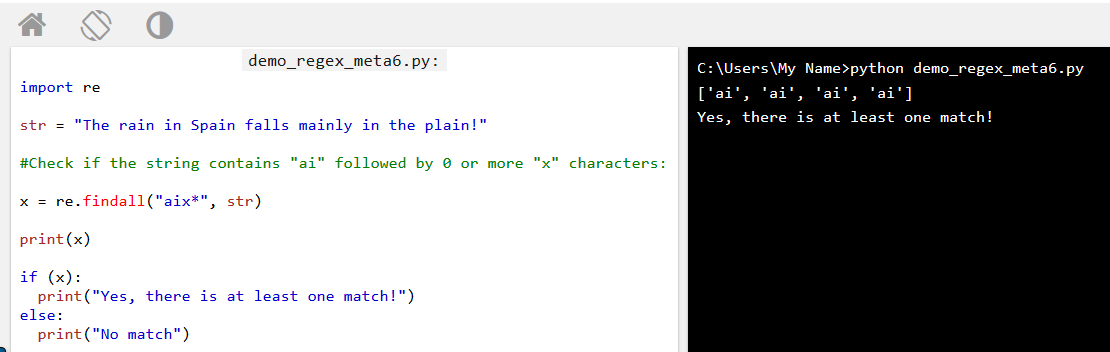


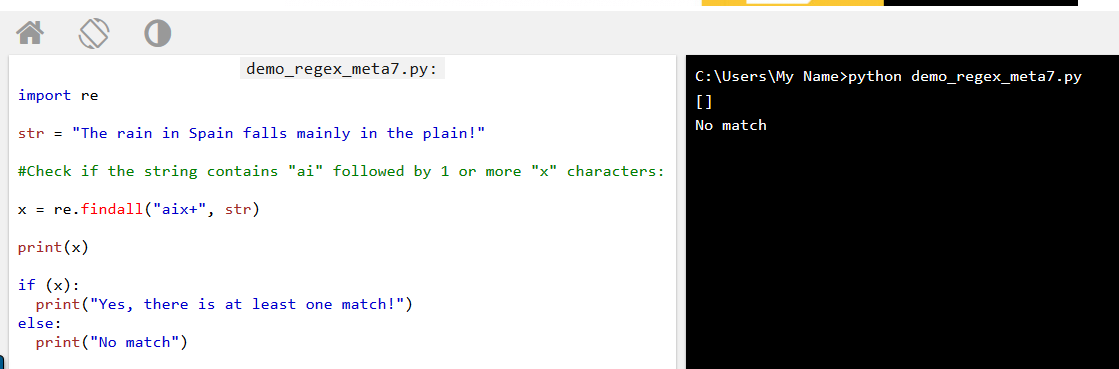


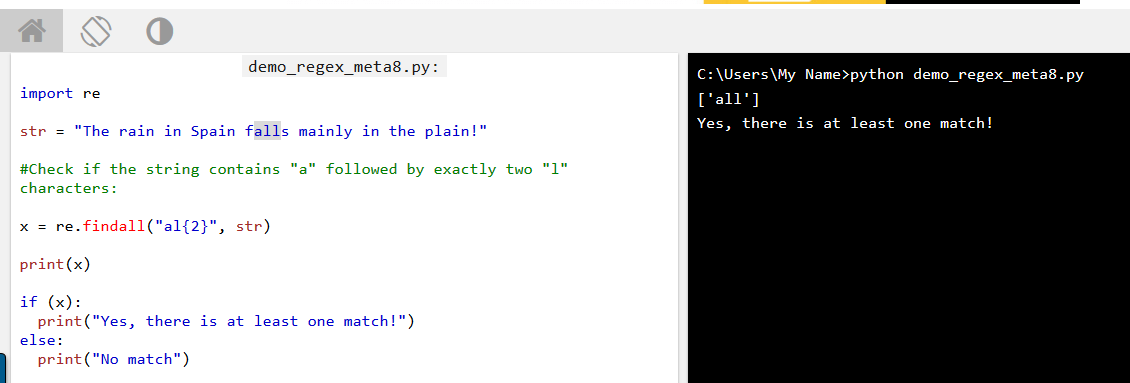


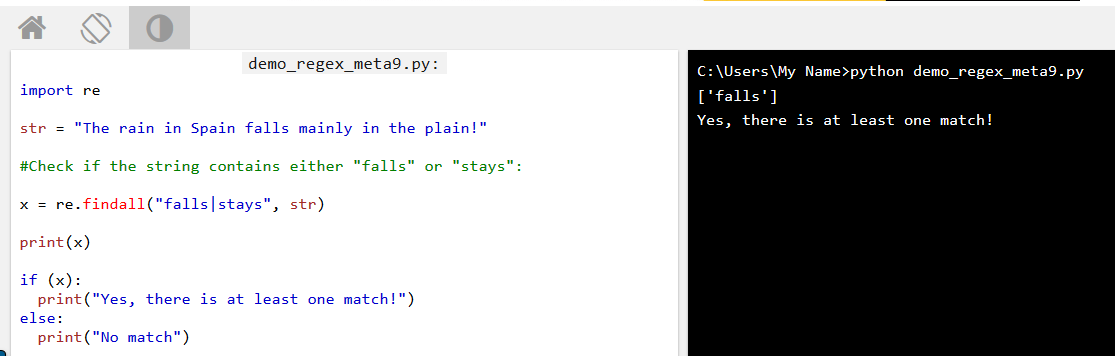












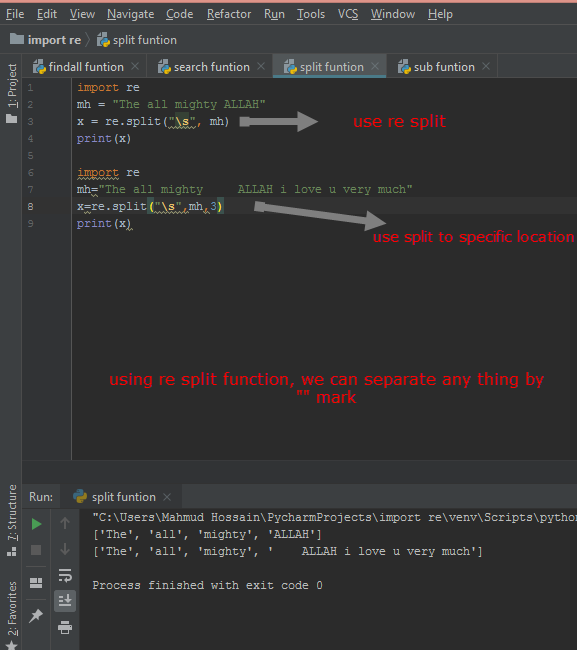
## The split() Function

The split() function returns a list where the string has been split at each match:

### Example

Split at each white-space character:

You can control the number of occurrences by specifying the maxsplit parameter:

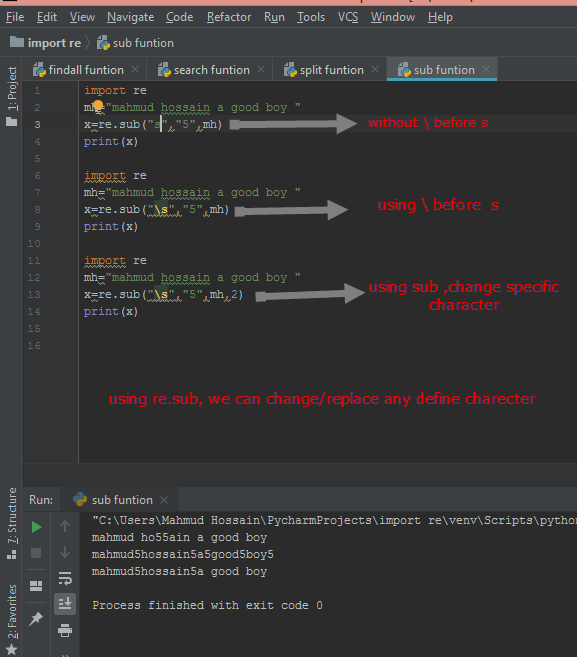


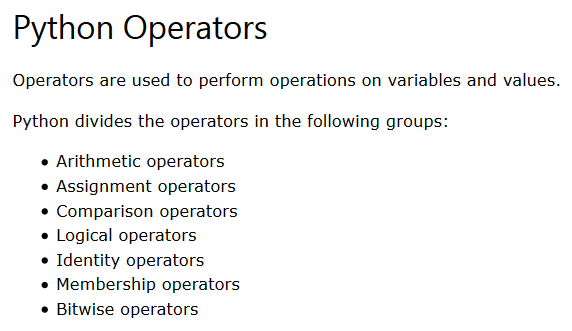
## The sub() Function

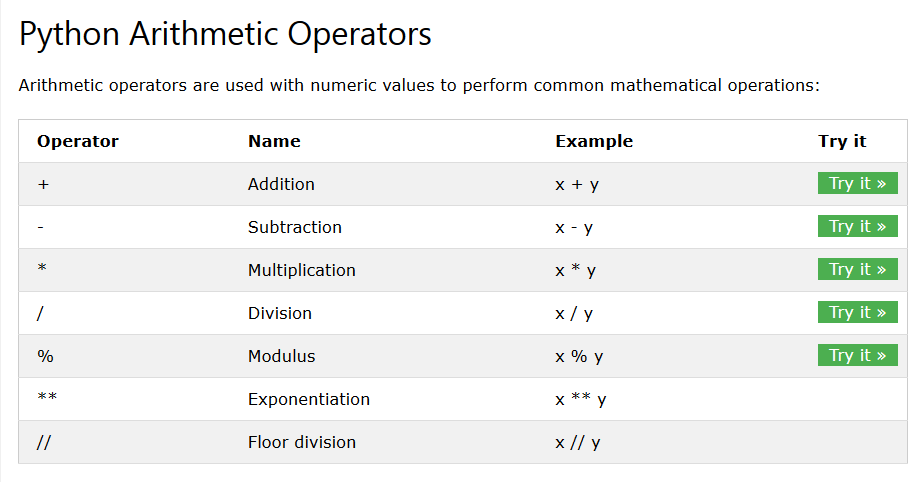
The sub() function replaces the matches with the text of your choice:

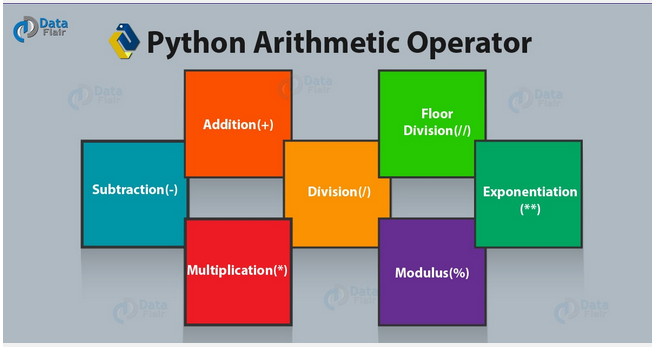
### Example

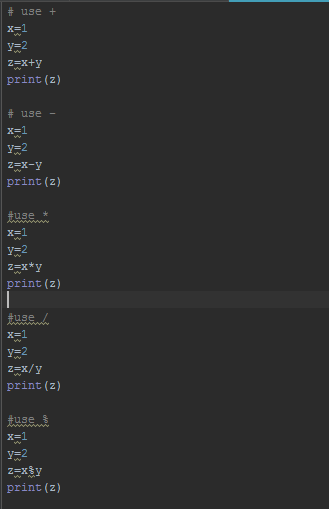
Replace every white-space character with the number 9:

