In [25]:
```
def fun(n):
    if n == 0: return 0
    elif n == 1: return 1
    else: return fun(n-1)+fun(n-2)

if __name__ == '__main__':
    b = int(input('enter:'));
    s=fun(b)
    print(s)
```

```
enter:1
1
```

In [2]:
```
def fun(var):
    result = [[]];
    for i in var:
        new=[]
        for j in result:

            new.extend([j + [i]]);

        result.extend(new);
    return result


    #return result;
#def fun2(orig,newset):

b=[1,2,3];
s=fun(b);
print(s)
```

```
[[], [1], [2], [1, 2], [3], [1, 3], [2, 3], [1, 2, 3]]
```

In [38]:
```
print(5/2)
```

```
2.5
```

In [61]:
```
def subs(l):
    if l == []:
        return [[]]
    x = subs(l[1:])
    print(x)
    return x + [[l[0]] + y for y in x]

l=[1,2,3]
print(subs(l))
```

```
[[]]
[[], [3]]
[[], [3], [2], [2, 3]]
[[], [3], [2], [2, 3], [1], [1, 3], [1, 2], [1, 2, 3]]
```

In [65]:
```
def k_subsets(k, set_):
    if k == 0:
        return [[]]
    else:
        subsets = []
        for ind in range(len(set_) - k + 1):
            for subset in k_subsets(k - 1, set_[ind + 1:]):
                subsets.append(subset + [set_[ind]])
        return subsets
l=[1,2,3]
print(k_subsets(1,l))
```

```
[[1], [2], [3]]
```

In [18]:
```
def power_set_2(set_,N):
    subsets = [[]]
    subsetsK=[];
    for element in set_:
        for ind in range(len(subsets)):
            subsets.append(subsets[ind] + [element])
            if len(subsets[ind] + [element])==N:
                subsetsK.append(subsets[ind] + [element])

    return subsetsK
l=[1,2,3,4,5]
s=power_set_2(l,3)
print(s)
```

```
[[1, 2, 3], [1, 2, 4], [1, 3, 4], [2, 3, 4], [1, 2, 5], [1, 3, 5], [2,
 3, 5], [1, 4, 5], [2, 4, 5], [3, 4, 5]]
```

In [80]:
```
def fun (i,j):
    if i==1 or j==1:
        yield 1;
    yield fun(i-1,j)+fun(i,j-1);

s=fun(6,6)
for i in s:
print(i)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-80-d3636f6c36b8> in <module>()
      4     yield fun(i-1,j)+fun(i,j-1);
      5 s=fun(6,6)
----> 6 for i in s:
      7     print(i)

<ipython-input-80-d3636f6c36b8> in fun(i, j)
      2     if i==1 or j==1:
      3         yield 1;
----> 4     yield fun(i-1,j)+fun(i,j-1);
      5 s=fun(6,6)
      6 for i in s:

TypeError: unsupported operand type(s) for +: 'generator' and 'generator'
```

In [134]:
```
def perm(var):
    if (len(var)==1):
        return [var];
    p=[]
    for i in range(0,len(var)):
        if i==len(var)-1:
            temp=var[:i];
        else:
            temp=var[:i]+var[i+1:]
        x=perm(temp)
        temp2=var[i]

        for j in x:
            p.append(temp2+j);

    return p;
l='abc';
b=print(perm((l)))
```

```
['abc', 'acb', 'bac', 'bca', 'cab', 'cba']
```

In [100]:
```

```

```
a
```

In [14]:
```
def compute_all_parens (n, left, right, s):
    if right == n:
        print (s)
        return
    if left < n:
        compute_all_parens(n, left+1, right, s + "(")
    if right < left:
        compute_all_parens(n, left, right+1, s + ")")
```

In [16]:
```
compute_all_parens(3,0,0,"")
```

```
((()))
(()())
(())()
()(())
()()()
```

