import random as r

import math

student\_id=input("1.  Enter your Student ID ")

lst=[int(lst)for lst in input("2.  Minimum And Maximum Value for the range of negative HP ").split()]

print(" ")

print(" ")

maxv=lst[0]

minv=lst[1]

depth=int(student\_id[0])\*2

turns=depth

branch=int(student\_id[2:3])

life=student\_id[-2:][::-1]

total\_leaf=int(branch)\*\*int(depth)

leaf\_list=[]

visited\_list=[]

for i in range(total\_leaf):

    leaf\_list.append(r.randrange(maxv,minv))

#leaf\_list=[]

#leaf\_list=[19,22,9,2,26,16,16,27,16]

#leaf\_list=[18,13,5,12,10,5,13,7,17,8,6,8,5,11,13,18]

def AlphabetaPrunning(Alpha,beta,branch,depth,maxplayer,pos):

    if(depth==0):

        cal=pos

        visited\_list.append(cal)

        return leaf\_list[cal]

    if(maxplayer==True):

        maxEvaluation=-(math.inf)

        for maxc in range(branch):

            maxplayer=False

            evaluation=AlphabetaPrunning(Alpha,beta,branch,depth-1,maxplayer,pos\*branch+maxc)

            maxEvaluation=max(maxEvaluation,evaluation)

            Alpha=max(Alpha,maxEvaluation)

            if(Alpha>=beta):

                break

        return maxEvaluation

    else:

        minEvaluation=math.inf

        for minc in range (branch):

            maxplayer=True

            evaluation=AlphabetaPrunning(Alpha,beta,branch,depth-1,maxplayer,pos\*branch+minc)

            minEvaluation=min(minEvaluation,evaluation)

            beta=min(beta,minEvaluation)

            if(Alpha>=beta):

                break

        return minEvaluation

value=AlphabetaPrunning(-math.inf, math.inf, branch, depth, True, 0)

print(f'{"1.  Depth and branches ratio is "}{depth}{":"}{branch}')

print(f'{"2.  Terminal States (leaf node values) are "}{leaf\_list}')

print(

    f'{"3.  Left life (HP) of the defender After maximum damage caused by the Attacker is "}{int(life)-int(value)}')

print(f'{"4.  After Alpha-beta Pruning Leaf Node Comparisons "}{len(visited\_list)}')