***Name and Title of the Report***

**Name:** Introduction to Artificial Intelligence; laboratory exercise I

**Title:** Ayo Game

***Authors***

AKINDE EMMANUEL CSC/2012/021 ENGINEERING

OLADELE IDOWU CSC/2012/044 ENGINEERING

OLUWADAMILER YETUNDE CSC/2012/077 ECONOMICS

BOLUWATIFE FASUGBA CSC/2013/123 ENGINEERING

OLUWAYEMI ONI CSC/2013/137 ECONOMICS

OLUWASHINA KURE-OJO CSC/2012/053 ENGINEERING

***Contents***

**Page**

Name and Title of the report 1

Author 1

Contents 2

Glossary of Notations and Terms 2

Introduction 3

Problem Statement 4

Discussion of Result 4

Interpretation and Application of results 19

Summary and Conclusion 20

***Glossary of Notations and Terms***

**Scoop:** To remove all seeds from a hole, noting that an empty hole cannot be scooped. Scoop occurs when playing or capturing.

**Hole (n):** This is the holes at each side of the board. “n” ranges from 1 to 6

**Capture:** This is the process by which a player wins a number of seed from the opponent. This is done clock wisely i.e. traverse backward.

**Traverse:** Move to the next hole anti-clock wisely, noting that the hole scooped cannot be visited. This is traversing forward.

**Board:** This is the flat surface, usually made of wood, where Ayo game is played. Usually called ***Opon****.*

**Game tree:** A diagram that represents all the possible move in the game.

**Game state:** The screenshot of a particular stage of the game.

***Introduction***

Ayo is a popular indigenous board game in West Africa, particularly among the Yoruba people. Ayo is a *Two person*, *zero sum*, *complete information* game. These terms are explained as follows:

**Two-person:** This means that there are two opposing individuals (or teams) involved in playing the game. The two individuals or teams are contending on the same resources, in this case, Ayo seeds. Their interactions is defined and guided by the rules of the game.

**Zero sum:** This implies that the *total reward* for the players when the game is completed is zero i.e. the sum whether if there is a winner and a loser or a draw is equal to zero.

**Complete information:** This means that all the information about the game is available to both players (as well as the spectators). All information about the state and progress of the game are as represented on the game board.

The Ayo game is played on a wooden board called *Opon* which contains two rows of six holes. At the beginning of the game, each hole contains 4 seeds, making a total of 48 seeds in the game. There is usually a protruding hole, about twice the size of the play hole, on each ends of the board. This is where the players keep captured seeds.

To move, a player chooses a non-empty hole from his side of the board and plays in an anti- clockwise direction. He can capture the seed in his opponent’s side if his movement ends in his opponent side, and there are two or three seeds at the last hole i.e. where the player drops his/her last seed. The capturing is done clock wisely, and a player cannot capture seeds at his own side.

The aim of the game is to win, but the game can also end as a draw. The game ends when one of the following condition occurs:

• There are so few seeds remaining on the board, so that it is not possible to capture any more (infinite configuration).

• One of the players has no hole containing enough seeds to reach his opponent’s side and his opponent’s holes are all empty (deadlock configuration).

• The players decide by mutual agreement to stop playing and share the remaining seeds according to their analysis of the situation. This is the most common way of ending games between good players when the seeds on the board are too few.

***Problem Statement***

**Task I**

1. Explore the game board status shown in Figures 9b, 9c and 9d and discuss the game situation for the two (2) players.

2. Construct the game trees for the game board status shown in Figures 10a through Figure 10d and use them to objectively determine the best set of moves for a selected player (you may select either of player A or B).

3. Discuss any three (3) game status that can result in infinite play round.

**Task II**

Using the Python programming language carry out the following tasks.

i.) Write a program to draw the game board as shown in Figure 6a.

ii.) Update your program to select and play a hole by Player A.

