# Advance problems

In [4]:

```
#Function to calculate average of all factorials
def factorial(n):
    fact=1
    for i in range(1,n+1):
        fact=fact*i
    #print(fact)
       return True
#factorial(5)
```

```
120
```

In [9]:

```
#Function to calculate average of all factorials in a given range
def factorailrange(lb,ub):

    for j in range(lb,ub+1):
        if factorial(j):
            print(fact)
factorailrange(1,5)
```

```
1
2
6
24
120
```

In [10]:

```
#Function to generate  armstrong numbers
def armstrong(n):
    rem=0
    while(n>0):
        rem=n%10
        for
        sum=sum+rem*rem*rem
```

In [11]:

```
#Function to generate N odd armstrong numbers
```

# Day objective:

- python Data Strucutes
- Python Data Structures
- Lists
- Tuples
- Dictionaries
- Basic Problem set on Data Structures
- Advanced Problem Set
- Packages and Modules in Python

In [ ]:

```
```

# Python Data Structures

- Lists

In [54]:

```
l=[123,978,654]

l# Access the entire list

l[1] #to access particular element with the index in a list

l[1:]  # All elements from Second to last element

l[::-1] # to print list in reverse order

l=l[::-1]# reversing list elements and updating in list

l=l[::-1]# rearranging the elements and updating in list

l[::2]#accessing even index element

l[1::2]#accessing odd index element

#Lists ca be Accessed,Manipulated in two different Ways
            #Direct Referencing-[Index]
            #Indirect Referencing-through functions

l.append(345)#adding an element in end of the list


l.insert(1,234) #adding an element at a particular index value

l.sort() #Accessing the data in ascending order

l.pop()#remove the last element in the list

l.pop(1)#remove an element at a particular element

l2=[12,23,45]

#merge list2 into list 1

l.extend(l2)

l

sum(l)

max(l)

len(l)

l.remove(12)

l
```

Out[54]:

```
[123, 345, 654, 23, 45]
```

In [56]:

```
#Average of a list
l=[12,34,56]
avg=sum(l)/len(l)
avg
```

Out[56]:

```
34.0
```

In [64]:

```
#Average of all alternative elements
l=[12,23,45,56]
avg=sum(l[::2])/len(l)
avg

#Avg of odd elements
avg=sum(l[1::2])/len(l)
avg
```

Out[64]:

```
19.75
```

In [ ]:

In [129]:

```
#Function to identify the second largest element in a list

def secondlargest(p):
    p.sort()
    #print(t)
    print(p[-2])
secondlargest([1,6,4,5,6])
```

```
6
```

In [ ]:

```
l=[12,45,23]
l.sort()
t=l
print(t)
print(t[-2])
```

In [ ]:

In [143]:

```
#Function to identify the second largest element in a list

def secondlargest(p):
    p.sort()
    p.remove(max(p))
    print(p[-3])
secondlargest([1,6,4,5,61,611,611])
```

```
6
```

In [ ]:

In [146]:

```
def genericlargest(l,n):
    l.sort()
    return l[-n]


genericlargest([12,45,34,56,23],3)
```

Out[146]:

```
34
```

In [159]:

```
#Function to search for data in a list
#search for a key in the list and return index value .If it is not found retur
n -1
def linearsearch(l,key):

    for i in l:
        if key==i:
            print(key)
        else:
            print()

linearsearch([12,34,56],34)
```

```
34
```

In [165]:

```
#Function to search for data in a list
#search for a key in the list and return index value .If it is not found retur
n -1
def linearsearch(l,key):
    for index in range(0,len(l)):
        if l[index]==key:
            return index
    return -1
linearsearch([12,34,56],34)
```

Out[165]:

```
1
```

In [166]:

```
#Function to search for data in a list
#search for a key in the list and return index value .If it is not found retur
n -1
def linearsearch(l,key):

    for i in l:
        if key==i:
            return l.index(i)
    return -1
linearsearch([12,34,56],34)
```

Out[166]:

```
1
```

In [173]:

```
l=[12,34,56]
try:
    print(l.index(1))
except:
    print(-1)
```

```
-1
```

In [174]:

```
def linearsearch(l,key):
    if key in l:
        return l.index(key)
    return -1
linearsearch(l,12)
```

Out[174]:

```
0
```

In [187]:

```
#Function to count the occurances of character in a list
#"python Programming",m-->2
def occurances(string,key):
    count=0
    for value in string:
        if key ==value:
            count=count+1
    print(count)
occurances("python programming","m")

#To find out multiple strings

def occurances2(string,key):
    return string.count(key)
occurances2("python programming","p")
```

```
2
```

Out[187]:

```
1
```

In [ ]:

In [190]:

```
#Function to find out substring
#ababacc--->2
def substring(string,key):
    count=0
    for value in string:
        if value==key:
            count=count+1
        print(count)
substring("aabaa","aa")
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-190-09fd91402523> in <module>
      7             count=count+1
      8         print(count)
----> 9 substring("aabaa",aa)

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

In [ ]: