In [1]:
```
print("Hello, world!")
```
```
Hello, world!
```

In [ ]:
```
#python is a scripting language which can be used for various development & analytical purpose
```

In [2]:
```
#data types

n1=10 #integer values(whole numbers)
```

In [3]:
```
#characters, string values

name="Harshit Shukla" #string data
```

In [4]:
```
#decimal numbers
f1=34.567 #float values
```

In [5]:
```
#store multiple together


#list of values

l1=[  1,"Harshit",23.456,10.91, -98,-87 ]
```

In [6]:
```
#tuple
tp=(1,"Harshit",23.45)
```

In [7]:
```
#dictionary

d1={ "id": 1 ,  "name": "Harshit", "data": 23.45  }
```

In [8]:
```
#mutable------->value that can be change(list, dictionaries)
```

In [9]:
```
# immutable------> cannot be changed(tuples, string)
```

In [10]:
```
print(  d1 )
```
```
{'id': 1, 'name': 'Harshit', 'data': 23.45}
```

In [12]:
```
print(  n1 ,                     name )
```
```
10 Harshit Shukla
```

In [ ]:
```
#separator option changes the characters used for separating items in a print statement
```

In [14]:
```
print(   l1, f1,sep="------" )
```
```
[1, 'Harshit', 23.456, 10.91, -98, -87]------34.567
```

In [15]:
```
print(   l1, f1,name,d1,sep="------" )
```
```
[1, 'Harshit', 23.456, 10.91, -98, -87]------34.567------Harshit Shukla------{'id': 1, 'name': 'Harshit', 'data':
23.45}
```

In [17]:
```
print(   l1, f1,name,d1,sep="^^^^^^" )
```
```
[1, 'Harshit', 23.456, 10.91, -98, -87]^^^^^^34.567^^^^^^Harshit Shukla^^^^^^{'id': 1, 'name': 'Harshit', 'data':
23.45}
```

In [19]:
```
print(   l1, f1,sep="------",end="&&&&&" )
print(   name,d1,sep="^^^^^^" )
```
```
[1, 'Harshit', 23.456, 10.91, -98, -87]------34.567&&&&&Harshit Shukla^^^^^^{'id': 1, 'name': 'Harshit', 'data':
23.45}
```

In [20]:
```
d1['name']
```

Out[20]: 'Harshit'

In [21]:
```
name = input("Enter your name: ") #name is data given as input
```
```
Enter your name: Harshit S
```

In [22]:
```
print(  name  )
```
```
Harshit S
```

In [ ]:
```
#use type function to verify data type
```

In [23]:
```
print (   type(f1)  )
```
```
<class 'float'>
```

In [24]:
```
print (   type(l1),type(d1)  )
```
```
<class 'list'> <class 'dict'>
```

In [25]:
```
print (   type(tp),type(name), sep="\n"  )
```
```
<class 'tuple'>
<class 'str'>
```

In [26]:
```
age=input("Enter your age: ")
```
```
Enter your age: 28
```

In [28]:
```
print(  type(age) )
```
```
<class 'str'>
```

In [29]:
```
age= int( input("Enter your age: " )  )
```
```
Enter your age: 28
```

In [30]:
```
print(  type(age) ) #
```
```
<class 'int'>
```

In [31]:
```
#arithmetic operations
print( 10 + 20 ) #addition
```
```
30
```

In [32]:
```
print( 10 - 20 ) #subtraction
```
```
-10
```

In [33]:
```
print( 10 * 20 ) #multiplication
```
```
200
```

In [34]:
```
print( 10 ** 2 ) #two stars is exponent operation ---> 10 raised to the power 2
```
```
100
```

```
In [35]:  print( 10 / 20 ) #division. float division

          0.5
```

```
In [36]:  print( 10 / 3 ) #division. float division

          3.3333333333333335
```

```
In [37]:  print( 10 // 3  ) #floor division

          3
```

```
In [39]:  #remainder of a division operation

          print( 10 % 3 ) #remainder.

          1
```

```
In [3]:    """
                   3  <--------------quotient

              -----------
           3  )   10
                - 9
                --------

                  1<-----------remainder

           """

          print("division example")

          division example
```

```
In [42]:  #take 2 numbers from the user. multiply and show result

          n1=int(input("Enter number 1: "))
          n2=int(input("Enter number 2: "))

          print("The result of " , n1 , " multiplied by " , n2, "is", n1*n2)

          Enter number 1: 10
          Enter number 2: 4
          The result of  10  multiplied by  4 is 40
```

```
In [43]:  #take 2 numbers from the user. multiply and show result

          n1=int(input("Enter number 1: "))
          n2=int(input("Enter number 2: "))

          print("The result of " , n1 , " multiplied by " , n2, "is", n1*n2)

          Enter number 1: 2
          Enter number 2: 3
          The result of  2  multiplied by  3 is 6
```

```
In [46]:  import sys
          sys.version

Out[46]:  '3.8.6 (default, Jan 27 2021, 15:42:20) \n[GCC 10.2.0]'
```

```
In [ ]:   # python 3.6+
```

```
In [45]:  print(  f"The result of {n1} multiplied by {n2} is {n1*n2}" ) #formatted string
```

```
The result of 2 multiplied by 3 is 6
```

In [48]:
```python
#conditional programming

age=int(input("Enter your age: "))

if age >= 18:
    print("You can vote")
```
```
Enter your age: 15
```

In [50]:
```python
#conditional programming

age=int(input("Enter your age: "))

if age >= 18:
    print("You can vote")
else:
    print("You are not allowed to vote as you not of voting age")
```
```
Enter your age: 25
You can vote
```

In [55]:
```python
age=int(input("Enter your age: "))
gender=input("Enter your gender: ")
if age < 18 and age > 0:
    print("You CANNOT MARRY AT THIS AGE")

elif age>=18 and gender == "female":
    print("you can legally marry in any state of India")

elif age>=21 and gender == "male":
    print("you can legally marry in any state of India")

elif age>=18 and age<=21 and gender=="male":
    print("You can marry once you turn 21")
else:
    print("Please check the age input given")
```
```
Enter your age: 31
Enter your gender: female
you can legally marry in any state of India
```

In [56]:
```python
import os
os.getcwd()
```
Out[56]:
```
'/home/harshit'
```

In [58]:
```python
n1=int(input("Enter a number: "))

# if you don't know the exact number of times a loop is to be executed, but you know the condition
#when to stop
while n1 < 10:
    print( n1 )
    n1=n1+1

print("hello")
```
```
Enter a number: 5
5
6
7
8
9
hello
```

In [59]:
```python
#for loop
"""
the exact iterations/cycle to be run should be clear
"""
#end value is not inclusive
#for(int count=1;count< 11 ; count++)
for count in range(1,11,1):
    print(count)
```

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

In [60]:

```
#for(count=10;count > 6; count --)
for count in range(10,6,-1):
    print(count)
```

```
10
9
8
7
```

In [61]:

```
#                     3   4    5  6  7  8  9  10

# 10

for count in range(10,2,-3):
    print(count)
```

```
10
7
4
```

In [62]:

```
#start value has a default of 0
#step value has a default of +1

for count in range(1,11):
    print(count)
```

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

In [63]:

```
for count in range(1,11):
    print(count,end="\t")
```

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

In [64]:

```
for count in range(11): #only one number means ending point
    print(count,end="\t")
```

```
0       1       2       3       4       5       6       7       8       9       10
```

In [65]:

```
print(   len("harshit") )
```

```
7
```

In [66]:

```
for index in range(  len("harshit")   ):
    print(index)
```

```
0
1
2
```

```
3
4
5
6
```

In [67]:
```python
#string class


name="Harshit Shukla"

print(f"{name} in upper case {name.upper()} ")
```
```
Harshit Shukla in upper case HARSHIT SHUKLA
```

In [68]:
```python
print(  "python".upper()  )
```
```
PYTHON
```

In [70]:
```python
print(  input("Enter a string: ").upper()  )
```
```
Enter a string: demo data
DEMO DATA
```

In [71]:
```python
print(  "PytHoN".lower() )
```
```
python
```

In [72]:
```python
print(  "TeMpDaTA".swapcase()   ) #interchange the casing
```
```
tEmPdAta
```

In [74]:
```python
help("data".count ) #don't call count function
```
```
Help on built-in function count:

count(...) method of builtins.str instance
    S.count(sub[, start[, end]]) -> int

    Return the number of non-overlapping occurrences of substring sub in
    string S[start:end].  Optional arguments start and end are
    interpreted as in slice notation.
```

In [76]:
```python
print(  "dataTodata".count("data") )
```
```
2
```

In [ ]:
```python
"""
upper
lower
swapcase
count
"""
```

In [77]:
```python
print(  "DataToData".replace( "a","X" )  ) #replace part of a string
```
```
DXtXToDXtX
```

In [78]:
```python
print(  "DataToData".find("To")  ) #first ocurrence from the start. gives index
```
```
4
```

In [79]:
```python
name=input("Enter your first name and last name together with a space in between: ")
```

```
In [79]: name=input("Enter your first name and last name together with a space in between: ")

         print( name.split(" ") )
```

```
Enter your first name and last name together with a space in between: Harshit Shukla
['Harshit', 'Shukla']
```

```
In [80]: print( input("Enter your first name and last name together with a space in between: ").split(" "))
```

```
Enter your first name and last name together with a space in between: Harshit Shukla
['Harshit', 'Shukla']
```

```
In [81]: msg="Enter your first name and last name together with a space in between: "
         firstName, lastName = input(msg).split(" ")
```

```
Enter your first name and last name together with a space in between: Harshit Shukla
```

```
In [82]: print(firstName)
```

```
Harshit
```

```
In [83]: print(lastName)
```

```
Shukla
```

```
In [84]: data=[ "this","is","a","test"   ]
         sentence=""
         for word in data:
             sentence= sentence+ word+ " "
         print(sentence)
```

```
this is a test
```

```
In [85]: print(  " ".join(data)   )
```

```
this is a test
```

```
In [86]: #trimming of data!!!!!


         name=input("Enter your name: ")
         print(name)
```

```
Enter your name:       Harshit Shukla
      Harshit Shukla
```

```
In [87]: name=input("Enter your name: ").strip() #trip and remove spaces from start and end
         print(name)
```

```
Enter your name:        harshit shukla
harshit shukla
```

```
In [90]: name=input("Enter your name: ").replace(" ","") #replace and remove spaces from start,middle and end
         print(name)
```

```
Enter your name:      h  a  r s h i t shukla
harshitshukla
```

```
In [88]: print(  "Harshit2378".isalpha()  )
```

```
False
```

```
In [89]: print(  "Harshit2378".isalnum()  )
```
```
True
```
```
In [91]: print(  "Harshit2   ^ & 378".isalnum()  )
```
```
False
```
```
In [93]: #program for checking if input given by the user has special characters


         if   input("enter a string: ").isalnum():
             print("No special characters found")

         else:
             print("SPECIAL CHARACTERS OBSERVED")
```
```
enter a string: har sh i t
SPECIAL CHARACTERS OBSERVED
```
```
In [97]: #for each loop

         for letter in "harshit":
             print(letter,end="---")
```
```
h---a---r---s---h---i---t---
```
```
In [98]: #vowels in a given string:

         vowels=['a','e','i','o','u']

         #for a letter in user input converted to lower case
         for letter in input("Enter a string: ").lower():
             #if the letter is also present in vowel list
             if letter in vowels:
                 print(letter) #print the letter as it is definitely a vowel
```
```
Enter a stringharshit
a
i
```
```
In [ ]: #indexing and slicing


        #providing a position and fetching the character at that position
```
```
In [ ]: """
            0         1            2           3            4          5           6
            h         a            r           s            h          i           t
           -7         -6           -5          -4           -3          -2        -1
        """
```
```
In [100…: name="harshit"
          print( name[5]  )
          print(name[-2])
```
```
i
i
```
```
In [101…: print(name[-6])
```
```
a
```
```
In [102…: #slicing part


          print(  name[  0:3:1   ]  )
```
```
har
```

In [103…
```
print(  name[  1:4:1   ]  )
```
```
ars
```

In [104…
```
print(  name[  1:6:2   ]  )
```
```
asi
```

In [105…
```
print(  name[  :6:2   ]  )
```
```
hrh
```

In [107…
```
print(  name[  -1:-4:-1   ]  )
```
```
tih
```

In [108…
```
print(  name[  ::-1   ]  )
```
```
tihsrah
```

In [109…
```
#program to reverse a user given string


print(   input("Enter a string")[ ::-1]  )
```
```
Enter a stringharshit shukla
alkuhs tihsrah
```

In [111…
```
#list objects


l1=[1,2,3,78.21,-98,78,"harshit",True,False, ['a','e','u']  ]
```

In [112…
```
print(l1[-1])
```
```
['a', 'e', 'u']
```

In [113…
```
print(   l1[1:4:1] )
```
```
[2, 3, 78.21]
```

In [114…
```
data=input("Enter some data in string form")

l1.append(data)
```
```
Enter some data in string formtemp
```

In [115…
```
print(l1)
```
```
[1, 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
```

In [116…
```
l1[0] = input("Enter some data in string form") #lists are mutable
print(l1)
```
```
Enter some data in string formpython
['python', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
```

In [117…
```
copy_list=l1.copy() #an actual copy
```

In [120…
```
print(copy_list)
print(l1)

print(id(copy_list)) #memory address
print(id(l1))
```

```
['python', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
['python', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
140163718752832
140163716094016
```

In [121…
```
temp = l1 #alternate name
print(temp)
print(l1)
print(id(temp)) #memory address
print(id(l1))
```

```
['python', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
['python', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
140163716094016
140163716094016
```

In [122…
```
temp[0]="XYZ"
print(temp)
print(l1)
```

```
['XYZ', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
['XYZ', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
```

In [ ]:
```
     temp--------------->[                   ]<--------------l1
```

In [123…
```
copy_list.clear() #removing all elements from the list. Truncating operation

print(copy_list)
```

```
[]
```

In [124…
```
print( l1 )
```

```
['XYZ', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
```

In [125…
```
#reverse the list IN-PLACE
print( l1 , id(l1),sep="\n")


l1.reverse() #reverse the given list

print( l1 , id(l1),sep="\n")
```

```
['XYZ', 2, 3, 78.21, -98, 78, 'harshit', True, False, ['a', 'e', 'u'], 'temp']
140163716094016
['temp', ['a', 'e', 'u'], False, True, 'harshit', 78, -98, 78.21, 3, 2, 'XYZ']
140163716094016
```

In [ ]:
```
"""
copy
append
clear
reverse
extend
"""
```

In [126…
```
name="harshit"
print(id(name))
```

```
140163846034096
```

In [128…
```
copy_string=name[::-1]
print(copy_string)
print(id(copy_string))
```

```
tihsrah
```

```
140163716142960
```

In [129…

```
l1=[9,8,7,6]

another=[  10,20,30,40 ] #separate list

l1.extend(  another ) #extend the list l1 to include all values from another
print(l1)
```

```
[9, 8, 7, 6, 10, 20, 30, 40]
```

In [ ]:

```
#list in python is a doubly linked list implementation
```

In [134…

```
#append--->add element to the end!

l1.insert(2,23.4567)
print(l1)
```

```
[9, 8, 23.4567, 23.4567, 23.4567, 23.4567, 23.4567, 7, 6, 10, 20, 30, 40]
```

In [135…

```
#deleting an entry from the list in 2 ways

#first---->specify the index

l1.pop( 2 )
print(l1)
```

```
[9, 8, 23.4567, 23.4567, 23.4567, 23.4567, 7, 6, 10, 20, 30, 40]
```