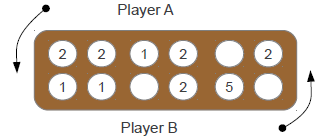
iii.) Update your program to compute the next move for Player A based on ii.).

iv.) Update your program in iii.) to compute a move using a two (2) level game tree and associated game matrix.

v.) Update your program to display the status of the game board after the move computed in iv.)

***TASK 1***

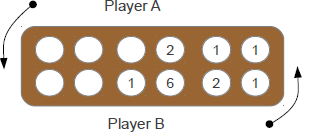
***QUESTION 1, Discussion of Result***



**b: Game state**

The best move for player A is to scoop hole (4) and transverse forward i.e. play. Though it’s possible for player A to capture four seeds by scooping hole (6) to play but in return will lose five seeds to player B when he/she scooped hole (5) to play, thus player B having the upper hand in the game.

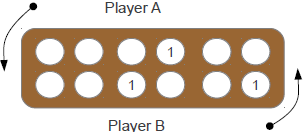
When player A has scooped hole (4) and played i.e. its best move, the best move for player B is to scoop hole (2) to play, though player B will not capture any seed but it will prevent player A from capturing four seeds either by scooping hole (5) or hole (6) to play. So if player B played from another hole, he/she will lose four seeds to player A.



**c: Game state**

For this game state the best move for player A is to scoop hole (1) containing one seed to play, thus preventing player B from capturing two seeds by scooping either hole (5) or hole (6) to play in return. Player A must not make the mistake of scooping hole (3) containing two seeds to play because player B will be given the chance of capturing two seeds even if he/she scoops hole (4), hole (5), or hole (6) to play.

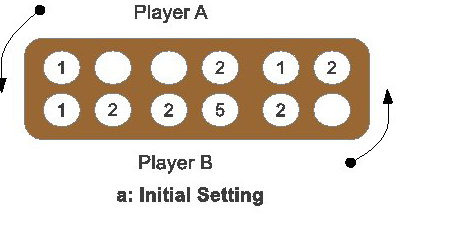
The best move for player B is to resort to “Odù” strategy i.e. by accumulating seeds to hole (4), by playing hole (3). In which the player B setting a trap for player A, in case player A made the mistake in playing hole (3) in return. If player A made the mistake, then player B scoops hole (4) containing seven seeds to play. Thus capturing four seeds in return.



**d: Game state**

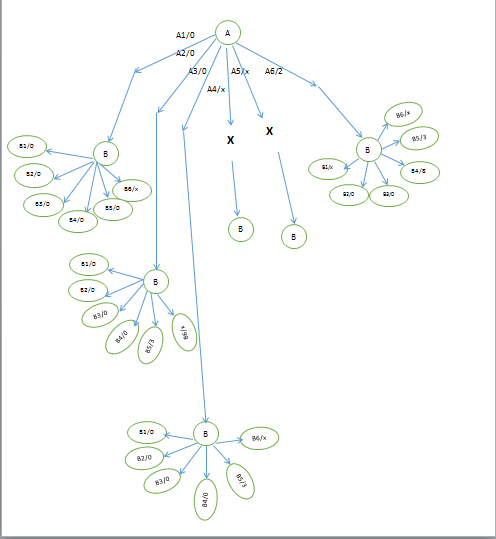
In this game state there are so few seeds on the board and this will either make the process not to end or lead to *infinite moves*, even when each player makes a permissible move, then the best way is to end the game and each player is awarded the seeds on his or her own side of the board.

***QUESTION 2 GAME TREES***

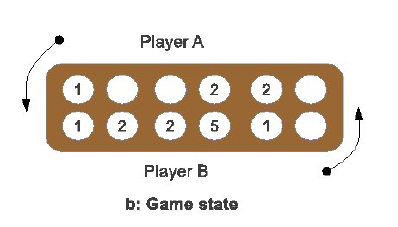


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PLAYER A /PLAYER B** | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **X** |
| **2** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **0** |
| **3** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **0** |
| **4** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **8** |
| **5** | **0**  **0** | **0**  **3** | **0**  **3** | **X** | **X** | **2**  **3** |
| **6** | **0**  **x** | **0**  **x** | **0**  **x** | **x** | **x** | **2**  **x** |

The best set of moves for player A to make is to scoop and play from hole 1 in order to avoid player B from capturing from player A’s hole…



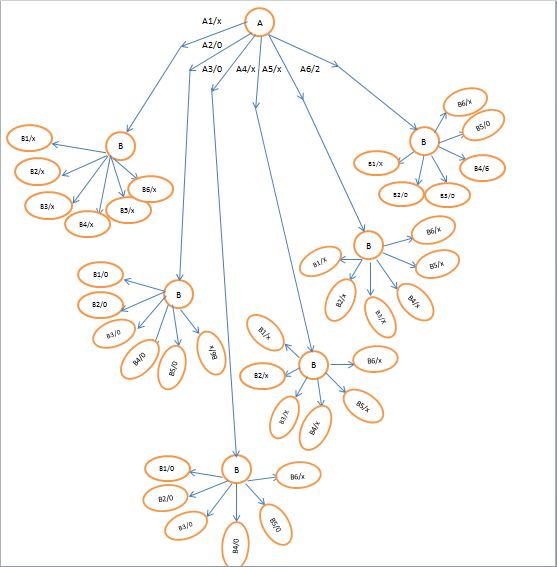
10 b



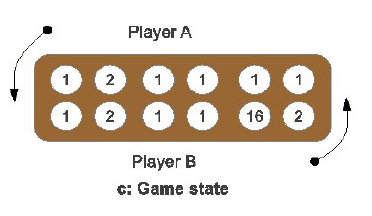
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PLAYER A /PLAYER B** | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | **X** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **0** |
| **2** | **X** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **0** |
| **3** | **X** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **0** |
| **4** | **X** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **0** |
| **5** | **X** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **6** |
| **6** | **X** | **0**  **0** | **0**  **0** | **X** | **X** | **2**  **X** |

BEST MOVE FOR 10 b

The best move is from player B because by playing hole 4, it will capture 6 seeds from player A. but the only way it can’t capture seed from player A is for player A to make a counter move by playing the seed in hole 3.



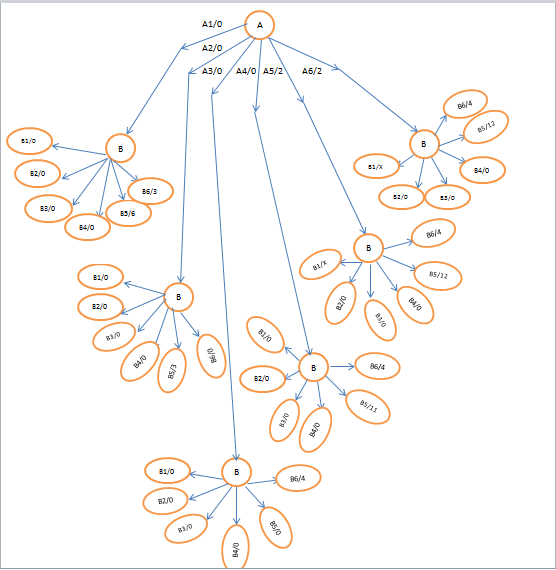
10 c

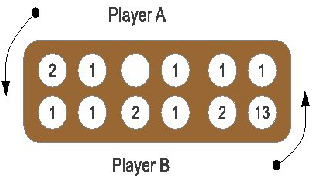


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PLAYER A /PLAYER B** | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | **0**  **0** | **0**  **0** | **0**  **0** | **0**  **0** | **2**  **X** | **2**  **X** |
| **2** | **0**  **0** | **0**  **0** | **0**  **0** | **0**  **0** | **2**  **0** | **2**  **0** |
| **3** | **0**  **0** | **0**  **0** | **0**  **0** | **0**  **0** | **2**  **0** | **2**  **0** |
| **4** | **0**  **0** | **0**  **0** | **0**  **0** | **0**  **0** | **2**  **0** | **2**  **0** |
| **5** | **0**  **6** | **0**  **3** | **0**  **0** | **0**  **11** | **2**  **12** | **2**  **12** |
| **6** | **0**  **3** | **0**  **0** | **0**  **4** | **0**  **4** | **2**  **4** | **2**  **4** |

THE BEST MOVE FOR 10C

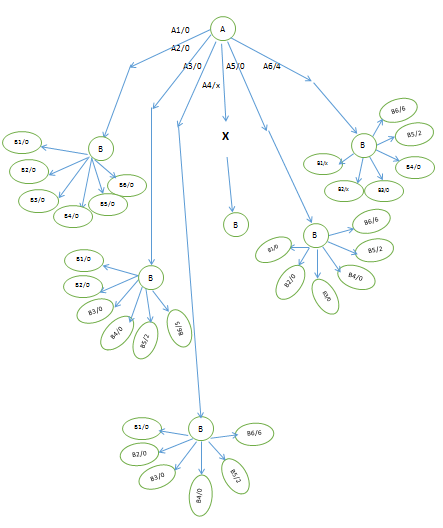
The best move from this section is for player B to play hole 5, which will make it to capture 12 seeds from it opponent but this will be restricted when player A make a counter move by playing seed in hole 3 before player B plays



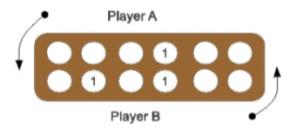
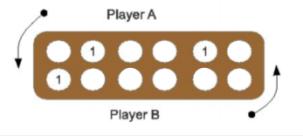


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PLAYER A /PLAYER B** | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **0**  **0** | **4**  **X** |
| **2** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **0**  **0** | **4**  **X** |
| **3** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **0**  **0** | **4**  **0** |
| **4** | **0**  **0** | **0**  **0** | **0**  **0** | **X** | **0**  **0** | **4**  **0** |
| **5** | **0**  **0** | **0**  **2** | **0**  **2** | **X** | **0**  **2** | **4**  **2** |
| **6** | **0**  **0** | **0**  **5** | **0**  **6** | **x** | **0**  **6** | **4**  **6** |

The best set of moves for player A to make is to scoop and play from hole 1 in order to avoid player B from capturing from player A’s hole…



***QUESTION 3***



**Figure 1**  **Figure 2**

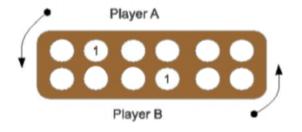


Figure 3

Game statuses that can result in infinite play round

As there are not much seeds remaining on the board it will be very difficult for either of them to capture the seeds of the opponent.

* In figure1, there is just 2 seeds on player A’s side, while there is 1 seed on player B’s side.
* In figure3, there is just 1 seed each at both sides of the board.
* In figure2, there are 2 seeds on player A’s side, while there is just 1 seed at player B’s side.

None of them can capture opponent’s seed, the best option is to agree to end the game, and each player will be awarded the number of seeds on his own side of the board.

***TASK 2***

GOTO “AYO” FOLDER AND READ “GROUP 3 README”.

THIS IS AN AYO-GAME IMPLEMENTED WITH PYTHON

TO RUN OPEN "ayo\_board.py" IN YOUR PYTHON EDITOR

AND RUN IT THE GUI SHLD APPEAR DEN U CAN START SCOOPING

TO PLAY AND COMPUTER WILL PLAY AFTERWARDS

1. Program to draw the game board as shown

import tkinter

import ayo\_player

#from ayo\_computer import game\_play

import time

from random import random

hole\_mat = []

