In [1]:
```
# define function
def function(a,b):
    c=a+b
    print(c)
```

In [2]:
```
function(4,5)
```

```
9
```

In [ ]:
```
# to call a function, use the function name followed by parenthesis
```

In [10]:
```
def my_function(fname, lname):
  print(fname + " " + lname)

my_function("riya",'sharma')
```

```
riya sharma
```

```
Arbitrary Arguments, *args
If we not know how many arguments that will be passed into your function, add
a * before the parameter name in the function definition.

This way the function will receive a tuple of arguments, and can access the
items accordingly:
```

In [11]:
```
employees=["rutuja","rashmi","ashutosh",'rahul']
```

In [17]:
```
def empl(args):
    print("Second Employee is", args[1])
```

In [18]:
```
empl(employees)
```

```
Second Employee is rashmi
```

In [19]:
```
def employee(*args):
    for x in args:
        print(x)
```

In [20]:
```
employee(23,"Pink","purple",45)
```

```
23
Pink
purple
45
```

In [21]:
```
def employee(*args):
    print(args)
```

In [22]:
```
employee(23,"Pink","purple",45)
```

```
(23, 'Pink', 'purple', 45)
```

```
Arbitrary Keyword Arguments, **kwargs
If you do not know how many keyword arguments that will be passed into your
function, add two asterisk: ** before the parameter name in the function
definition.

This way the function will receive a dictionary of arguments, and can access
the items accordingly:
```

In [25]:
```
def employee(**args):
    print(args)
```

In [30]:
```
employee(first="Pink",second="purple")
```

```
{'first': 'Pink', 'second': 'purple'}
```

In [33]:
```
def details(**args):
    print(args)
```

In [34]:
```
details(name="priyanks",ahe=23,city="pune")
```

```
{'name': 'priyanks', 'ahe': 23, 'city': 'pune'}
```

In [47]:
```
# Function to get the max element from a list.
def max_ele(list1):
    maximum=list1[0]
    for x in list1:
        if x > maximum:
            maximum=x
    print(x," is maximum element")
```

In [48]:
```
max_ele([34,43,32,56])
```

```
56  is maximum element
```

In [56]:
```
def max_e(l):
    l.sort()
    print(l[-1],"is maximum")
```

In [57]:
```
max_e([23,45,88,76,88])
```

```
88 is maximum
```

In [59]:
```
max_e([23,45,88,76,88,56,45,100])
```

```
100 is maximum
```

In [68]:
```
# Function to check whether a string is palindrome or not
def palindrome(s):
    s1=str(s)
    if s1==s1[::-1]:
        print(f"{s1} is palindrome")
    else:
        print(f"{s1} is not palindrome")
```

In [69]:
```
palindrome(123321)
```

```
123321 is palindrome
```

In [70]:
```
palindrome('racecar')
```

```
racecar is palindrome
```

In [71]:
```
palindrome("Racecar")
```

```
Racecar is not palindrome
```

In [72]:
```
def palindrome(s):
    s1=str(s).lower()
    if s1==s1[::-1]:
        print(f"{s1} is palindrome")
    else:
        print(f"{s1} is not palindrome")
```

In [73]:
```
palindrome("Racecar")
```

```
racecar is palindrome
```

In [94]:
```
# for check number is plaindrome or not
```

In [95]:
```
pal('mosquito')
```

```
mosquito is not palindrome
```

In [96]:
```
pal("madam")
```

```
madam is not palindrome
```

In [90]:
```
pal("MalayalAm")
```

```
malayalam is not palindrome
```

```
  Find the frequency of each distinct element in the list using a Python
dictionary.
Input:  [1, 1, 1, 5, 5, 3, 1, 3, 3, 1, 4, 4, 4, 2, 2, 2, 2]
Output:  {1:5, 2:4, 3:3, 4:3, 5:2}
```

In [111]:
```
def freq(I):
    dic={}
    freq_c=[]
    count=0
    for x in I:
        if x not in dic:
            count+=1
            dic[x]=I.count(x)
    sorted(dic.keys())
    print(dic)
```

In [112]:
```
Input=[1, 1, 1, 5, 5, 3, 1, 3, 3, 1, 4, 4, 4, 2, 2, 2, 2]
freq(Input)
```

```
{1: 5, 5: 2, 3: 3, 4: 3, 2: 4}
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: