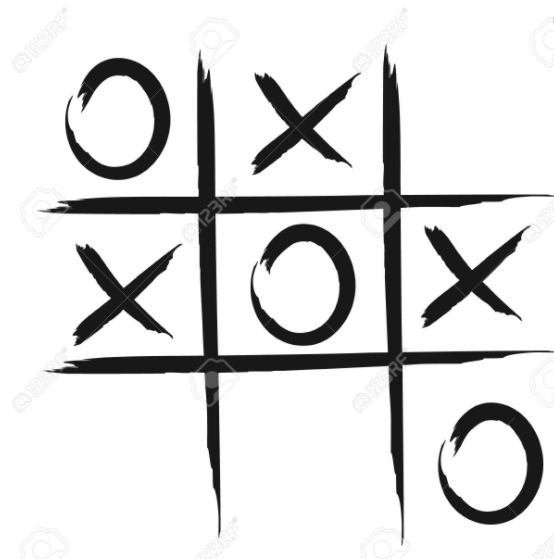


TIC TAC TOE GAME

By ALVIN K JOBI





AKNOWLEDGEMENT

First of all, I would like to thank Principal Rev. Father Antony Madavanakadu for giving me a great opportunity to show up my capabilities. I also thank our vice principal Mrs.Vinitha Mendez for her tremendous support.

Without the care and help given by Mrs. Mariamma Thomas this project would not have been possible for me. I would also like to show my gratitude to all my friends who helped me for the success of this project.

As the outset, I especially am indebted to lord Almighty for flooding his grace and blessings on me in all aspects to terminate this project in fine manner and to the schedule

Thank you everyone for your great support

CONTENTS

1. INTRODUCTION
2. SYSTEM REQUIREMENTS AND SPECIFICATIONS
3. DATABASE DICTIONARY
4. SOURCE CODE
5. OUTPUT
6. BIBLIOGRAPHY

INTRODUCTION

This project is about a game named Tic Tac Toe, which was played in the past First Century BC by the Roman Empire.

The is a Fun game played by 2 players, the game is played by marking ‘X’s and ‘O’s on the spaces provided in a 3x3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical or diagonal row is the winner.

Here the data in about the players is collected through the Login window and this is stored in a temporary table in MySQL, which is later stored into a permanent table.

The Project is basically aimed to provide the users Entertainment, according to this project the user can check the highest score, scores of other players and can terminate the game at any point of the game.

SYSTEM REQUIREMENTS AND SPECIFICATIONS

HARDWARE SPECIFICATIONS

1. SYSTEM-IBM-COMPATIBLE PC
2. PROCESSOR – INTEL(R) CORE
3. SPEED – 2.0 GHZ
4. MEMORY – 4 GB RAM
5. HARD DISK – 500 GB

SOFTWARE SPECIFICATIONS

1. OPERATING SYSTEM – WINDOWS 10
2. DEVELOPMENT ENVIRONMENT – PYTHON IDLE
3. LANGUAGE USED – PYTHON
4. FRONT END – PYTHON
5. BACK END – MYSQL

DATA DICTIONARY

TABLE 1 : To store the details of various players in order to show the player history

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Remarks** |
| name | char(50) | The names of the players are stores here according to their player ID permanently |
| point | int(11) | The points of the respective players are stored here depending upon their score |

TABLE 2 : To store the score of the single player’s

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Remarks** |
| name | char(50) | The names of the players are stores here according to their player ID temporarily |
| point | int(11) | The points of the respective players are stored here depending upon their score |
| game | char(29) | (just added to make the coding easier) |

SOURCE CODE

import mysql.connector as mc

import matplotlib.pyplot as pl

import tkinter

from tkinter import \*

from tkinter import messagebox

def main(a,b,password):

def helpp():

for key in board\_keys:

Board[key]=' '

print("""\n\n\t\tHI GUYS DO YOU LIKE TO PLAY TIC TAC TOE

\t\tPRE KNOWLEDGE ABOUT THE GAME :

\t\tTHE POSITIONS ARE MENTIONED AS BELOW

\n

1 | 2 | 3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4 | 5 | 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7 | 8 | 9

\n \tAND IN ANYWAY THE FIRST PERSON WILL BE ASSIGNED 'X'

\tAND THE SECOND AS 'O'

\t\tTHEN LET'S GET STARTED""")

plays=0

Board = {7: ' ' , 8: ' ' , 9: ' ' ,4: ' ' , 5: ' ' , 6: ' ' ,1: ' ' , 2: ' ' , 3: ' ' }

board\_keys = []

for key in Board:

board\_keys.append(key)

if plays==0:

mcon=mc.connect(host="localhost",user="root",passwd=password,database='mysql')

mcur=mcon.cursor()

try:

mcur.execute("create table ttt(name char(50) primary key,point int)")

except:

pass

P1=b

P2=a

def gameboard(b):

print("\t\t"+b[1] + '|'+ b[2] +'|'+ b[3] )

print('\t\t------------')

print("\t\t"+b[4] + '|'+ b[5] +'|'+ b[6] )

print('\t\t------------')

print("\t\t"+b[7] + '|'+ b[8] +'|'+ b[9] )

def table(a,b):

mcur.execute("create table score(name char(20),score int,game char(29))")

for i in range(0,len(a)):

mcur.execute("insert into score values(%s,%s,'tic')",(a[i],b[i]))

mcon.commit()

def graph():

mcur.execute("select \* from score")

data=list(mcur.fetchall())

n=[]

s=[]

for k in data:

n.append(k[0])

s.append(k[1])

pl.pie(s,labels=n,autopct='%0.1f%%',shadow=True)

pl.show()

mcur.execute("drop table score")

x=[] #points scored

o=[]

def game():

turn='X'

count=0

for i in range(0,10):

validity=0

gameboard(Board)

print("\tITS THE TURN OF "+ turn +". TELL THE POSITION.")

try:

move=int(input())

if move>=1 and move<=9:

if Board[move]==' ':

Board[move]=turn+" "

count+=1

else:

messagebox.showinfo("Error","THE PLACE WAS ALREADY FILLED\nPLEASE TRY AGAIN")

continue

if count>=5:

if Board[1]==Board[2]==Board[3]=='X ' or Board[1]==Board[2]==Board[3]=='O ':

if Board[1]=="X ":

name=P1

x.append("1")

else:

name=P2

o.append("1")

print('\n'+name + " IS THE WINNER")

gameboard(Board)

break

elif Board[4]==Board[5]==Board[6]=='X ' or Board[4]==Board[5]==Board[6]=='O ':

if Board[4]=="X ":

x.append("1")

name=P1

else:

o.append("1")

name=P2

print('\n'+name+ " IS THE WINNER")

gameboard(Board)

break

elif Board[7]==Board[8]==Board[9]=='X ' or Board[7]==Board[8]==Board[9]=='O ':

if Board[7]=="X ":

x.append("1")

name=P1

else:

o.append("1")

name=P2

print('\n'+name+ " IS THE WINNER")

gameboard(Board)

break

elif Board[1]==Board[4]==Board[7]=='X ' or Board[1]==Board[4]==Board[7]=='O ':

if Board[1]=="X ":

x.append("1")

name=P1

else:

o.append("1")

name=P2

print('\n'+name+ " IS THE WINNER")

gameboard(Board)

break

elif Board[2]==Board[5]==Board[8]=='X ' or Board[2]==Board[5]==Board[8]=='O ':

if Board[2]=="X ":

x.append("1")

name=P1

else:

o.append("1")

name=P2

print('\n'+name+ " IS THE WINNER")

gameboard(Board)

break

elif Board[3]==Board[6]==Board[9]=='X ' or Board[3]==Board[6]==Board[9]=='O ':

if Board[3]=="X ":

x.append("1")

name=P1

else:

o.append("1")

name=P2

print('\n'+name+ " IS THE WINNER")

gameboard(Board)

break

elif Board[1]==Board[5]==Board[9]=='X ' or Board[1]==Board[5]==Board[9]=='O ':

if Board[1]=="X ":

x.append("1")

name=P1

else:

o.append("1")

name=P2

print('\n'+name+ " IS THE WINNER")

gameboard(Board)

break

elif Board[3]==Board[5]==Board[7]=='X ' or Board[3]==Board[5]==Board[7]=='O ':

if Board[3]=="X ":

x.append("1")

name=P1

else:

o.append("1")

name=P2

print('\n'+name+ " IS THE WINNER")

gameboard(Board)

break

if count==9:

print("\n\t\*\*\*\*\*GAME OVER ITS A TIE\*\*\*\*\*\*")

#gameboard(Board)

x.append("1")

o.append("1")

break

if turn=='X':

turn='O'

else:

turn='X'

else:

messagebox.showinfo("Error","Input is out of range!!!!!.")

try:

mcur.execute("drop table score")

quit()

except:

quit()

except:

messagebox.showinfo("Error","Input is out of range!!!!!.")

try:

mcur.execute("drop table score")

quit()

except:

quit()

if validity==0:

z=input("\n\tDO YOU WANT TO RESTART THE GAME(Y/N) :")

if z=="y" or z=="Y":

for key in board\_keys:

Board[key]=' '

game()

elif z=="N" or z=="n":

print("\n\t\tTHANK YOU FOR PLAYING")

name=(P1,P2,)

score=(len(x),len(o))

for i in range(0,2):

try:

mcur.execute('insert into ttt values(%s,%s)',(name[i],score[i]))

mcon.commit()

except:

if P1!=P2:

p=input("""Your PLAYER ID '""" + name[i]+"""' already exist,

Do you want to over write your previous history(Y/N) :""")

if p=='Y' or p=='y':

d='delete from ttt where name ="' + name[i] +'"'

mcur.execute(d)

mcur.execute('insert into ttt values(%s,%s)',(name[i],score[i]))

mcon.commit()

else:

pass

for key in board\_keys:

Board[key]=' '

try:

mcur.execute("drop table score")

except:

pass

table(name,score)

graph()

else:

messagebox.showinfo("Error","Input is out of range!!!!!.")

try:

mcur.execute("drop table score")

quit()

except:

quit()

def d():

s=input("\nDo you want to view your individual score(Y/N) :")

return s

def history():

try:

mcur.execute('select \* from ttt order by point desc')

f=mcur.fetchall()

if len(f)==0:

messagebox.showinfo("Error","No players played yet.")

else:

print('\n\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*TOP BEST PLAYERS\*\*\*\*\*\*\*\*\*\*\*\*\*\n')

for i in range(5):

c=f[i]

for i in range(2):

print('\t\*',"{:^15}".format(c[i]),end="\*")

print('\n')

print('\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

except:

print('\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

mcur.execute('select \* from ttt order by point desc')

f=mcur.fetchall()

if len(f)!=0:

s=d()

m=[]

for i in f:

m.append(i[0])

if s=='y' or s=='Y':

name=input('\nEnter the PLAYER ID :')

if name in m:

for i in f:

if i[0]==name:

c=['NAME','SCORE']

print('\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

for f in range(2):

print('\t\*',"{:^15}".format(c[f]),end="\*")

print('\n')

print('\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

for f in range(2):

print('\t\*',"{:^15}".format(i[f]),end="\*")

print('\n')

print('\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

else:

pass

else:

messagebox.showinfo("Error","No such PLAYER ID.")

elif s=='n' or s=='N':

print("\n\tTHANK YOU...")

else:

messagebox.showinfo("Error","Invalid input.")

def end(master):

try:

mcur.execute("drop table score")

exit()

except:

exit()

def main\_window():

root=tkinter.Tk()

root.geometry("600x500")

root.title("TICTACTOE")

root.configure(background="BLACK")

r=tkinter.Button(root,text='GAME',height=1,width=20,command=lambda:game(),

border=15,bg='black',fg='yellow',font=('arial',20,'bold','underline'),

activeforeground='red',activebackground='yellow')

r2=tkinter.Button(root,text='HELP',height=1,width=20,command=lambda:helpp(),

border=15,bg='grey',fg='yellow',font=('arial',20,'bold','underline'),

activeforeground='red',activebackground='yellow')

r3=tkinter.Button(root,text='QUIT',height=1,width=20,command=lambda:end(root),

border=15,bg='grey',fg='yellow',font=('arial',20,'bold','underline'),

activeforeground='red',activebackground='yellow')

Tops = Frame(root, bg='BLACK', bd=20, pady=20, relief=RIDGE)

Tops.pack(side=TOP)

lblTitle = Label(Tops, font=('Copperplate Gothic Bold', 25, 'bold'),

text=' <<<TIC TAC TOE>>> ',

bd=15, bg='yellow',fg='BLACK', justify=CENTER)

lblTitle.grid(row=0)

r5=tkinter.Button(root,text='HISTORY',height=1,width=20,command=lambda:history(),

border=15,bg='black',fg='yellow',font=('arial',20,'bold','underline'),

activeforeground='red',activebackground='yellow')

r.pack()

r2.pack()

r5.pack()

r3.pack()

root.mainloop()

main\_window()

def last(master):

master.destroy()

confirm=tkinter.Tk()

def RESUME():

confirm.destroy()

first()

def QUIT():

quit()

def confirmation():

confirm.geometry("580x370")

confirm.title("confirmation")

confirm.configure(bg="black")

Tops = Frame(confirm, bg='black', bd=20, pady=20, relief=RIDGE)

Tops.pack(side=TOP)

l = Label(confirm, text = "\nDO YOU REALLY \nWANT TO QUIT THIS GAME??")

l.config(font =("Copperplate Gothic Bold",16,'italic','bold','underline'),bg='black',fg='yellow')

l.pack()

lblTitle = Label(Tops, font=('Copperplate Gothic Bold', 25, 'bold'), text=' <<<CONFIRM>>> ',

bd=15, bg='yellow',fg='BLACK', justify=CENTER)

lblTitle.pack()

r=tkinter.Button(confirm,text='RESUME',height=1,width=10,command=lambda:RESUME(),

border=15,bg='black',fg='yellow',font=('arial',20,'bold','underline'),

activeforeground='red',activebackground='yellow')

r2=tkinter.Button(confirm,text='QUIT',height=1,width=10,command=lambda:QUIT(),

border=15,bg='black',fg='yellow',font=('arial',20,'bold','underline'),

activeforeground='red',activebackground='yellow')

r.pack(side=RIGHT,padx=30)

r2.pack(side=LEFT,padx=30)

confirm.mainloop()

confirmation()

def first():

loginwn=tkinter.Tk()

def log\_in():

loginwn.geometry("580x350")

loginwn.title("Log in")

loginwn.configure(bg="black")

Tops = Frame(loginwn, bg='black', bd=20, pady=20, relief=RIDGE)

Tops.pack(side=TOP)

lblTitle = Label(Tops, font=('Copperplate Gothic Bold', 25, 'bold'), text=' <<<LOGIN>>> ',

bd=15, bg='yellow',fg='BLACK', justify=CENTER)

lblTitle.grid(row=0)

l1=tkinter.Label(loginwn,text=" SECOND PLAYER NAME: ",relief="groove",bg='black',

fg='yellow',bd=5)

l2=tkinter.Label(loginwn,text=" FIRST PLAYER NAME: ",relief="groove",bg='black',

fg='yellow',bd=5)

l3=tkinter.Label(loginwn,text=" ENTER YOUR MYSQL PIN: ",relief="groove",bg='black',

fg='yellow',bd=5)

l3.pack(side="top")

e3=tkinter.Entry(loginwn,show="\*") # mysql password e3 pin

e3.pack(side="top")

l2.pack(side="top")

e2=tkinter.Entry(loginwn) #player 1 name

e2.pack(side="top")

l1.pack(side="top")

e1=tkinter.Entry(loginwn) #player 2 name

e1.pack(side="top")

b=tkinter.Button(loginwn,text="SUBMIT",

command=lambda: check\_log\_in(loginwn,e1.get().strip(),e2.get().strip(),e3.get().strip()),

bg='black',fg='yellow',bd=5,relief="groove")

b.pack(side="top")

b1=tkinter.Button(loginwn,text="<<QUIT>>",command=lambda: quit1(),bg='black',

fg='yellow',bd=5,relief="groove")

b1.pack(side='top')

loginwn.mainloop()

def check\_log\_in(master,player2,player1,pin):

if pin =='root':

if player1=='':

messagebox.showinfo("Error","FIRST PLAYER ID not given!!!!!.")

elif player2=='':

messagebox.showinfo("Error","SECOND PLAYER ID not given!!!!!.")

else:

master.destroy()

main(player2,player1,pin)

elif pin != 'root':

messagebox.showinfo("Error","INCORRECT PASSWORD.")

quit()

def quit1():

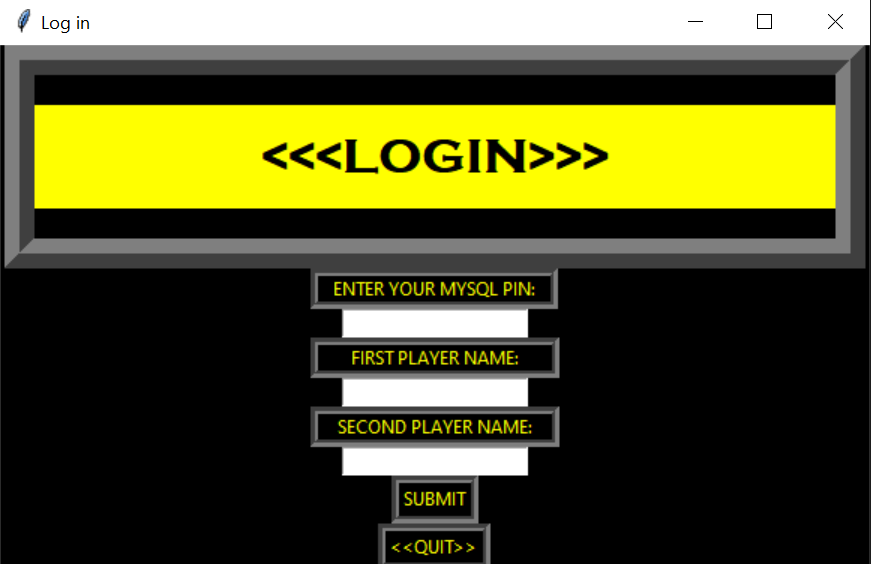
last(loginwn)

log\_in()

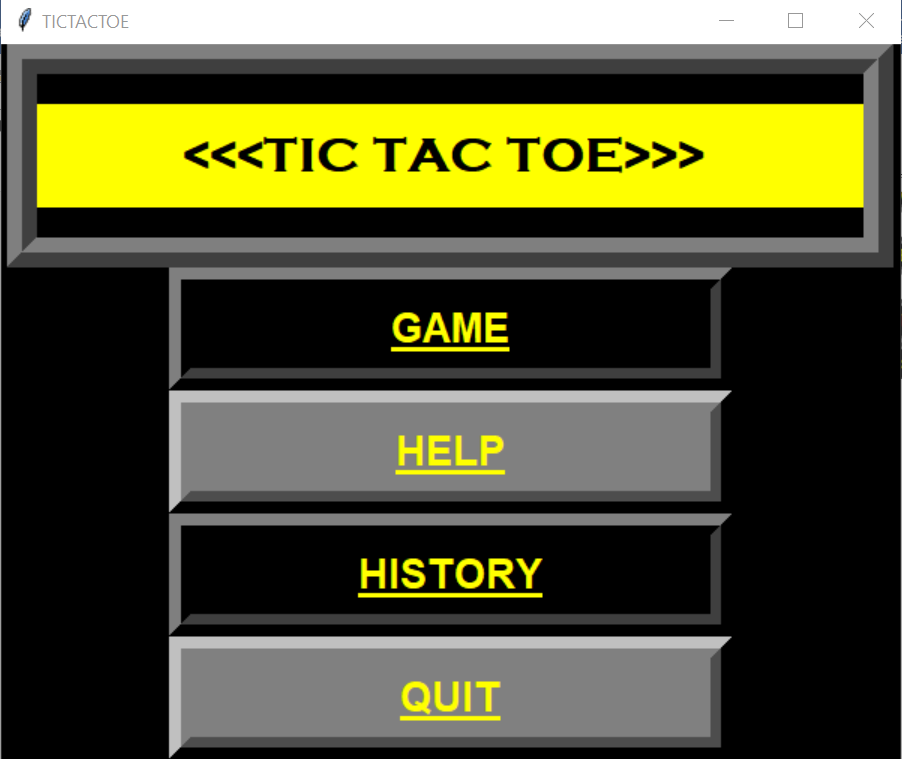
first()

OUTPUT

LOGIN WINDOW

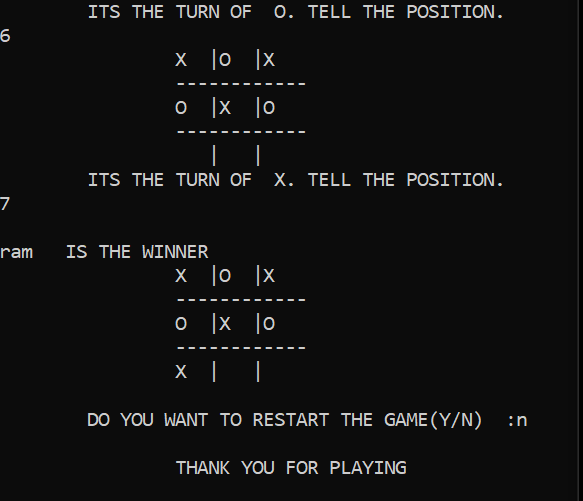
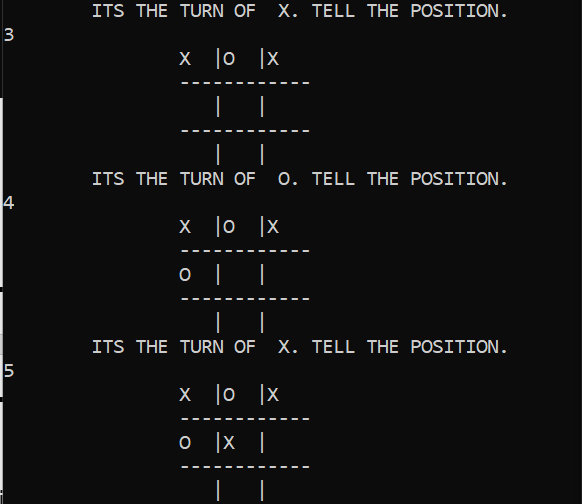
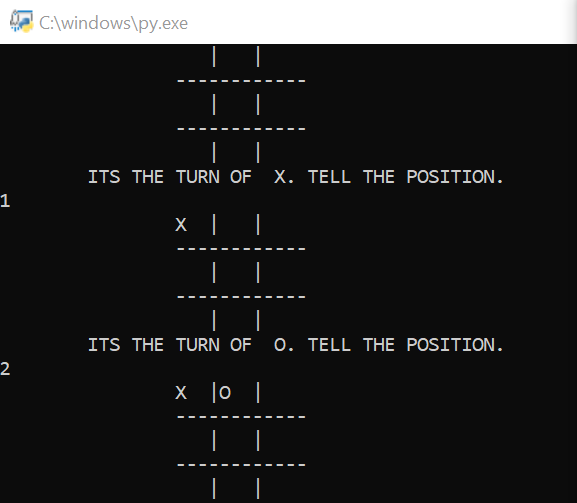