#global board\_mat

#board\_mat = [[(1,1),(1,2),(1,3),(1,4),(1,5),(1,6)],

# [(2,1),(2,2),(2,3),(2,4),(2,5),(2,6)]]

DEFAULT\_BOARD = [[4,4,4,4,4,4],

[4,4,4,4,4,4]]

board\_mat = list(DEFAULT\_BOARD)

#constant sizes

WIN\_WIDTH = 900

WIN\_HEIGHT = WIN\_WIDTH / 3

#variable sizes

B\_WIDTH = WIN\_WIDTH \* 2 / 3

B\_HEIGHT = B\_WIDTH / 3

X\_ALLOWANCE = B\_WIDTH \* 1 / 60

Y\_ALLOWANCE = B\_HEIGHT \* 1 / 20

WIN\_SIZE = str(WIN\_WIDTH)+"x"+str(WIN\_HEIGHT)

SIZE\_X = B\_WIDTH/(len(board\_mat[0]))

SIZE\_Y = B\_HEIGHT/(len(board\_mat))

#popup class

class popup(tkinter.Frame):

def say\_hi(self):

print ("hi there, everyone!")

def createWidgets(self):

self.messageVar = tkinter.StringVar()

self.messageVar.set(self.message)

tkinter.Label(self, text=self.message, foreground="red").grid(column=0,row=0,sticky=(tkinter.N,tkinter.W, tkinter.S, tkinter.E))

#self.hi\_there["command"] = self.say\_hi

#self.hi\_there.pack({"side": "left"})

def \_\_init\_\_(self, master=None, message="Hello"):

tkinter.Frame.\_\_init\_\_(self, master)

self.message = message

self.pack()

self.createWidgets()

def createpopup(message="Hello"):

tk = tkinter.Tk()

cpopup = popup(master=tk, message=message)

tk.mainloop()

class ayo\_board(tkinter.Frame):

GAME\_ON = 1

GAME\_OVER = 0

MAX\_CAPTURELESS = 30

def \_\_init\_\_(self, parent):

tkinter.Frame.\_\_init\_\_(self,parent)

self.parent = parent

self.initialise()

self.turn = 2

self.p1 = ayo\_player.ayo\_player(1,ayo\_player.ayo\_player.COMPUTER,level=2)

self.p2 = ayo\_player.ayo\_player(2,ayo\_player.ayo\_player.HUMAN,level=2)

self.next\_player = self.p2

self.draw\_scores((0,0))

self.game\_state = self.GAME\_ON

self.captureless\_count = 0

self.board\_mat = [1,2]

self.board\_mat[0]= list(DEFAULT\_BOARD[0])

self.board\_mat[1]= list(DEFAULT\_BOARD[1])

self.matrix\_copy = list(self.board\_mat)

def initialise(self, board\_mat=board\_mat):

self.create\_menu()

self.parent.title("Ayo Board")

self.pack(fill=tkinter.BOTH, expand=1)

self.board\_mat = [1,2]

self.board\_mat[0]= list(DEFAULT\_BOARD[0])

self.board\_mat[1]= list(DEFAULT\_BOARD[1])

print (self.board\_mat)

self.columnconfigure(0, weight=1)

self.rowconfigure(0, weight=1)

self.columnconfigure(1, weight=1)

self.scoreboard = tkinter.Canvas(self, relief=tkinter.SUNKEN, bg="white", width=SIZE\_X+(1+1)\*X\_ALLOWANCE, height=B\_HEIGHT+(2+1)\*Y\_ALLOWANCE) #Add allowance to the ends

self.canvas = tkinter.Canvas(self, relief=tkinter.SUNKEN, bg="brown", width=B\_WIDTH+X\_ALLOWANCE, height=B\_HEIGHT+Y\_ALLOWANCE) #Add allowance to the ends

self.canvas.grid(column=0,row=0)#, sticky=(Tkinter.N, Tkinter.W))

self.scoreboard.grid(column=1,row=0)

#self.score\_canvas = Tkinter.Canvas(self, relief=Tkinter.SUNKEN, bg="brown", width=SIZE\_X, height=B\_WIDTH+Y\_ALLOWANCE)

