In [135]:
```
#different between c, java, python

c-int a=10;
printf("",a)

java-int

a=10
system.out.printin(""+a);

python-int
a=10
print(a)
```

```
  File "C:\Users\exam3\AppData\Local\Temp/ipykernel_1460/3547167734.py", line 3
    c-int a=10;
          ^
SyntaxError: invalid syntax
```

In [ ]:
```
# hello world program
print("salman khan")
```

In [ ]:
```
# Assign a variable to values

a="rvr.jc college"
print(a)
```

In [2]:
```
a*10
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_5200/226977657.py in <module>
----> 1 a*10

NameError: name 'a' is not defined
```

In [140]:
```
print('khan\n'*10)
```

```
khan
khan
khan
khan
khan
khan
khan
khan
khan
khan
```

In [139]:

```
35
```

In [138]:
```
s=25
c=35
p=s-c
print(p)
```

```
-10
```

In [ ]:
```
(p)
```

In [137]:
```
p=20
s=15
n=p*s
print(n)
```

```
300
```

In [141]:
```
a=350
b=5
c=a/b
print(c)
```

```
70.0
```

In [142]:
```
# change a string to lower to upper
string='salman'
string.upper()
```

Out[142]:
```
'SALMAN'
```

In [143]:
```
string='SALMAN'
string.lower()
```

Out[143]:
```
'salman'
```

In [144]:
```
# stirng
a="patan"
b="salman"
c="khan"
d=a+b+c
print(d)
```

```
patansalmankhan
```

In [ ]:
```
# accessing first element of a given string
a="salmankhan"
a[0]
```

In [145]:
```
# accessing last element of a given string
a="salmankhan"
a[-1]
```

Out[145]:
```
'n'
```
In [146]:
```
a='saida'
a[::-1]
```
Out[146]:
```
'adias'
```
In [147]:
```
# length of the given string
a="salman"
print(len(a))
```
```
6
```
In [148]:
```
a[0:6]
```
Out[148]:
```
'salman'
```
In [149]:
```
a='gopal'
a[2:5]
```
Out[149]:
```
'pal'
```
In [150]:
```
# dynamic values additions
a=10
b=20
c=a+b
print(c)
```
```
30
```
In [151]:
```
a=int(input("enter a value"))
b=int(input("enter b value"))
c=a+b
print("additional of two numbers A&B is:",c)
```
```
enter a value20
enter b value20
additional of two numbers A&B is: 40
```
In [152]:
```
a=int(input("enter a value"))
b=int(input("enter b vlue"))
c=a-b
print(c)
```
```
enter a value20
enter b vlue30
-10
```
In [153]:
```
a=int(input("enter a value"))
b=int(input("enter b value"))
c=a*b
print(c)
```

In [154]:
```
a=int(input("entr a value"))
b=int(input("entr b value"))
c=a/b
print(c)
```

```
entr a value50
entr b value30
1.6666666666666667
```

In [155]:
```
# how to multiplication table in python

n=12
for i in range(1,11):
    print(n,'*',i,'=',n*i)
```

```
12 * 1 = 12
12 * 2 = 24
12 * 3 = 36
12 * 4 = 48
12 * 5 = 60
12 * 6 = 72
12 * 7 = 84
12 * 8 = 96
12 * 9 = 108
12 * 10 = 120
```

In [156]:
```
n=1
for i in range(1,11):
    print(n,'*',i,'=',n*i)
```

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

In [157]:
```
n=2
for i in range(1,11):
    print(n,'*',i,'=',n*i)
```

```
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
```

In [158]:
```
n=3
for i in range(1,11):
    print(n,'*',i,'=',n*i)
```

```
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
```

In [160]:
```
n=120
for i in range(1,11):
    print(n,'*',i,'=',n*i)
```

```
120 * 1 = 120
120 * 2 = 240
120 * 3 = 360
120 * 4 = 480
120 * 5 = 600
120 * 6 = 720
120 * 7 = 840
120 * 8 = 960
120 * 9 = 1080
120 * 10 = 1200
```

In [161]:
```
n=11
for i in range(1,11):
    print(n,'*',i,'=',n*i)
```

```
11 * 1 = 11
11 * 2 = 22
11 * 3 = 33
11 * 4 = 44
11 * 5 = 55
11 * 6 = 66
11 * 7 = 77
11 * 8 = 88
11 * 9 = 99
11 * 10 = 110
```

In [162]:
```
n=int(input("entr a value"))
for i in range(1,11):
    print(n,'*',i,'=',n*i)
```

```
entr a value20
20 * 1 = 20
20 * 2 = 40
```

this is a comment

this is a comment

this is a comment

this is a comment

this is a comment

this is a comment

this is a comment

In [163]:
```
# to print the 1-10 natural numbars by using "for" loop
```

In [164]:
```
for i in range(11):
    print(i,end=" ")
```

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

In [ ]:
```
# TO GIVE THE step value to print the odd no's sterting from 1-100
```

In [27]:
```
for i in range(1,100,2):
  print(i,end=" ")
```

```
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61
63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

In [ ]:
```
# to print the values starting charachter 0 ending charachter 50
```

In [28]:
```
for i in range(0,50,3):
     print(i,end=" ")
```

```
0 3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48
```

In [ ]:
```
# to print the natural numbars in ascending order
```

In [29]:
```
for i in range(1,51,1):
    print(i,end=" ")
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
```

In [30]:
```
n=int(input("enter a natural numbers size"))
for i in range(1,n+1):
    print(i,end=" ")
```

```
enter a natural numbers size20
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

In [ ]:

In [ ]:

```
enter a natural numbers size20
20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```
In [32]:
```
n=int(input("enter a natural numbers size"))
for i in range(n,0,-1):
   print(i,end=" ")
```

```
enter a natural numbers size50
50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22
21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```
In [ ]:
```
# break statement example in python
```

In [3]:
```
print("salman")
```

```
salman
```
In [18]:
```
for i in 'apssdc':
    if i=='d':
        break
    else:
        print(i,end=" ")
```

```
a p s s
```
In [36]:
```
for i in '12345678910':
    if i=='5':
        break
    else:
        print(i,end=" ")
```

```
1 2 3 4
```
In [38]:
```
hi='salman'
print(hi)
```

```
salman
```
In [39]:
```
# to print the even numbers in between1 to 20 using continue
```

In [51]:
```
for i in range(1,51):
    if (i%2!=0):
        continue
    else:
        print(i,end=" ")
```

```
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50
```
In [57]:
```
# swap between two numbers
```

a=5 b=10

In [69]:
```
a = 25
b = 30
a,b =a,b
print("a =", b)
print("b =", a)
```

```
a = 30
b = 25
```

In [71]:
```
# how to generat a number random number
```

In [82]:
```
import random
a=(random.randint(11,100))
print(a)
```

```
62
```

In [83]:
```
# to print the alphabet in python
```

In [98]:
```
print('Uppercase Alphabets:')
upperAlpha()

print('Lowercase Alphabets:')
lowerAlpha()
```

```
Uppercase Alphabets:
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Lowercase Alphabets:
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

In [99]:
```
upperAlpha()
lowerAlpha()
```

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

In [100]:
```
upperAlpha()
lowerAlpha()
```

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

In [101]:
```
# program to desplaycalender of the given month and year
```

In [125]:
```
import calendar
year=1998
month=7
print(calendar.month(year,month))
```

functions

1.resuability of the code

2.easy debugging

function is a group of statments it can perform one specific task

function keyword def in python by using "def" keyword we can perform the function

syntax:

def function_name(argument_list): statmnets

return value.

1. types of functions:

1. with arguments and with return values syntax:

def function the additon of two numbers?

n2=int(input("enter n1 value")) n2=int(input("enter n2 value")) def addition(a,b): c=a+b return c addition(n1,n2)

In [4]:
```
n1=int(input("enter n1 value"))
n2=int(input("enter n2 value"))
def addition(a,b):
    c=a+b
    return c
addition(n1,n2)
```

```
enter n1 value20
enter n2 value30
```
Out[4]:50

1. with arguments and with out return values

def function_name(argument_list): statements print values

to perfom the subtraction of two numbers?

n1=int(input("enter n1 value")) n2=int(input("enter n2 value"))

c=a+b print c

In [33]:
```
n1=int(input("enter n1 value"))
n2=int(input("enter n2 value"))
def subtraction(a,b):
    c=a-b
    print(c)
subtraction(n1,n2)
```

In [40]:
```
n1=int(input("enter n1 value"))
n2=int(input("enter n2 value"))
def subtraction(a,b):
        c=a-b
        print(c)
        subtraction(n1,n2)
```

```
enter n1 value25
enter n2 value20
```

In [41]:
```
n1=int(input("enter n1 value"))
n2=int(input("enter n2 value"))
def addition(a,b):
    c=a+b
    return c
addition(n1,n2)
```

```
enter n1 value30
enter n2 value25
```

Out[41]:55

In [94]:
```
def adding ():
    a=20
    b=30
    sum=a+b
    print("after calling",sum)
adding()
```

```
after calling 50
```

In [102]:
```
 def multiplication():
     a=35
     b=20

     multi=a*b

     return multi

 print("after calling multi:",multiplication())
```

```
after calling multi: 700
```

lists

In [7]:
```
# a list  is a collection of charachters variables and
# numbers variables and boolean values data types
# a list is a to store multiple data with in a single variable
#> a list is a ordered type of data
#> a list is a denoted type of data
#> a list item as denoted with double quotes.
```

```
  File "C:\Users\exam3\AppData\Local\Temp/ipykernel_1460/3015370823.py", line 8
    syntax:
    ^
IndentationError: unexpected indent
```

In [115]:
```
## exmaple
li=["mango,banana"]
```

In [116]:
```
# type of list
print(li)
```

```
['mango,banana']
```

In [85]:
```
# lenth of the list
print(len(li))
```

```
12
```

In [117]:
```
# accessing first element of list
print(li[0])
```

```
mango,banana
```

In [118]:
```
# accessing the item in list or not
if"apple" in li:
    print("yes")
else:
    print("no")
```

