作业

题【1】

最坏情况且无溢出的条件下，循环语句执行了n-1次，此时时间复杂度为O(n)；最好情况下，最后一位为0，则A[i]=1执行了一次。由于各种情况出现的概率均等，因此平均情况下的时间复杂度是O(n)=n/2

题【2】

代码如下：

b=[]

def combNumber(m, n):

global b

for i in range(m, n-1, -1):

b.append(i)

if n-1>0:

combNumber(i-1,n-1)

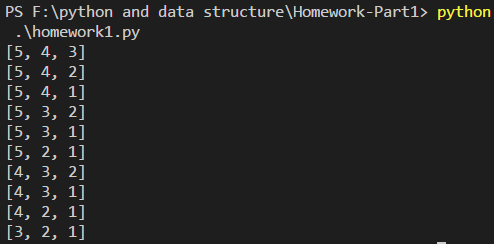
else:

print b

b.pop()

combNumber(5,3)

结果：



题【3】

代码如下：

def dset(li):

arr=[]

b=[]

def pick(li,m):

for i in range(len(li)-1,m-2,-1):

b.append(li[i])

ls=li[0:i]

if m-1>0:

pick(ls,m-1)

else:

#print sorted(b)

arr.append([num for j,num in enumerate(sorted(b))])

b.pop()

for id in range(3,0,-1):

pick(li,id)

return sorted(arr)

group=[1,2,3]

print dset(group)

结果：



题【4】

def MaXD(A):

max=A[0]

min=A[0]

for i in range(len(A)):

if A[i]>max:

max=A[i]

if A[i]<min:

min=A[i]

return max-min

结果：



题【5】

def mindistance(A, x, y):

min = len(A)

point = 0

for i in range(len(A)):

if A[i] == x or A[i] == y:

if A[point]!=x and A[point]!=y:

point=i

else:

if A[point] != A[i]:

if min > abs(point-i):

min = abs(point-i)

point=i

return min

A = [1,2,6,5,6,7,4,3,2,3,1,5,9,8,6,4,2,5]

print mindistance(A,5,1)

结果：



题【6】

def MaxSub\_sum(A):

sum=0

maxSum=A[0]

for i in range(len(A)):

sum += A[i]

if sum > maxSum:

maxSum = sum

elif sum <0:

sum = 0

return maxSum

A=[-2, 11, -4, 13, -5, -2]

B=[-6, 2, 4, -7, 5, 3, 2, -1, 6, -9, 10, -2]

print MaxSub\_sum(A)

print MaxSub\_sum(B)

结果

