def maxprofit(price,n):

profit = [0] \* n

max\_price = price[n-1]

for i in range(n - 2,0,-1):

if price[i] > max\_price:

max\_price = price[i]

profit[i] = max(profit[i +1], max\_price - price[i])

min\_price = price[0]

for i in range(1, n):

if price[i] < min\_price:

min\_price = price[i]

profit[i] = max(profit[i-1], profit[i] +(price[i] - min\_price

result = profit[n - 1]

return result

price = [2, 30, 15, 10, 8, 25, 80]

print("maximum profit is",maxprofit(price, len(price)))

first\_num = int(input("enter the the first number....."))

second\_num = int(input("enter the second number....."))

third\_num = int(input("enter the third number....."))

my\_list = []

print("the first number is")

print(first\_num)

print("the second number is ")

print(second\_num)

print(" the third number is ")

print(third\_num)

my\_list.append(first\_num)

my\_list.append(second\_num)

my\_list.append(third\_num)

for i in range(0,3):

for j in range(0,3):

for k in range(0,3):

if(i!=j&j!=k&k!=i):

print(my\_list[i],my\_list[j],my\_list[k])

def solve(nums):

count = 0

n = lens(nums)

for i in range(n):

for j in range(i+1,n):

if nums[i] == nums[j]:

count+=1

return count

nums = [5, 6, 7, 5, 5, 7]

print(solve(nums))

def add\_binary\_nums(x,y):

max\_len = max(len(x), len(y)))

x = x.zfill(max\_len)

y = y.zfill(max\_len)

result =' '

carry = 0

for i in range(max\_len - 1, -1, -1):

r = carry

r += 1 if x[i] == '1' else 0

r += 1 if y[i] == '1' else 0

result = ('1' if r % 2 == 1 else '0') + result

carry = o if r < 2 else 1

if carry ! = 0: result = '1' + result

return result.zfill(max\_len)

print(add\_binary\_num('1101', '100'))

def minjimps(arr, l, h):

if (h == l):

return 0

if (arr[l] == 0):

return float('iinf')

min = float('inf')

for i in range(l + 1, h+1):

if (i < l + arr[l] + 1):

jumps = minjumps(arr, i, h)

if (jumps != float('inf') and jumps + 1< min):

min = jumps + 1

return min

arr = [1, 3, 6, 3, 2, 3, 6, 8, 9, 5]

n = len(arr)

print('minimum number of jumps to reach','end is', min jumps(arr, o, n - 1))

def reverse(s):

str = " "

for i in s:

str = i + str

return str

s = "12345"

print("the original string is : ", end=" ")

print(s)

print("the reversed string(using loops) is : ", end=" ')

print(reverse(s))

from itertools import permutations

a = permutation ([1,2,3],2)

for i in a:

print(i)

s1 = input("enter a string")

s2 = input("enter a string")

if(sorted(s1)==sorted(s2)):

print("anagram")

else:

print("not anagram")

def editdistance(str1, str2):

n = len(str1)

m = len(str2)

ans = editdistancehelper(n, m, str1, str2)

return ans

print(editdistance("insertion", "execution"))