In [136…

```
l1.pop(0)
print(l1)
```

```
[8, 23.4567, 23.4567, 23.4567, 23.4567, 7, 6, 10, 20, 30, 40]
```

In [137…

```
item = l1.pop(-2)
print(item)

print(l1)
```

```
30
[8, 23.4567, 23.4567, 23.4567, 23.4567, 7, 6, 10, 20, 40]
```

In [138…

```
#second method-----> specify the name of the item

l1.remove(7)
print(l1)
```

```
[8, 23.4567, 23.4567, 23.4567, 23.4567, 6, 10, 20, 40]
```

In [ ]:

```
"""
copy
append
clear
reverse
extend
insert
pop
remove
"""
```

In [139…

```
#sorting

data=[    -19,-17,21,13,10,9,-4    ]

data.sort(reverse=False) #ascending order

print(data)
```

```
[-19, -17, -4, 9, 10, 13, 21]
```

In [140…
```
data=[    -19,-17,21,13,10,9,-4   ]

data.sort(reverse=True) #descending order

print(data)
```
```
[21, 13, 10, 9, -4, -17, -19]
```

In [142…
```
#can't sort a list of string and numbers mixed
test=[ 1,-1,10, "harshit","ytry"]

test.sort(reverse=False) #cannot sort string and integers
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-142-9b1648953ee1> in <module>
      1 test=[ 1,-1,10, "harshit","ytry"]
      2 
----> 3 test.sort(reverse=False)

TypeError: '<' not supported between instances of 'str' and 'int'
```

In [143…
```
students=[   [1,"Akshay"],[3,"John"],[2,"Hiten"]                       ]

students.sort(  key= lambda x : x[0]  )
print(students)
```
```
[[1, 'Akshay'], [2, 'Hiten'], [3, 'John']]
```

In [144…
```
students=[   [1,"Akshay"],[3,"John"],[2,"Hiten"]                       ]

students.sort(  key= lambda x : x[1]  )
print(students)
```
```
[[1, 'Akshay'], [2, 'Hiten'], [3, 'John']]
```

In [145…
```
employees=[   [11,"Akshay",24000],
              [33,"John",18000],
              [22,"Hiten",50000]
          ]


#sort this list in ascending order of their salaries

employees.sort(  key= lambda x : x[2]  )
print(employees)
```
```
[[33, 'John', 18000], [11, 'Akshay', 24000], [22, 'Hiten', 50000]]
```

In [146…
```
#a dictionary is a pair of key and value
student={  "id": 1 , "name" : "John", "age": 23 }

print(  student.keys()  ) #all the keys from the dictionary
```
```
dict_keys(['id', 'name', 'age'])
```

In [147…
```
for key in student.keys():
    print(key)
```
```
id
name
age
```

In [148…
```
print(  student.values() )
```
```
dict_values([1, 'John', 23])
```

In [149…
```
for value in student.values():
    print(value)
```

```
1
John
23
```

In [150…

```
print( student.items() )
```

```
dict_items([('id', 1), ('name', 'John'), ('age', 23)])
```

In [151…

```
for  key, value in student.items():
    print(f"{key}----->{value}")
```

```
id----->1
name----->John
age----->23
```

In [152…

```
#example---->return the name of the student

print( student[ 'name' ] )
```

```
John
```

In [153…

```
print( student[ 'age' ] )
```

```
23
```

In [154…

```
print( student[ 'xyz' ] )
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-154-98d67359171c> in <module>
----> 1 print( student[ 'xyz' ] )

KeyError: 'xyz'
```

In [156…

```
#take 2 labels from the users one by one. show the corresponding values


key1=input("Enter your key: ")
print(student[key1])

key2=input("Enter your key: ")
print(student[key2])
```

```
Enter your key: xyz
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-156-814cb74e5bec> in <module>
      3
      4 key1=input("Enter your key: ")
----> 5 print(student[key1])
      6
      7 key2=input("Enter your key: ")

KeyError: 'xyz'
```

In [157…

```
key1=input("Enter your key: ")
print( student.get(key1, "key not found")  )

key2=input("Enter your key: ")
print(student.get(key2, "key not found"))
```

```
Enter your key: xyz
key not found
Enter your key: age
23
```

In [158…
```
#modify / add entries to dictionary


student.update(  {"marks":[   23,45,67  ] }   )
print(student)
```
```
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67]}
```

In [159…
```
student.update(  {"gender":"male" }   )
print(student)
```
```
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67], 'gender': 'male'}
```

In [160…
```
student.update(  {"gender":"DEMO DATA" }   )
print(student)
```
```
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67], 'gender': 'DEMO DATA'}
```

In [161…
```
student.setdefault(  'gender', "female"  )
```
Out[161…
```
'DEMO DATA'
```

In [162…
```
print(student)
```
```
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67], 'gender': 'DEMO DATA'}
```

In [163…
```
student.setdefault(  'language', "english"  )
```
Out[163…
```
'english'
```

In [164…
```
print(student)
```
```
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67], 'gender': 'DEMO DATA', 'language': 'english'}
```

In [165…
```
#remove entry by specifying key

removed_val =  student.pop('gender', "KEY IS NOT FOUND")
print(removed_val)
```
```
DEMO DATA
```

In [166…
```
print(student)
```
```
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67], 'language': 'english'}
```

In [167…
```
removed_val =  student.pop('xyz', "KEY IS NOT FOUND")
print(removed_val)
print(student)
```
```
KEY IS NOT FOUND
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67], 'language': 'english'}
```

In [168…
```
#popitem

removed_item= student.popitem()
print(removed_item)
```
```
('language', 'english')
```

In [169…
```
print(student)
```

```
{'id': 1, 'name': 'John', 'age': 23, 'marks': [23, 45, 67]}
```

In [176…
```
keys=[1,2,3]
values=["harshit","mayur","arun"]

example={}
for k,v in zip(keys,values):
    example.update(  { k : v  }  )

print(example)
```
```
{1: 'harshit', 2: 'mayur', 3: 'arun'}
```

In [179…
```
employees={
"mumbai":    {1: 'harshit', 2: 'mayur', 3: 'arun'},

 "delhi":   {1: 'john', 2: 'mathew', 3: 'jacob'}

}

print(employees)
```
```
{'mumbai': {1: 'harshit', 2: 'mayur', 3: 'arun'}, 'delhi': {1: 'john', 2: 'mathew', 3: 'jacob'}}
```

In [180…
```
import pandas as pd
pd.DataFrame(employees)
```
Out[180…

	mumbai	delhi
1	harshit	john
2	mayur	mathew
3	arun	jacob

In [184…
```
#tuple---->immutable data type

tp=("23.45N","45.67W")
```

In [185…
```
print(tp)
```
```
('23.45N', '45.67W')
```

In [186…
```
print(tp[-1])
```
```
45.67W
```

In [187…
```
print(tp[::-1])
```
```
('45.67W', '23.45N')
```

In [188…
```
tp.count("23.45N")
```
Out[188… 1

In [189…
```
tp.index("23.45N")
```
Out[189… 0

In [190…
```
tp[0] = "33.96N"
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-190-94630f4582fe> in <module>
----> 1 tp[0] = "33.96N"
```

```
TypeError: 'tuple' object does not support item assignment
```

In [ ]:
```
#function is a task designed for some purpose
```

In [191…
```
def magic():
    print("this is line 1")
    print("magic called")
```

In [192…
```
print(10 % 3)

print("hello")


if 1 > 3:
    print("do something")

magic()
```

```
1
hello
this is line 1
magic called
```

In [193…
```
#addition function for accepting 2 numbers and printing the result

def addition(x , y):
    print( x  + y )
```

In [194…
```
addition( 10,20   )

addition( 10.21,20.789   )

addition("python", " language")

addition(  (1,2) , (3,4)  )
```

```
30
30.999000000000002
python language
(1, 2, 3, 4)
```

In [ ]:
```
l1.sort() #sort method of list class
```

In [ ]:
```
sort = sorted(tp) #sorted function of python standard library
sort
```

In [ ]:
```
#addition function for accepting 2 numbers and printing the result

def addition(x , y):
    print( x  + y )
```

In [195…
```
#subtraction function for accepting 2 numbers and returning the difference

def subtraction(x , y):
    return  x  - y


print ( subtraction(10,5) )
```

```
5
```

In [196…
```
#how to pass arguments as keywords?

def subtraction(x , y):
    return  x  - y


print( subtraction( 10, 5   ) )
print(  subtraction(5,10) )
```

```
5
-5
```

In [197…
```
#how to pass arguments as keywords?

def subtraction(x , y):
    return  x  - y

#keyword arguments
print( subtraction( x=10, y=5   ) )
print(  subtraction(y=5,x=10) )
```
```
5
5
```

In [198…
```
#default values
def subtraction(x=10 , y=5):
    return  x  - y

#keyword arguments
print( subtraction( x=20, y=6   ) )
print(  subtraction(y=6,x=0) )
```
```
14
14
```

In [ ]:
```
l1.sort()
```

In [199…
```
print(  subtraction(y=6) )
```
```
4
```

In [200…
```
addition(10) #this will not work as addition has no default
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-200-dca566d4b68d> in <module>
----> 1 addition(10)

TypeError: addition() missing 1 required positional argument: 'y'
```

In [203…
```
#variable length arguments

# def add(x,y):
#     return x+y


def add( *args  ):  #a tuple called args
    print(f"tuple is {args}")
    sum=0
    for value in args:
        sum=sum+value
    print(sum)

add(10)
add(20,10)
add(30,40,10)
add(67,18.-19,20,10,20,30)
add()
```
```
tuple is (10,)
10
tuple is (20, 10)
30
tuple is (30, 40, 10)
80
tuple is (67, -1.0, 20, 10, 20, 30)
146.0
tuple is ()
0
```

In [207…
```
#variable length arguments

# def add(x,y):
#     return x+y


def subtract( x,*args  ):  #a tuple called args
```

```
    print(f"tuple is {args}")
    total=0
    for value in args:
        total=total+value
    print(total)

    print(   x - total )


subtract( 10,20,30   )

subtract(10)

subtract()
```

```
tuple is (20, 30)
50
-40
tuple is ()
0
10
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-207-63dc6c35d553> in <module>
     19 subtract(10)
     20
---> 21 subtract()

TypeError: subtract() missing 1 required positional argument: 'x'
```

In [210…

```
def addition(x,y,z):
    print(x+y+z)

tp=(10,20,30)
addition( *tp )
```

```
60
```

In [ ]:

```
#first-class-function
```

In [211…

```
print( type(addition))
```

```
<class 'function'>
```

In [212…

```
#you can do everything with a function object that you can normally do with object any other type

#functions can be stored in variables

f1=addition
```

In [213…

```
addition(10,20,30)
```

```
60
```

In [214…

```
f1(10,20,30)
```

```
60
```

In [215…

```
f2 =  lambda x,y,z : print(x+y+z)

f2(10,20,30)
```

```
60
```

In [218…

```
def helper(  l1 , f1  ):
    for num in l1:
        f1(num)
```

In [219…

```
def percent90(x):
    print(0.9*x)
```

```
def sqrt(x):
    print(  x**(1/2) ) #any number raised to the power of 1/2 gives its square root
```

In [220…

```
helper( [1,2,3,4,5] , percent90    ) ##WOW! function passed to funtion
```

```
0.9
1.8
2.7
3.6
4.5
```

In [221…

```
helper([1,2,3,4,5], sqrt)
```

```
1.0
1.4142135623730951
1.7320508075688772
2.0
2.23606797749979
```

In [222…

```
#store functions in containers??

func=[  sqrt,percent90  ]

print(func)
```

```
[<function sqrt at 0x7f7a40353c10>, <function percent90 at 0x7f7a40353d30>]
```

In [223…

```
func[-1](1000.00)
```

```
900.0
```