```
no
```

In [88]:
```
li
```

Out[88]:
```
'mango,banana'
```

In [119]:
```
li[0]="apple"
li
```

Out[119]:
```
['apple']
```

In [120]:
```
li
```

Out[120]:
```
['apple']
```

In [122]:
```
li
```

Out[122]:
```
['apple']
```

In [125]:
```
li.insert(1,"banana,""mango")
li
```

Out[125]:
```
['apple', 'banana,mango', 'banana,mango', 'banana']
```

Out[126]:['apple', 'banana,mango', 'banana,mango', 'banana']

tuple
it is as collection of different types of data.
it is immutable (con't change)
we can using round brackets() to write a tuple.
top create an empty tuple
tuple_name=()
to create single values
tuple_name(values)

In [167]:
```
# to create single values
```

In [168]:
```
# create a tuple
t1=(10,20,30)
t1
print(type(t1))
```

```
<class 'tuple'>
```

In [169]:
```
# single value tuple
t2=(10)
print(type(t2))
t3=(20,)
print(type(t3))
```

```
<class 'int'>
<class 'tuple'>
```

In [170]:
```
t3
```

Out[170]:(20,)

In [171]:
```
t2
```

Out[171]:10

In [172]:
```
# how to acces the values from to  the tuple

t1
print(t1[2])
```

```
30
```

```
(10,)
```

In [174]:
```
t2=(10,20,10,30,20,40,60)
# to count the number of ocurences
t2.count(10)
```

Out[174]: 2

In [176]:
```
#index
t2.index(10)
```

Out[176]: 0

In [177]:
```
t2.index(30)
```

Out[177]: 3

In [178]:
```
t2.index(60)
```

Out[178]: 6

In [179]:
```
t2.index(40)
```

Out[179]: 5

In [183]:
```
tuple=("apple", "banana", "cherry", "apple", "cherry")
print(tuple)
```

```
('apple', 'banana', 'cherry', 'apple', 'cherry')
```

In [2]:
```
# dictionary:
#- it is collection of different data types
#- it is group of key and values(key:value)->item
#- in dictionary keys are unique
#- writen in ({})
#- each and every item seperated with commas(,)
#- accesing dictionary values by using key names
#- it is a multiple(changable)
```

In [ ]:
```
to create a empty dictionary:
    -dictionary_name={}
```

In [ ]:
```
to create the dictionaries value:
    dictionaries_name={key:value,key:value2...}
```

In [3]:
```
# to create a dictionaries with values
d1={'a':10,'b':34,'c':45}
print(d1)
print(type(d1))
```

In [7]:
```
# to create a dictionaraies with different data types..
d2={'a':100,'name':'salman','branch':'cse','b':45.8}
print(d2)
```

```
{'a': 100, 'name': 'salman', 'branch': 'cse', 'b': 45.8}
```

In [8]:
```
# accesing the dictionaries values using the key names
print(d2['name'])
print(d2['b'])
print(d2['a'])
```

```
salman
45.8
100
```

In [9]:
```
#update the dictionaries values
print(d2)
d2['branch']='eee'
print(d2)
```

```
{'a': 100, 'name': 'salman', 'branch': 'cse', 'b': 45.8}
{'a': 100, 'name': 'salman', 'branch': 'eee', 'b': 45.8}
```

In [10]:
```
print(dir(dict))
```

```
['__class__', '__class_getitem__', '__contains__', '__delattr__', '__delitem__', '__dir
__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '_
_gt__', '__hash__', '__init__', '__init_subclass__', '__ior__', '__iter__', '__le__',
'__len__', '__lt__', '__ne__', '__new__', '__or__', '__reduce__', '__reduce_ex__', '__r
epr__', '__reversed__', '__ror__', '__setattr__', '__setitem__', '__sizeof__', '__str_
_', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys', 'pop', 'po
pitem', 'setdefault', 'update', 'values']
```

In [16]:
```
#keys
print(d2)
print(d2.keys())
```

```
{'a': 100, 'name': 'salman', 'branch': 'eee', 'b': 45.8}
dict_keys(['a', 'name', 'branch', 'b'])
```

In [17]:
```
#values
print(d2)
print(d2.values())
```

```
{'a': 100, 'name': 'salman', 'branch': 'eee', 'b': 45.8}
dict_values([100, 'salman', 'eee', 45.8])
```

In [18]:
```
#copy()
print(d2)
d3=d2.copy()
print(d3)
print(type(d3))
```

```
{'a': 100, 'name': 'salman', 'branch': 'eee', 'b': 45.8}
{'a': 100, 'name': 'salman', 'branch': 'eee', 'b': 45.8}
<class 'dict'>
```

In [19]:

```
#get
print(d2)
print(d2.get('a'))
print(d2.get('name'))
```

```
{'a': 100, 'name': 'salman', 'branch': 'eee', 'b': 45.8}
100
salman
```

In [ ]: