#max-min Algorithm

import math

i\_M=0

def minimax (position,depth,maximizingplayer,le,myval):

global i\_M

if depth == 0:

i\_M+=1

return le[myval]

if maximizingplayer:

maxival= -math.inf

for x in range((position\*3)+1,(position\*3)+4):

if x<len(le):

eval=minimax (x,depth-1,False,le,x)

maxival=max(maxival,eval)

myval=maxival

#print(" MAX ",maxival)

return myval,i\_M

else :

minival= +math.inf

for x in range((position\*3)+1,(position\*3)+4):

if x<len(le):

eval=minimax (x,depth-1,True,le,x)

minival=min(minival,eval)

myval=minival

#print(" MIN ",minival)

return myval

# Alpha Beta Pruning Algorithm

import math

i\_A=0

def max\_value (position,depth,le,a,B):

global i\_A

if depth == 0:

i\_A+=1

return le[position]

v=-math.inf

for x in range((position\*3)+1,(position\*3)+4):

eval=min\_value (x,depth-1,le,a,B)

v=max(v,eval)

if v >= B:

return v

a=max(a,v)

#print(" MAX ",a)

return a,i\_A

def min\_value (position,depth,le,a,B):

global i\_A

if depth == 0:

i\_A+=1

return le[position]

v= math.inf

for x in range((position\*3)+1,(position\*3)+4):

eval=max\_value (x,depth-1,le,a,B)

v=min(v,eval)

if v <= a:

return v

B=min(B,v)

#print(" MIN ",B)

return v

def Alpha\_Beta\_Search(State):

v,i\_A= max\_value(0,2,State,-math.inf,math.inf)

return v,i\_A

import numpy as np

import random

file=open('/content/test03.txt')

line\_1=file.readline().strip()

turn=int(line\_1)

line\_2=file.readline().strip()

branch=int(line\_2)

line\_3=file.readline().strip()

sub\_line\_3=line\_3[0]

min\_node=int(sub\_line\_3)

sub\_line\_3=line\_3[1:]

max\_node=int(sub\_line\_3)

depth=2\*turn

print("Depth:",depth)

print("Branch:",branch)

Terminal\_States=pow(3,depth)

print("Terminal States (Leaf Nodes):",Terminal\_States)

length=0

for i in range(branch):

length=length+pow(3,i)

pos=[]

for i in range(length):

pos.append(random.randint(min\_node,max\_node))

#print(pos)

M\_number,i\_M= minimax (0,2,True,pos,4200)

A\_number,i\_A= Alpha\_Beta\_Search(pos)

if M\_number == A\_number:

print("Maximum amount:",A\_number)

print("Comparisons:",i\_M)

print("Comparisons:",i\_A)