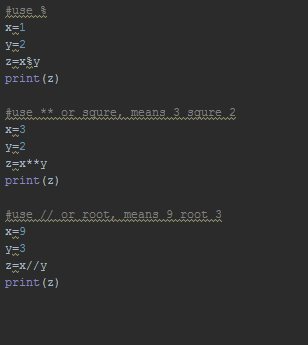


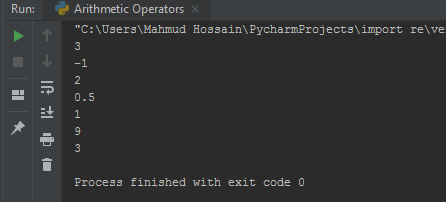


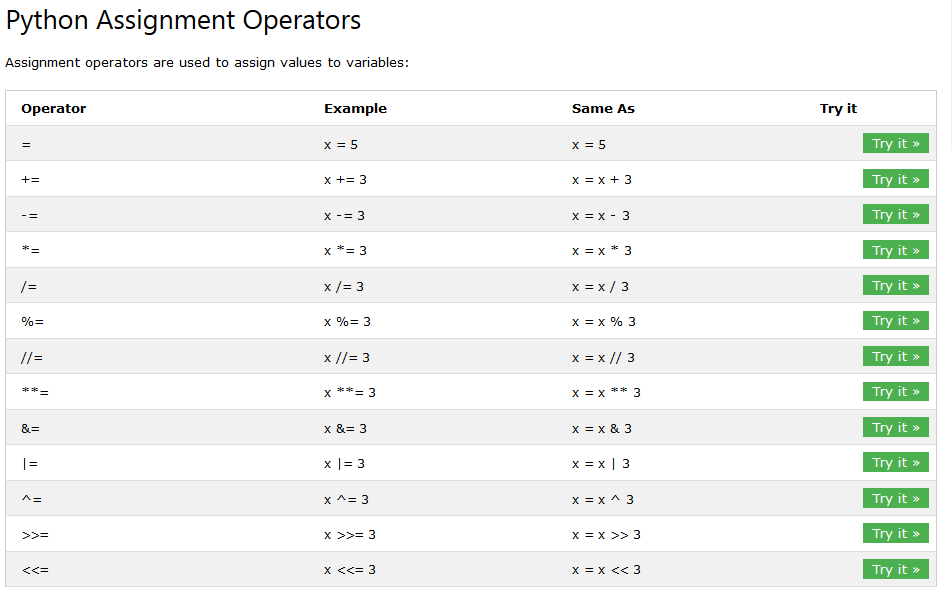


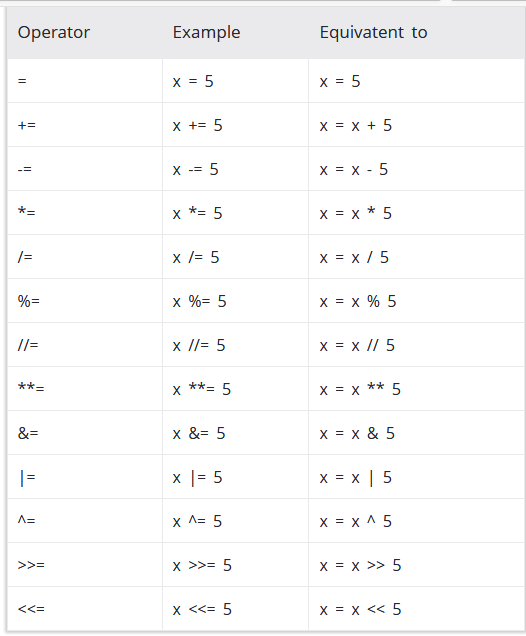


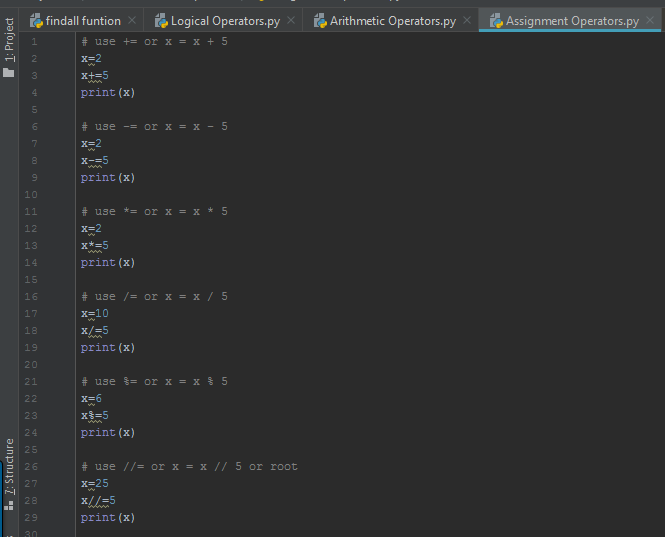


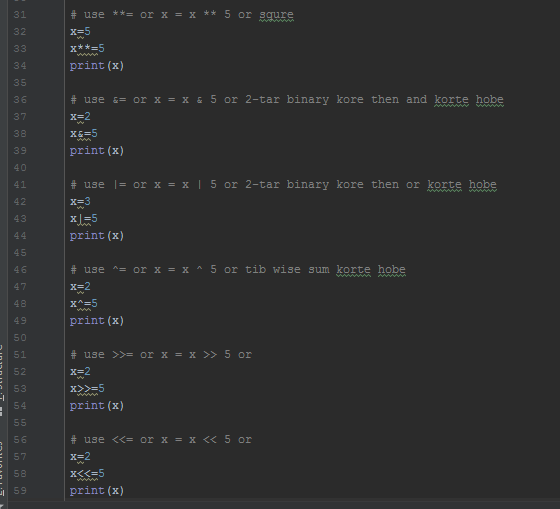


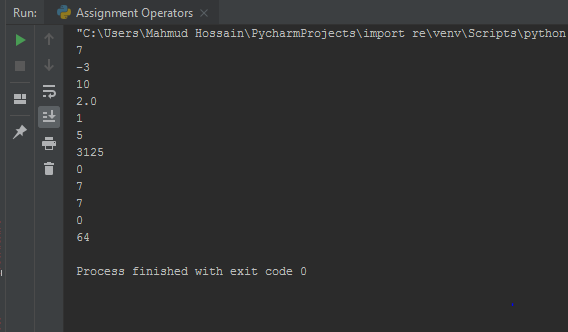


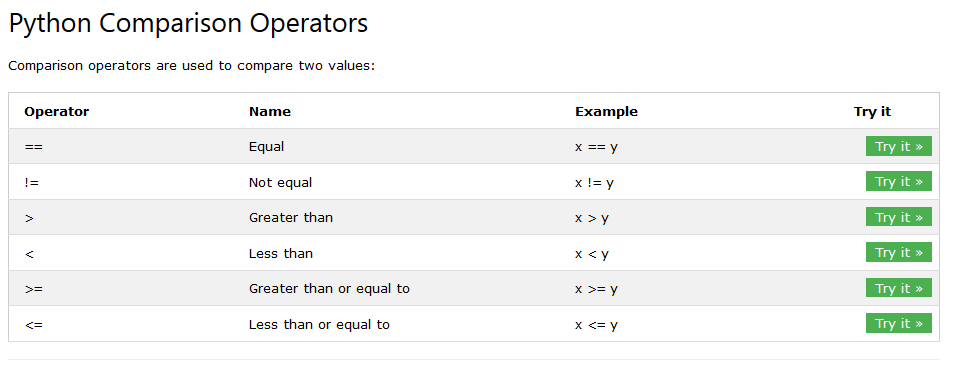


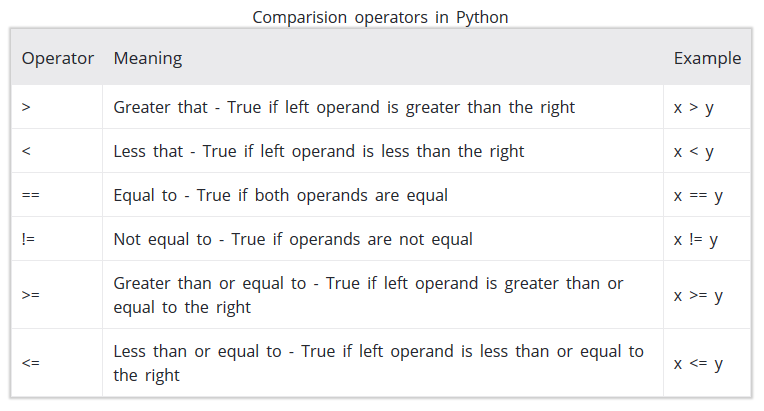


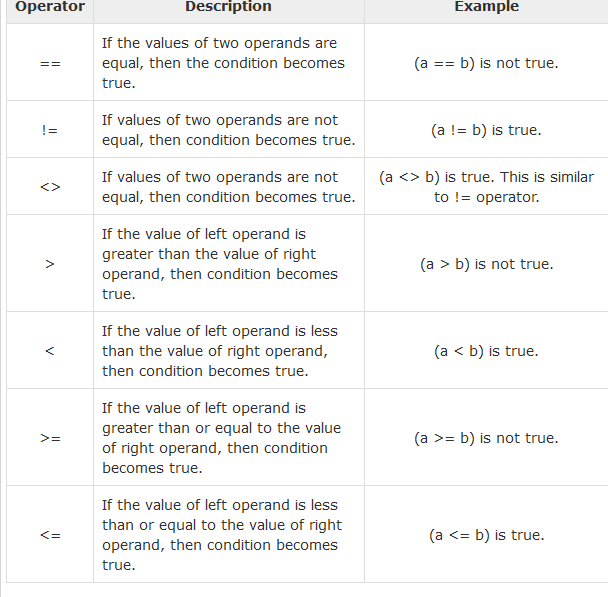


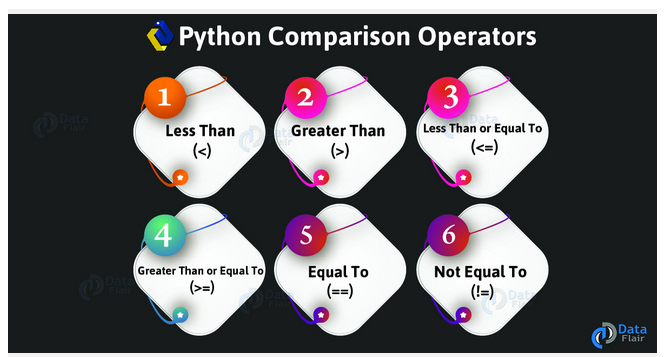




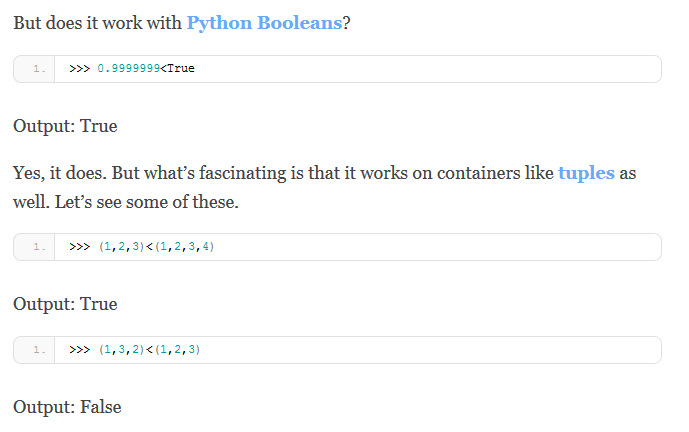


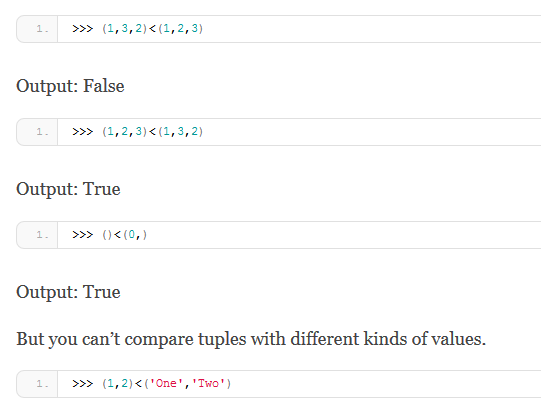


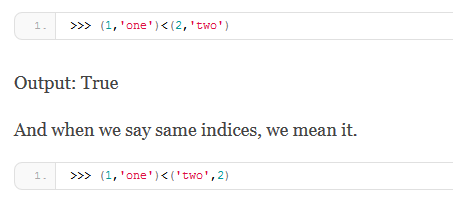


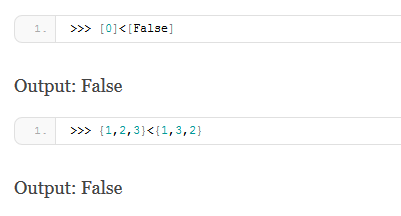


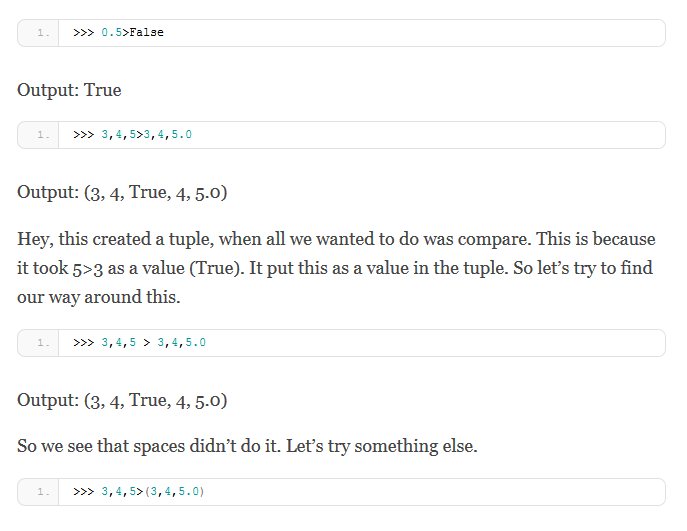


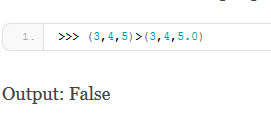


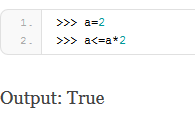


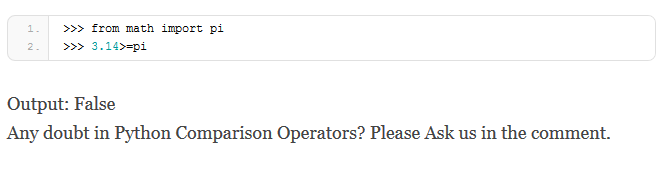


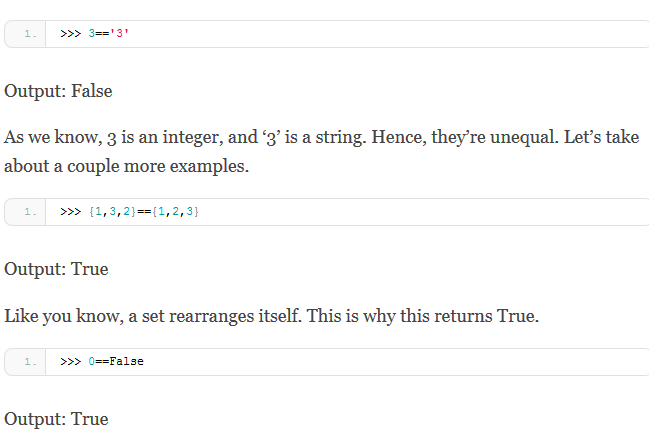




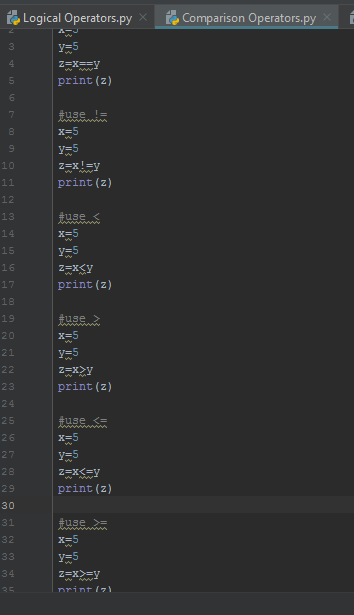