MAIN WINDOW

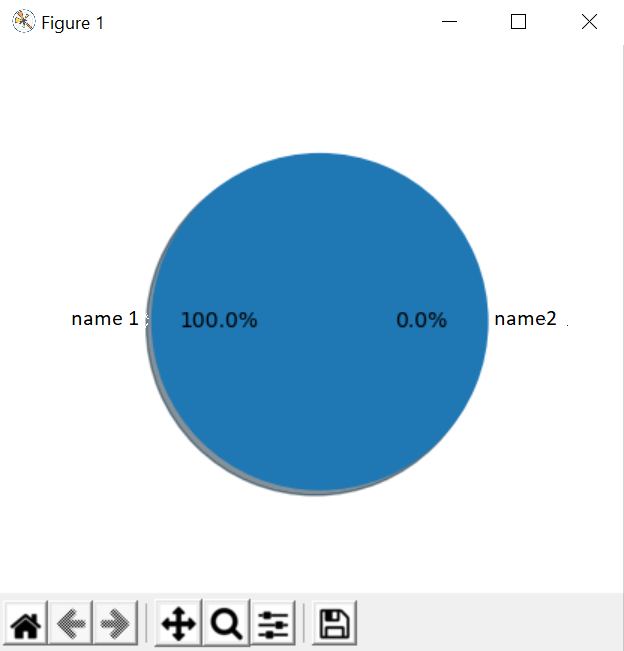


CONFIRMATION WINDOW (Only shown when there is direct quiting from LOGIN window)

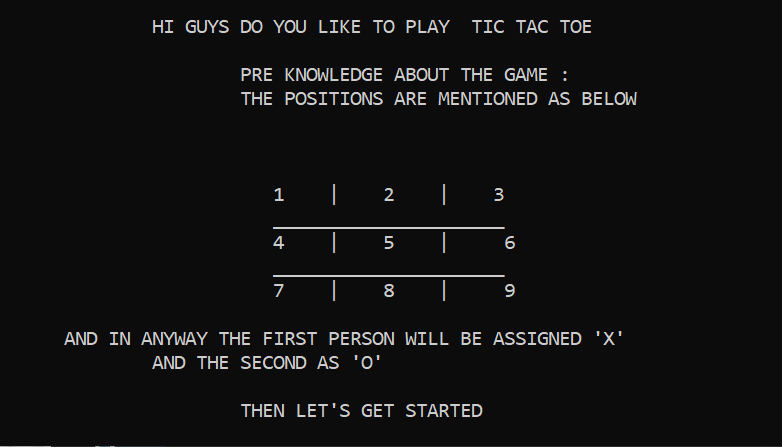


Option 1 GAME:  
This allows a dual player game 

Also show the graph related to the previous game

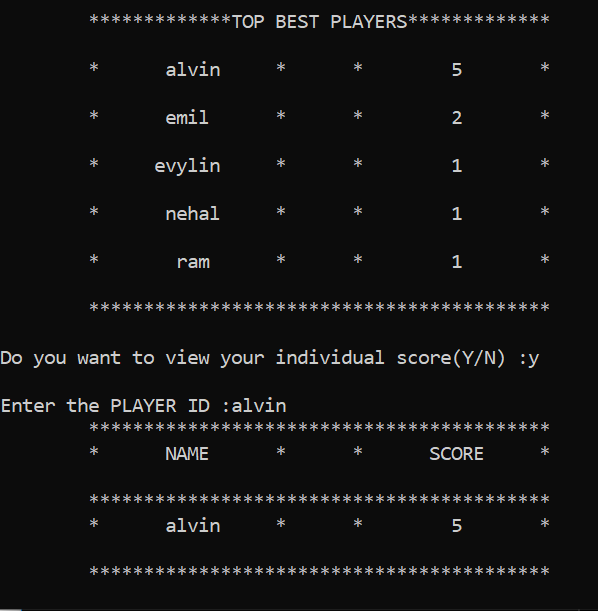


Option 2 HELP:  
Tells about the game



Option 3 HISTORY:

This shows the top 5 best played scores, and also we ac check our personal score.



Option 4 QUIT:

This directly terminate the game.

BIBILIOGRAPHY

This project would not have been a success without a below mentioned source

1. Sultan Chand Computer Science with Python by Sumita Arora
2. Tutorials point

<https://www.tutorialspoint.com/python/python_gui_programming.htm>

1. Many YouTube videos