In [17]:
```
def fun(money,coins,index):

    if (money==0):
        return 1;
    if (index>=len(coins)):
        return 0;
    amountwithCoin = 0;
    counter=0;

    while (amountwithCoin <= money):
        r = money - amountwithCoin;

        counter = counter + fun(r,coins,index+1);
        amountwithCoin += coins[index];
    return counter;

s=fun(5,[1,2,3],0)
print(s)
```

```
5
```

In [18]:
```
def fun(a,col,n):
    if (col>=n):
        return True;
    for i in range(n):
        if isSafe(a,i,col,n):
            a[i][col]=1;
            if (fun(a,col+1,n)):
                return True;
            a[i][col]=0;
    return False;
    for i in range(n):
        for j in range(n):
            print(a[i][j])

def isSafe(a, row, col,N):

    for i in range(col):
        if a[row][i] == 1:
            return False

    for i,j in zip(range(row,-1,-1), range(col,-1,-1)):
        if a[i][j] == 1:
            return False
    for i,j in zip(range(row,N,1), range(col,-1,-1)):
        if a[i][j] == 1:
            return False
    return True;

import numpy as np
a = np.zeros((4,4))
print(fun(a,0,4))
```

```
True
```

In [29]:
```

```

```
[[False False False False]
 [False False False False]
 [False False False False]
 [False False False False]]
```

In [30]:
```
row=4;
col=4
for i,j in zip(range(row,-1,-1), range(col,-1,-1)):
        print(i)
        print(j)
```

```
4
4
3
3
2
2
1
1
0
0
```

In [71]:
```
def paint(a,x,y,newColor,oldColor):
    if (a[x][y]!=oldColor):
        return ;

    a[x][y]=newColor;


    if (x>0):
        paint(a,x-1,y,newColor,oldColor);
    if (x>0 and y>0):
        paint(a,x-1,y-1,newColor,oldColor);
    if (x>0 and y<len(a[0])-1):
        paint(a,x-1,y+1,newColor,oldColor);

    if x<len(a)-1:
        paint(a,x+1,y,newColor,oldColor);
    if (x<len(a)-1 and y>0):
        print(x+1)
        print(y-1)
        paint(a,x+1,y-1,newColor,oldColor);
    if x<len(a)-1 and y<len(a[0])-1:
        paint(a,x+1,y+1,newColor,oldColor);

    if y<len(a[0])-1:
        paint(a,x,y+1,newColor,oldColor);
    if (y>0):
        paint(a,x,y-1,newColor,oldColor);


a=[[1,2,3],[2,3,4],[2,3,6]]
print(a)
paint(a,0,1,-1,2)
print(a)
```

```
[[1, 2, 3], [2, 3, 4], [2, 3, 6]]
1
0
[[1, -1, 3], [-1, 3, 4], [-1, 3, 6]]
```

In [47]:
```
a=[[1,2,3],[2,3,4]]
print(a[1][2])
```

```
4
```

In [76]:
```
def createGenerator():
    mylist = range(3)
    for i in mylist:
        yield i*i
mygenerator = createGenerator() # create a generator
for i in mygenerator:
    print(i)
print(mygenerator)
```

```
0
1
4
<generator object createGenerator at 0x7fead347bba0>
```

In [14]:
```
def sub(arr,t):

    l=[];
    if len(arr)==1:
        return arr;
    left = arr[:len(arr)//2];
    right = arr[len(arr)//2:];

    l1=sub(left,0);
    l2=sub(right,0);
    for i in l1:
        l.append(i);
    for i in l2:
        l.append(i);
    for i in l1:
        for j in l2:
            l.append(i+j);

    if t==1:
        l.append(' ')
    return l;
```

In [16]:
```
a=['1','2','3','4'];
print(sub(a,1))
```

```
['1', '2', '12', '3', '4', '34', '13', '14', '134', '23', '24', '234',
 '123', '124', '1234', ' ']
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: