import random as rd

lst=[]

lst2=[]

numberlst=[]

variationArray=[]

weightl=[]

#READING INPUT FROM FILE

with open('F:\\CSE422\\New folder\\genetic.txt')  as file:

    lst=file.read().split()[1:]

#LIST HANDELING

def listAdjust():

    for item in lst:

        if item=='l' :

            lst2.append('l')

            lst.remove(item)

        elif item=='d':

            lst2.append('d')

            lst.remove(item)

    iterate=len(lst2)

    return iterate

#GENERATE INITIAL POPULATION

def generate\_popuation(iterate):

    population\_size=2\*\*len(lst2)-1

    check=1

    while check<=population\_size:

        variation=""

        for j in range (iterate):

            variant=str(rd.randrange(0,2))

            variation+=variant

        if variation not in variationArray:

            variationArray.append(variation)

            check=check+1

    variationArray.sort()

    variationArray.pop(0)

    # print("This is Population Generation")

   # print(variationArray)

    return population\_size

#FITNESS CALCULATION

def FitnessGeneration():

    weight=[]

    value='0'

    for i in variationArray:

        weightv=0

        iterator=0

        for j in i:

            if(j=='1') and (lst2[iterator]=='l'):

                weightv=weightv-int(lst[iterator])

            elif (j=='1') and (lst2[iterator]=='d'):

                weightv=weightv+int(lst[iterator])

            iterator+=1

        weight.append(weightv)

        if(weightv==0):

            value= i

    weightl=weight

    #print(weightl)

    return value

#PARENTSELECTION

def parentSelection():

    parent=variationArray

#CROSSOVER

def crossover(iterate,population\_size):

    j=0

    i=0

    #print(f'{"Variation len"}{len(variationArray)}')

    if(len(variationArray)%2==0):

        #print("WOK")

        while i <=(population\_size-1):

            j=i+1

            if(j<=population\_size-1):

            #   print(f'{variationArray[i]}{" "}{variationArray[j]}' ,end=" ")

                value=rd.randrange(0, iterate)

                temp=variationArray[i]

                p1=variationArray[i]

                p2=variationArray[j]

                p1=p1[:value]+p2[value:]

                variationArray[i]=p1

                p2=p2[:value]+temp[value:]

                variationArray[j]=p2

            i=j+1

    elif(len(variationArray)%2!=0):

        #print("WWOK")

        while i <=(population\_size-1):

            j=i+1

            if(i==(population\_size-1)):

                #print("WWOK")

                j=i-1

                value=rd.randrange(0, iterate)

                print(f'{"Randrange "}{value}')

                temp=variationArray[i]

                p1=variationArray[i]

                p2=variationArray[j]

                p1=p1[:value]+p2[value:]

                variationArray[i]=p1

                p2=p2[:value]+temp[value:]

                variationArray[j]=p2

                break

            elif(j<=population\_size-1):

            #   print(f'{variationArray[i]}{" "}{variationArray[j]}' ,end=" ")

                value=rd.randrange(0, iterate)

                temp=variationArray[i]

                p1=variationArray[i]

                p2=variationArray[j]

                p1=p1[:value]+p2[value:]

                variationArray[i]=p1

                p2=p2[:value]+temp[value:]

                variationArray[j]=p2

            i=j+1

    #print(variationArray)

    #print(f'{"Variation len"}{len(variationArray)}')

    variationArray.sort()

    if(int(variationArray[0])==0):

        variationArray.pop(0)

    #print("This is Crossover")

#Mutation

def Mutation(iterate):

    bit\_flip=rd.randrange(0, iterate)

   # print(f'{"bit\_flip "}{bit\_flip}')

    i=0

    while(i<len(variationArray)):

        l=variationArray[i]

        if(l[bit\_flip]=='1'):

            l=l[:bit\_flip]+'0'+l[bit\_flip+1:]

            variationArray[i]=l

        elif(l[bit\_flip]=='0'):

            l=l[:bit\_flip]+'1'+l[bit\_flip+1:]

            variationArray[i]=l

        i=i+1

    variationArray.sort()

    if(int(variationArray[0])==0):

        variationArray.pop(0)

    #print("After Mutation")

    #print(variationArray)

# x=listAdjust()

# population\_size=generate\_popuation(x)

# l=FitnessGeneration()

# print(f'{"Hello "}{l}')

# crossover(x,population\_size)

# l=FitnessGeneration()

# print(f'{"Hello "}{l}')

# Mutation(x)

# l=FitnessGeneration()

# print(f'{"Hello "}{l}')

#This is Method

def geneticAlgoMain():

    parentSelection()

    x=listAdjust()

    population\_size=generate\_popuation(x)

    l=FitnessGeneration()

    cross=0

    mute=0

    while(True):

        if(l=='0' and cross==0):

            crossover(x,population\_size)

            l=FitnessGeneration()

            cross=cross+1

            if(len(l)>0 and int(l)!=0):

                #print("Cross")

                print(f'{"Output "}{l}')

                break

        elif(l=='0' and mute==0):

            Mutation(x)

            l=FitnessGeneration()

            mute=mute+1

            if(len(l)>0 and int(l)!=0):

                #print("Mute")

                print(f'{"Output "}{l}')

                break

        elif(len(l)>0 and int(l)!=0):

            print(f'{"Output "}{l}')

            break

        elif(mute>0 and cross>0):

            print(f'{"Output "}{-1}')

            break

geneticAlgoMain()