#self.score\_canvas.pack(side=Tkinter.BOTTOM, expand=True, fill=Tkinter.BOTH, padx=(WIN\_WIDTH+2\*B\_WIDTH-X\_ALLOWANCE)/2, pady=int((WIN\_HEIGHT-B\_HEIGHT-Y\_ALLOWANCE)/2))

#self.canvas.pack(side=Tkinter.BOTTOM, expand=True, fill=Tkinter.BOTH, padx=int((WIN\_WIDTH-B\_WIDTH-X\_ALLOWANCE)/2), pady=int((WIN\_HEIGHT-B\_HEIGHT-Y\_ALLOWANCE)/2))

print ('canvas created')

# draw holes

self.hole = [['','','','','',''],['','','','','','']]

self.text\_id = [['','','','','',''],['','','','','','']]

print (DEFAULT\_BOARD)

for j, line in enumerate(self.board\_mat, 1): # j is player id (1,2)

for i, value in enumerate(line, 1): # i is hole position (1,2,3,4,5,6)

x1 = (i-1)\*SIZE\_X + X\_ALLOWANCE

y1 = (j-1)\*SIZE\_Y + Y\_ALLOWANCE

x2 = (i)\*SIZE\_X

y2 = (j)\*SIZE\_Y

self.canvas.create\_oval(x1, y1, x2, y2, fill="black")

self.hole[j-1][i-1] = (x1, y1, x2, y2)

c\_x, c\_y = self.center\_pos(x1,y1,x2,y2) # get center of circles, which are approximately the text positions and will eventually become the centres of the images containing seeds

self.text\_id[j-1][i-1] = self.canvas.create\_text(c\_x, c\_y, anchor=tkinter.CENTER, font="Purisa", text=str(value), fill="white")

#print self.text\_id[j-1][i-1]

self.canvas.bind("<ButtonPress-1>", self.clicked)

#draw scoreboard

self.scoretext\_id = [0,0]

hole\_1\_pos = ((X\_ALLOWANCE,Y\_ALLOWANCE), (SIZE\_X+X\_ALLOWANCE, SIZE\_Y+X\_ALLOWANCE))

hole\_2\_pos = ((X\_ALLOWANCE,SIZE\_Y+2\*Y\_ALLOWANCE), (SIZE\_X+X\_ALLOWANCE, 2\*SIZE\_Y+2\*Y\_ALLOWANCE))

self.scoreboard.create\_oval(hole\_1\_pos[0][0],hole\_1\_pos[0][1], hole\_1\_pos[1][0], hole\_1\_pos[1][1], fill="black")

self.scoreboard.create\_oval(hole\_2\_pos[0][0],hole\_2\_pos[0][1],hole\_2\_pos[1][0],hole\_2\_pos[1][1], fill="black")

self.scoretext\_id[1-1] = self.scoreboard.create\_text((hole\_1\_pos[0][0]+hole\_1\_pos[1][0])/2,(hole\_1\_pos[0][1]+hole\_1\_pos[1][1])/2, font="Purisa", fill="white")

self.scoretext\_id[2-1] = self.scoreboard.create\_text((hole\_2\_pos[0][0]+hole\_2\_pos[1][0])/2,(hole\_2\_pos[0][1]+hole\_2\_pos[1][1])/2, font="Purisa", fill="white")

self.countdown\_text\_id = self.scoreboard.create\_text(X\_ALLOWANCE+SIZE\_X/2,SIZE\_Y+1.5\*Y\_ALLOWANCE, fill="black")

#self.game\_play = game\_play()

def create\_menu(self):

self.menubar = tkinter.Menu(self.parent)

self.parent.config(menu=self.menubar)

self.fileMenu = tkinter.Menu(self.menubar)

self.fileMenu.add\_command(label="Exit", command=self.on\_exit)

self.gameMenu = tkinter.Menu(self.menubar)

self.gameMenu.add\_command(label="Restart", command=self.restart)

self.gameMenu.add\_command(label="Switch P1 mode", command=self.switchmode1)

self.gameMenu.add\_command(label="Switch P2 mode", command=self.switchmode2)

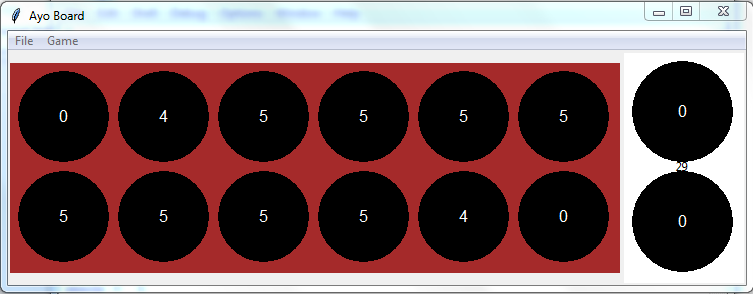
self.menubar.add\_cascade(label="File", menu=self.fileMenu)

self.menubar.add\_cascade(label="Game", menu=self.gameMenu)

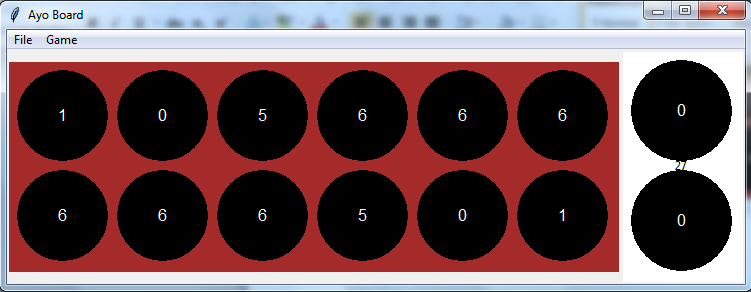
***GAME INTERFACE***

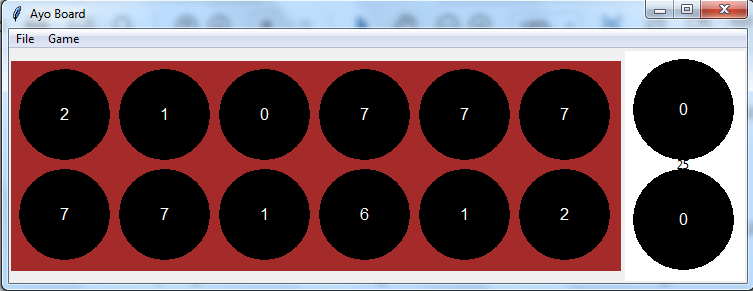


1. Update your program to select and play a hole by Player A.



3) gtUpdate your program to compute the next move for Player A based on 2)



4)

**TECHNOLOGICAL INTERPRETATION AND APPLICATION OF RESULTS**

The game is design on python platform which give the design view of the real life system to be designed based on Artificial Intelligent system.

The knowledge gotten from the ayo game is applicable in real life situations e.g.

* Chess game, strategy in a chess game, a player tries to maximize their winning chance by trying to calculate his/her opponent’s moves and at the same time plan a move to counter each of the moves previously calculated. So as to make the moves that will give him a likely victory.
* Commerce setting, a situation between a buyer and seller, the buyer aims at getting more goods at a lower price, while the seller aims at getting more profit with selling fewer goods. Each of them aims at winning the other, but at the very end one of them wins.

**SUMMARY AND CONCLUSION**

This experiment was based on a subfield of AI (gaming) and a typical example is the African ayo-game (awale). We have been able to determine game states, game trees and also implement the game with a programming language (PYTHON).

We have been able to evolve a machine-Ayo game player that can emulate human expertise in Ayo game playing in order to expand our understanding of human intelligent processes through computer simulation. The Ayo player is computationally effective and can improve AI performance and make computer players more adaptable and responsive.