```
# -*- coding: utf-8 -*-
"""
Created on Sun Oct  9 12:13:13 2016

@author:
ericgrimson
"""

def bisect_search2(L, e):
    def bisect_search_helper(L, e,
low, high):
        print('low: ' + str(low) + '; high: ' + str(high))  #added to visualize

    if high == low:
            return L[low] == e
        mid = (low + high)//2
        if
L[mid] == e:
            return True
        elif L[mid] > e:
            if low == mid:
#nothing left to search
                return False
            else:
                return
bisect_search_helper(L, e, low, mid - 1)
        else:
            return
bisect_search_helper(L, e, mid + 1, high)
    if len(L) == 0:
        return False
    else:

     return bisect_search_helper(L, e, 0, len(L) - 1)

testList = []
for i in range(100):

testList.append(i)

print(bisect_search2(testList, 76))


def genSubsets(L):
    res = []

  if len(L) == 0:
        return [[]] #list of empty list
    smaller = genSubsets(L[:-1]) #
all subsets without last element
    extra = L[-1:] # create a list of just last element

new = []
    for small in smaller:
        new.append(small+extra)  # for all smaller
solutions, add one with last element
    return smaller+new  # combine those with last element
and those without


testSet = [1,2,3,4]
print(genSubsets(testSet))
```