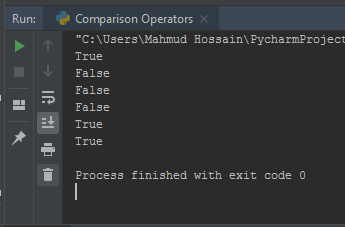


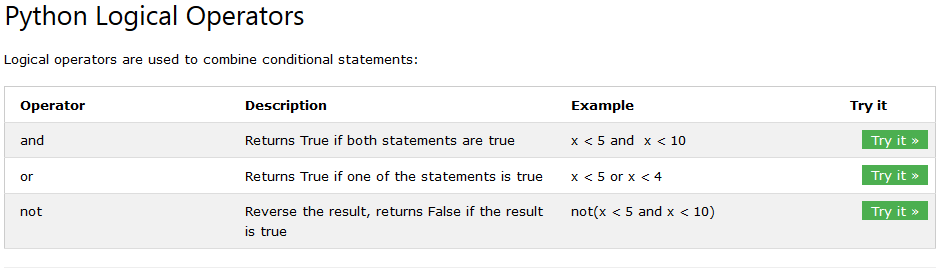


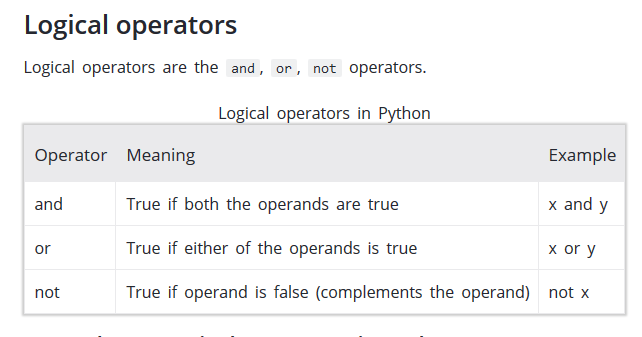


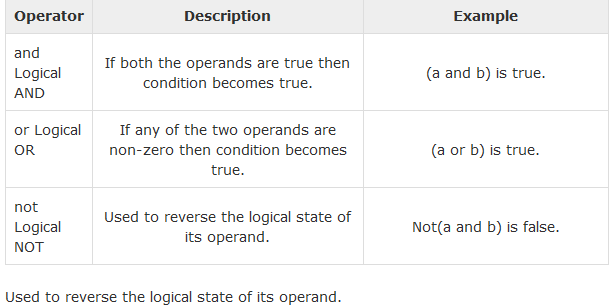


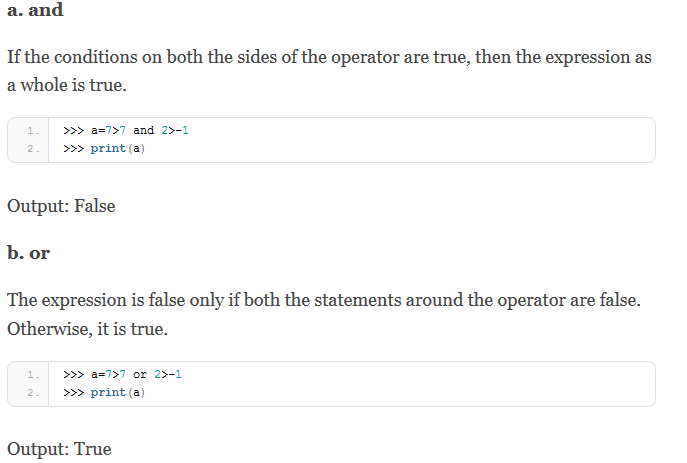


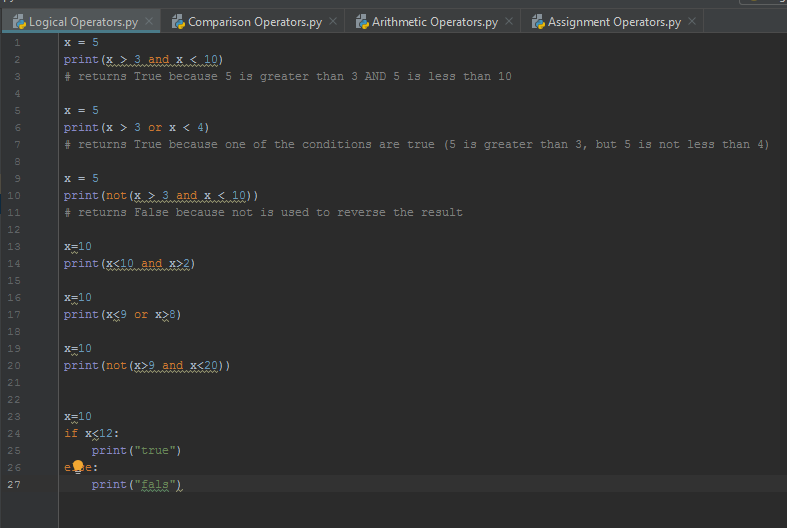


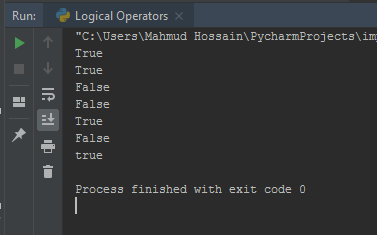


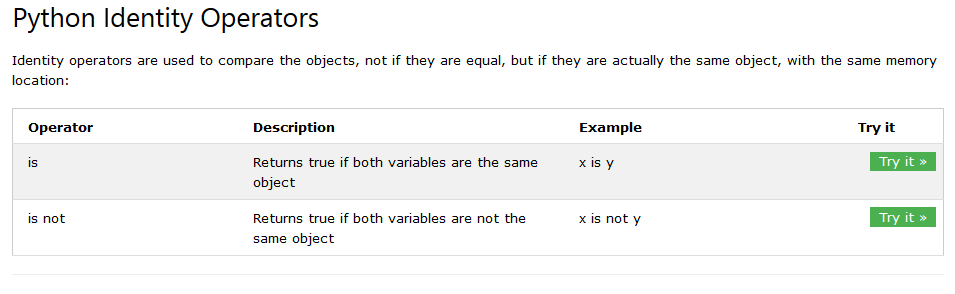


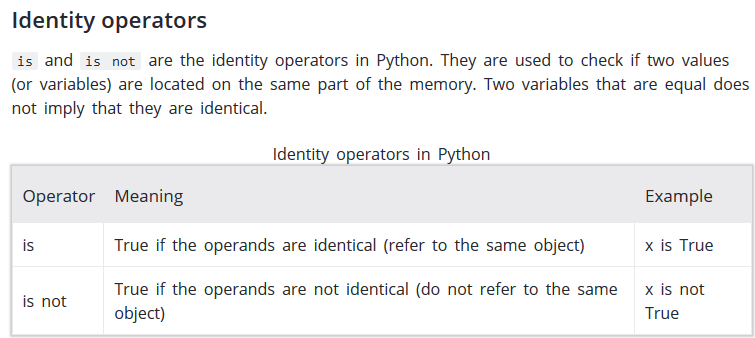


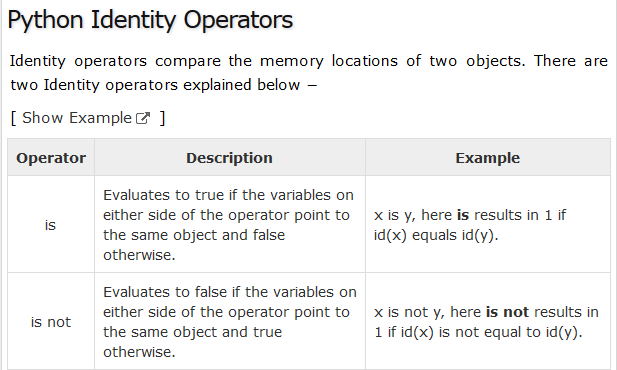


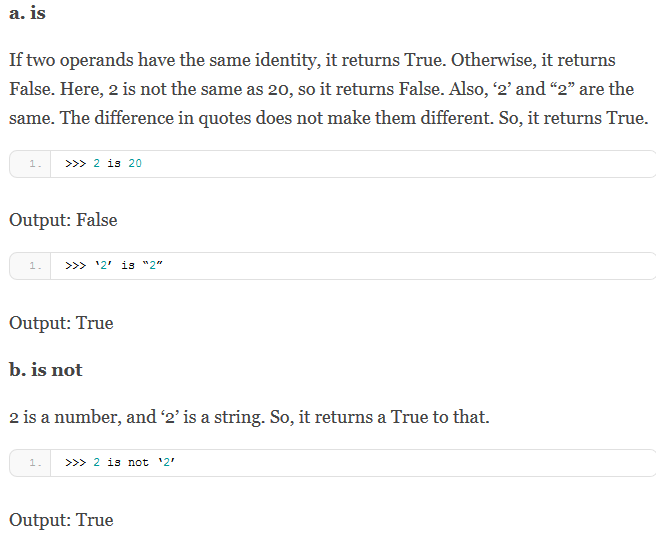


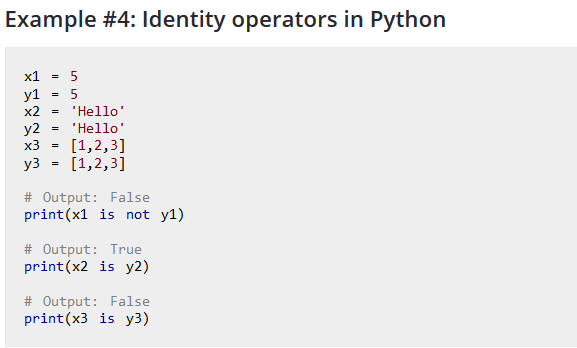


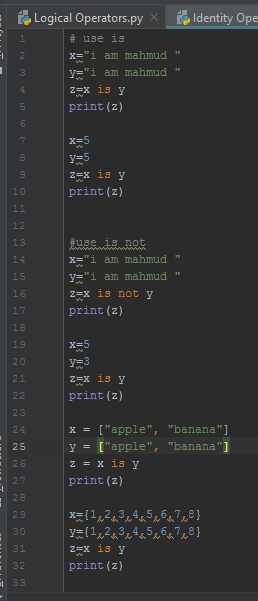


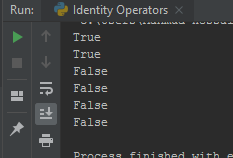


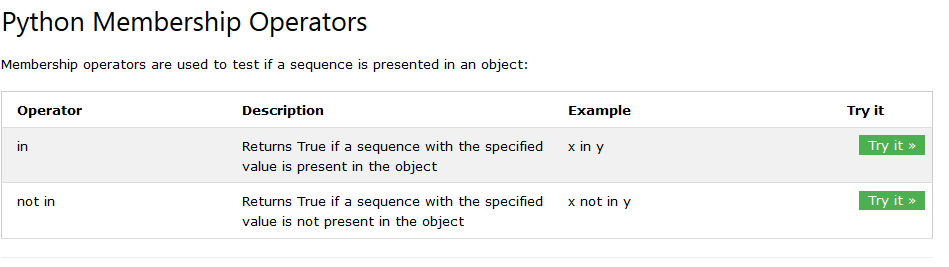


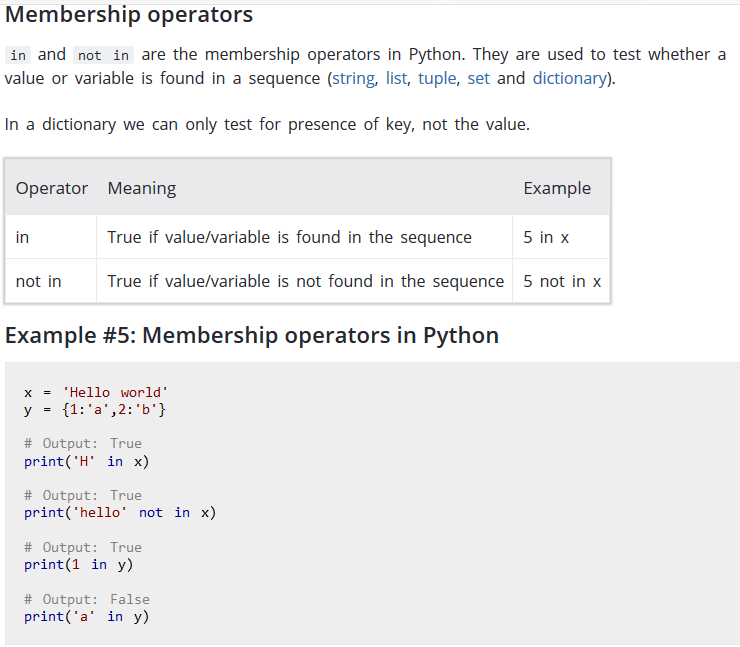


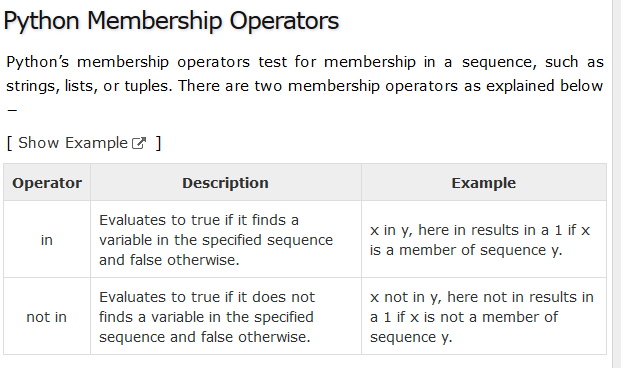


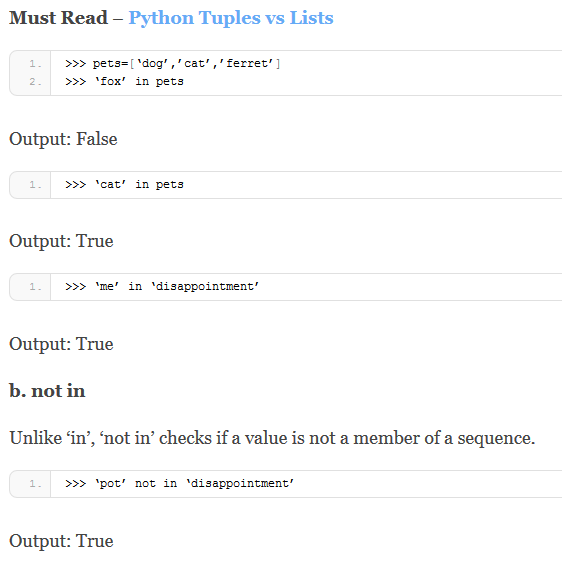


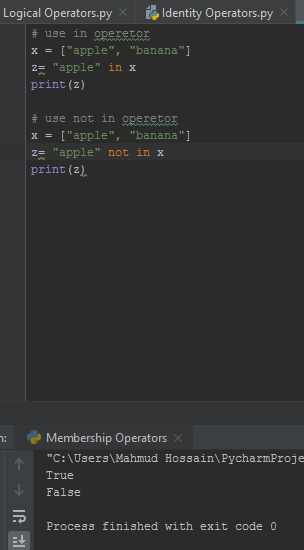


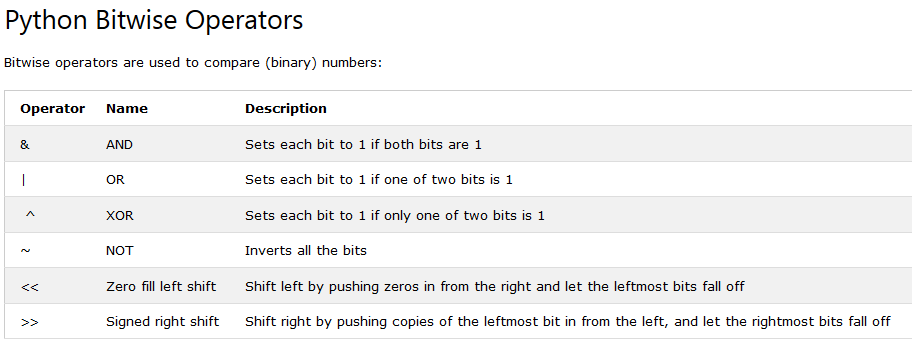


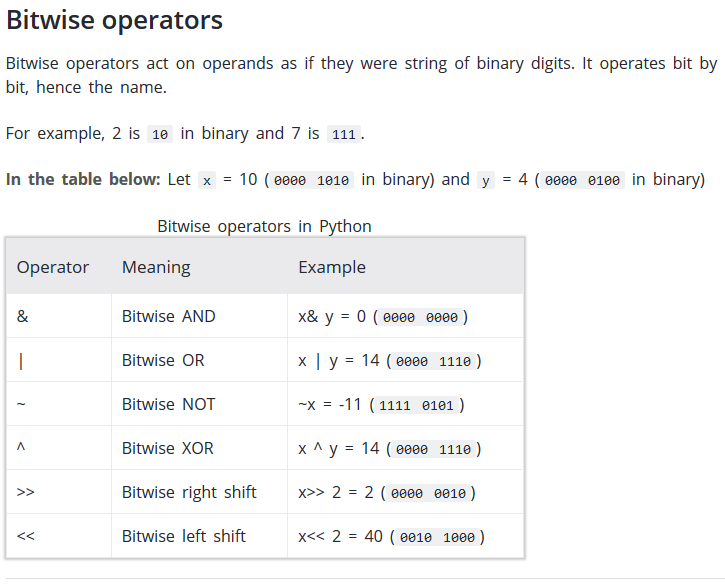


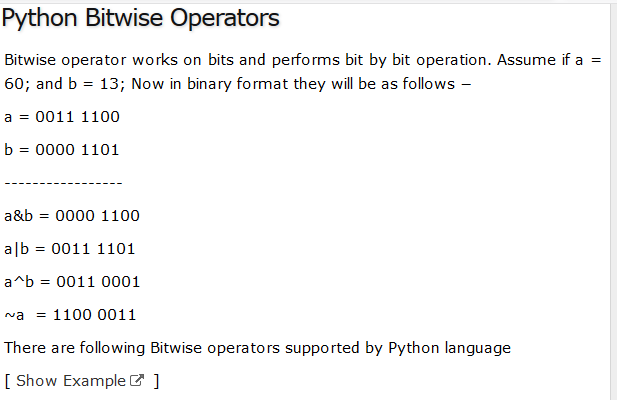


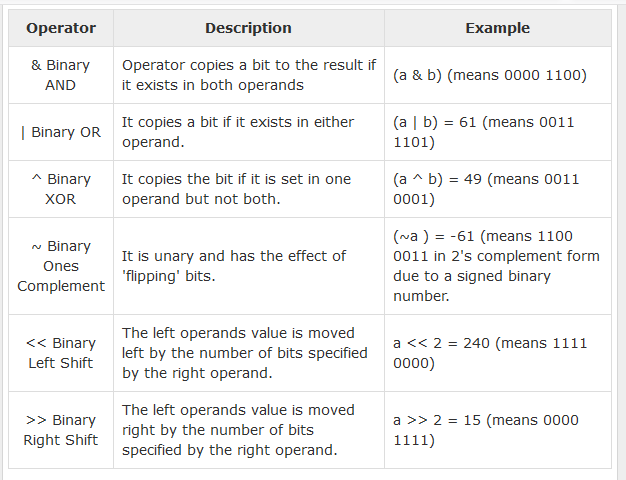


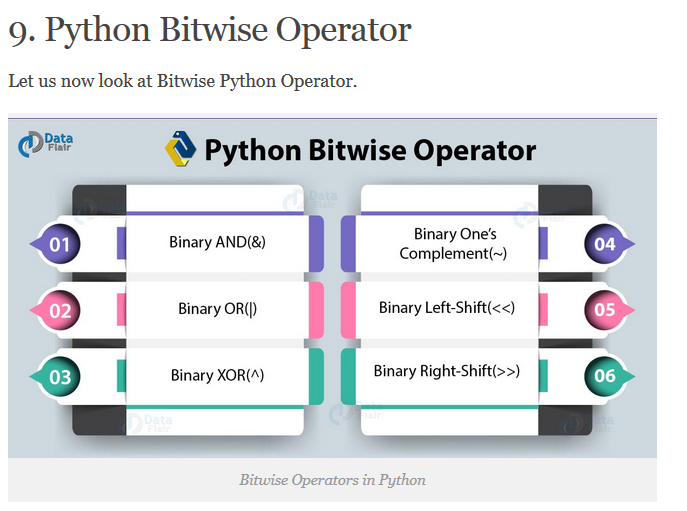


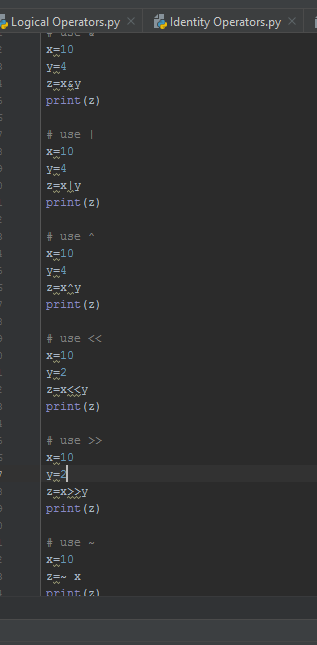


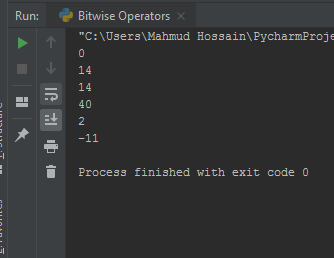




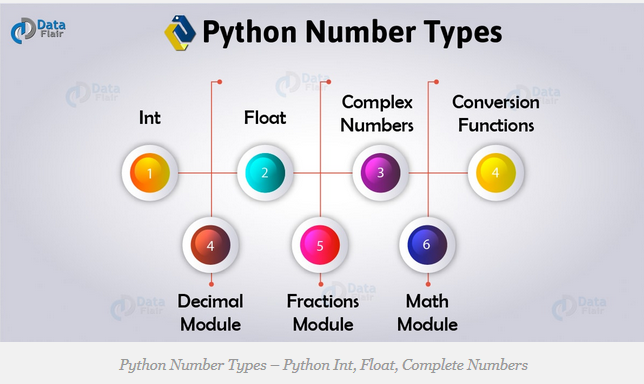








Python Number system with math function with math module



**math Module**

syntax and example:

# syntax  
*"""import math  
math.factorial(5)"""*import math  
x=math.tan(90)  
print(x)  
  
import math  
x=math.tan(45)  
print(x)

import math

x=math.tan(30)

print(x)

## fractions Module

syntax and example:

#syntax  
"""from fractions import Fraction  
(Fraction(1.5))"""  
  
from fractions import Fraction  
x=(Fraction(1.5))  
print(x)  
  
x=(Fraction(3.5))  
print(x)  
  
x=Fraction(1,3)  
print(x)

## Decimal Module

syntax and example:

## bin,oct,hexa convertion

x=13  
print(bin(x))  
print(oct(x))  
print(hex(x))

0b1101

0o15

0xd

Phython string

