1.Factorial

In [1]:
```
i=1
def gcd(x,y):
    factor_x=[i for i in range(1,x+1) if x%i==0]
    factor_y=[i for i in range(1,y+1) if y%i==0]
    commom_factor=max(set(factor_x)&set(factor_y))
    return commom_factor
```

In [2]:
```
gcd(10,25)
```

Out[2]: 5

In [3]:
```
gcd=lambda x,y:max([i for i in range(1,min(x,y)+1) if x%i==0 and y%i==0])
```

In [4]:
```
gcd(30,5)
```

Out[4]: 5

In [5]:
```
gcd=lambda x,y:max([i for i in range(1,min(x,y)+1) if x%i==0 and y%i==0])
```

In [6]:
```
gcd(10,40)
```

Out[6]: 10

2.Prime number

In [7]:
```
gcd=lambda x :([i for i in range(2,x) if x%i==0])
```

In [8]:
```
gcd(30)
```

Out[8]: [2, 3, 5, 6, 10, 15]

In [9]:
```
prime=lambda x:([ i for i in range(1,x) if  x%i==0])
```

In [10]:
```
prime(20)
```

Out[10]: [1, 2, 4, 5, 10]

In [11]:
```
prime_n=lambda x: "prime" if len([i for i in range(2,x) if x%i==0])==0 else "not prime"
```

In [12]:
```
prime_n(24)
```

Out[12]: 'not prime'

In [13]:
```
prime_number= lambda x:[i for i in range(2,x) if x%i==0]
```

In [14]:
```
prime_number(17)
```

Out[14]: []

3.Recursive

In [15]:
```
def fun(x):
    if x==1:
        return 1
    else:
        return fun(x-1)+2
```

In [16]:
```
fun(3)
```

Out[16]:
```
5
```

4.Factorial

In [17]:
```
def fac(x):
    if x==1:
        return 1
    else:
        return x*fac(x-1)
```

In [18]:
```
fac(6)
```

Out[18]:
```
720
```

5.The Fibonacci

In [19]:
```
#1,1,2,3,5,8,13,21
```

In [20]:
```
def fun(x):
    if (x==1) or (x==2):
        return 1
    else:
        return fun(x-1)+fun(x-2)
```

In [21]:
```
fun(7)
```

Out[21]:
```
13
```

6.Square root

In [22]:
```
l=[1,4,7,34,12,53]
```

In [23]:
```
l1=(i**2 for i in l)
l1
```

Out[23]:
```
<generator object <genexpr> at 0x000001D1BDE8AD60>
```

In [24]:
```
next(l1)
```

Out[24]:
```
1
```