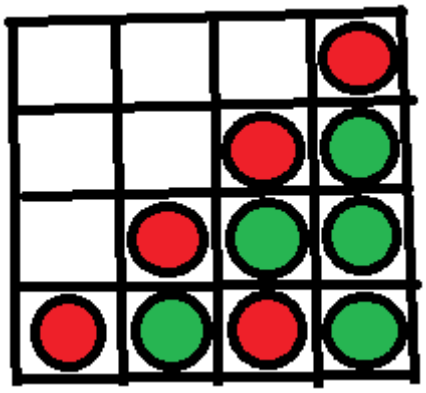
CONNECT 4

By Maalavika C S



BONAFIDE CERTIFICATE

Certified to be the Bonafide Record of work done by Miss Maalavika C S of Class XII in SRI SANKARA SENIOR SECONDARY SCHOOL, Chennai

During the Year 2020-2021

Dated _____

Subject Teacher _____

Chennai

Submitted for Practical Examination

held in _____

at _____CHENNAI

Dated _____

Examiner _____

Seal

ACKNOWLEDGEMENTS

I would like to express my gratitude and deep regards to my teachers Mrs. R. Revathy and Mrs. S. Vidyalakshmi for their guidance and encouragement given throughout the course of this project. I would also like to thank Mr. Jose Cherian for his support in providing the Lab Equipments.

I further thank our Principal Mrs. Mita Venkatesh, and the staff members of Sri Sankara Senior Secondary School, for the apparatus and valuable information provided by them.

I am obliged to my parents, my brother and my classmates for helping me finalize this project report.

INDEX

SYNOPSIS

ABOUT THE GAME

Connect4 is a game inspired from the Two-player Board game Connect Four. The two players take turns to drop coins of 2 colours from the top of the 8×8 array. The coins slide down and occupy the lowermost available position within the column. The first one to get four coins straight in a row, column or diagonal wins.

For my project, I chose the topic of e-Business and developed the Connect4 game. In this game, an array of 8×8 is displayed with column numbers provided at the top of each column, numbered from 0 to 7. Initially, the players are asked to enter their names. Once the game begins, the players can type the column in which they want to drop their coin, which results in the coin being found in the lowermost row possible.

If an entire column is filled, an error message is displayed if further coins are dropped there.

If the entire grid is filled, the game is declared as a Tie.

Once the game is completed, a Score Board is shown, displaying the Total games played between the current players and number of games won by each player. This is followed by the Leader Board table where the details of the Top 3 players are shown.

Once this is done, the players are given an option to play with the same person again, play with a different person or quit the game.

In the beginning of the game, I have also inserted music and an animation showing CONNECT 4. Also, each time the player enters some details or an error message is being shown, a warning sound emanates from the computer.

Logic of the code:

Initially, a list, say A, of the coordinates is created, with each coordinate represented as (x,y). The coordinates of the corresponding points in the screen is stored in a different list, say B. A dictionary D mapping every coordinate in A with ‘none’ is created.

Each time a player enters a column number, the corresponding lowermost coordinate (least y) of the column in A is removed. In D, the coordinate is mapped to the player name, now, instead of to ‘none’. So, D has the map of each point to its corresponding player or ‘none’. For plotting the point, the corresponding coordinate in B is taken.

In each iteration, every row, column and diagonal is checked by checking if the corresponding coordinates in D is mapped to one player, using a nested for loop.

If 4 coins are found together, the winner is declared and the game ends. Or else, the same process happens again, until the winner is found or the game is a tie.

The players are allowed to quit the game at any time by entering the letter ‘q’ in the input box. In every iteration, the game checks for this input and ends the current game, if found.

For error messages:

I have created a function called inputval() for this purpose. For each input from the users, the set of values allowed is passed, along with the input message. So, this function asks the users for the input with this message and if the value entered is different from the ones allowed, an error

message is shown and the user is asked until a valid input is received.

Using the above, when a column is completely filled, the allowed values passed doesn’t contain this column. So, when this column is entered, an error message is shown.

SOFTWARE USED

Front-end Tools:

Turtle module:

For the User interface, I have used Turtle module provided by the Python Library, since my game involves animations and graphics. The following functions from the Turtle Module were used. To name a few:

- screen() – to set my screen size, background colour
- write() – to display messages
- textinput() - to get user input in a textbox
- fd(),bd() - to move forward/backward
- circle(),dot() – to draw the required shapes
- setpos() - to set the position of the turtle
- color() - to change the colour of the turtle

Turtle facilitated me to make the game colourful, attractive and user- friendly.

Winsound module:

To add music and error message alerts to my game, I used the PlaySound() function from the Winsound module.

For doing this, I had to have music stored in .wav format. Calling the function PlaySound(<filename.wav>,winsound.SND_ASYNC) makes the sound play, without any interruptions to the game.

Back-End Tools:

Python:

The entire code was written in the Python language, learnt as a part of our curriculum in Class 12. Python provides various modules to facilitate code creation and makes the programme concise.

Time module:

The time module, with the function sleep(), helped me in freezing the screen for a particular time duration when the users had to read some message.

Pickle module:

To store details of the players in a Binary file, I used this module. More details about this has been given in the subsequent pages.

Mysql connectivity:

I used the mysql.connector module present in the Python library to connect to the mysql databases in order to store data about each player and the games played by them. More details about this has been given in the subsequent pages.

DETAILS OF FILE HANDLING

Storing information about the players, the statistics of who has played with whom and who has won more times, was necessary to build a complete game, where the players could challenge their capabilities and master the game.

For this purpose, I decided to use a Binary (.dat) file so that I could store the data in the form of a dictionary, mapping the two players with the details of their games, so far played. This way, I could store data about each pair of players.

(player1,player2):(number of games won by player1,player2)

Each time, a player won, I could just add one to the number of games, he/she played. For the total number of games played, I simply could add the two numbers.

I used the pickle module, compatible with Python, so that I could read the details already present in the file using pickle.load() function, alter the necessary details and insert

the dictionary back into the same file using pickle.dump() function.

To reduce duplication in the player names, so that more precise data could be provided, I stored all the names in capital case.

DETAILS OF MYSQL CONNECTIVITY

In order to store data about each player, number of games they have played and won, overall, I used Mysql Database. To connect it to the Python code, I used the module mysql.connector, in which a unique connection can be made to the table required to procure, change and delete data. Similar to using Mysql, queries can be passed from python using the cursor object.

After every game, I updated the data in the table and if necessary, created new records for new players, using the “update” query. Also, I displayed the top 3 players in the database using a comparison ratio called ‘Win ratio’ which is the Number of games won/ Total games played, using the “select” query.

CODE

```
import turtle
import time
import pickle
import winsound
import mysql.connector as sql

t=turtle.Turtle()

def buttonsound():

winsound.PlaySound('Buttonlightsound.wav',winsound.SND_ASYNC)

def errormsg():
    turtle.setpos(-600,10)
    turtle.color('red')
```

```
    turtle.ht()
    turtle.speed(0)
    turtle.write('CHECK YOUR INPUT!', align="left", font=("Calibri",
20))
    winsound.PlaySound('Buttonsound.wav',winsound.SND_ASYNC)
    time.sleep(2)
    turtle.undo()



def inputval(l,m,val):  #l,m are messages and val are values
accepted
        while True:
                var=turtle.textinput(l,m)
                if var in val:
                        return var
                else:
                        errormsg()
```

#Screen setting

```
screen=turtle.Screen()
screen.setup(width=1350,height=700,startx=0,starty=0)
screen.title('Connect 4')


winsound.PlaySound('Song
(mp3cut.net).wav',winsound.SND_ASYNC)


ts=t.getscreen()
ts.bgcolor('black')
```

```
t.ht()
t.color('red')
t.speed(10)
t.pensize(5)

#Creating animation Connect 4
t.pu()
t.setpos(-250,200)
t.pd()
t.lt(180)
t.fd(10)
t.circle(50,90)
t.fd(30)
t.circle(50,90)
t.fd(10)

t.pu()
t.setpos(-225,150)
t.rt(90)
t.pd()
t.fd(30)
t.circle(50,180)
t.fd(30)
t.circle(50,180)

t.pu()
```

```
t.setpos(-100,70)
t.rt(180)
t.pd()
t.fd(130)
t.rt(150)
t.fd(150)
t.lt(150)
t.fd(130)

t.pu()
t.setpos(-25,70)
t.pd()
t.fd(130)
t.rt(150)
t.fd(150)
t.lt(150)
t.fd(130)

t.pu()
t.setpos(90,200)
t.lt(90)
t.pd()
t.fd(30)
t.lt(90)
t.fd(130)
t.lt(90)
t.fd(30)
```

```
t.pu()
t.setpos(60,115)
t.pd()
t.fd(30)

t.pu()
t.setpos(160,200)
t.rt(180)
t.pd()
t.fd(10)
t.circle(50,90)
t.fd(30)
t.circle(50,90)
t.fd(10)

t.pu()
t.setpos(250,200)
t.pd()
t.rt(180)
t.fd(60)
t.pu()
t.setpos(220,200)
t.pd()
t.lt(90)
t.fd(130)

t.color('blue')
```

```
t.pu()
t.setpos(0,65)
t.rt(30)
t.pd()
t.fd(86)
t.lt(120)
t.fd(86)
t.pu()
t.setpos(0,65)
t.rt(90)
t.pd()
t.fd(130)
t.pu()
t.lt(90)

time.sleep(2)
t.clear()

while True:


    while True:
        t.setpos(0,10)
        t.color('yellow')

        #Asking login
        t.write('''                    MULTIPLAYER
```

Please enter the Player names in the box. One name in one box.

NOTE: If you already have played, your score gets added to your account provided you enter

the same name. Or else, a new name is created for you.''', align="center", font=("Calibri", 20))

name1= turtle.textinput('PLAYER 1','Enter Name').upper()

#To reduce chances of duplication, all input characters are made in upper case

buttonsound()

name2= turtle.textinput('PLAYER 2','Enter Name').upper()

buttonsound()

t.clear()

#Rules being displayed

t.setpos(0,-70)

t.color('red')

t.write(''' RULES :

1. There are 8 columns in the grid. Each numbered 0 to 7 from the left.

2. To put a coin in a column, enter the column number.

3. Player 1 plays first and player 2 plays second.

4. If you get 4 in a row, diagonally, vertically or horizontally, then you win!

5. If you want to quit the current game at any time, enter 'q' in the column box.

SO ARE YOU READY TO PLAY? :)

To continue, press ENTER''', align='center', font=('Calibri',20))

```
x=turtle.textinput('To continue','Press enter')
buttonsound()

while True:
    valuesallowed=['0','1','2','3','4','5','6','7','q']
    t.clear()

    t.lt(90)



    t.rt(90)
    t.color('blue')
    t.setpos(-300,-300)

    #Drawing grid
    x=-300
    t.speed(0)
    for i in range(10):
        t.pd()
        t.fd(600)
```

```
        t.pu()
        t.setpos(-300,x)
        x+=75
t.pu()
t.lt(90)

x=-300
for i in range(10):
        t.pd()
        t.fd(600)
        t.pu()
        t.setpos(x,-300)
        x+=75

y=305
x=-262
for i in range(0,8):
        t.pu()
        t.setpos(x,y)
        t.pd()
        t.color('red')
        t.write(i,align='center',font=('Calibri',20))
         #Writing numbers on top of grid
        x+=75

#Creating the coordinate points in the grid
t.pu()
```

```
                pd={}
                #rows numbered down to up 0 to 7
(column,row)
                p=[(i,j) for i in range(8) for j in range(8)]


                coord=[-262,-185,-110,-35,35,110,185,262]


                #Making dictionary pd with match of each box
as none
                for i in p:
                        pd[i]='none'


                ctr=1
                while 1:
                        if ctr%2==0:
                                color='red'
                                name=name2
                        else:
                                color='green'
                                name=name1


                                m=inputval('Column
no',name,tuple(valuesallowed))
        #Asking what column for each member;
datatype=string
                                buttonsound()


                                if m=='q':
```

```
                    break

            c=int(m)  #c is the column no to be entered in integer

            rlist=[]
            n=len(p)
            for i in range(0,n):
                if p[i][0]==c:
                    rlist+=[p[i][1]]
                    break
#create list of all rows in p with column=c with rows having no coin in them. So, the minimum of #this gives the row where the new coin is to be input, ie the lowermost row.
            else:
                valuesallowed.remove(str(c)) # in case all the rows are filled
                errormsg()
                continue

            r=min(rlist)
            t.setpos(coord[c],coord[r])
            t.dot(50,color)

            p.remove((c,r))

            pd[(c,r)]=name
#in dictionary, this point is marked with name of player
```

```
                    t.setpos(350,100)

                    randomct=0

                    if 'none' not in pd.values():      #if all the
boxes are filled

                        t.setpos(430,40)
                        t.write('IT IS A
TIE!',align='center',font=('Calibri',20))
                        randomct+=1
                        winner='tie'
                        time.sleep(3)
                        break



                    for i in range(0,8):
#Finding if 4 in a row, column or diagonal have the same
name assigned to them – to find winner
                        for j in range(0,8):

                            if (i,j+1) in pd and (i,j+2) in pd and
(i,j+3) in pd and \
                            (i,j) in pd and \

    pd[(i,j+1)]==pd[(i,j+2)]==pd[(i,j+3)]==pd[(i,j)]!='none':
                            #checking if 4 in a column

                                winner=str(pd[(i,j)])
```

```
        t.write(winner,align='center',font=('Calibri',20))
                                        t.setpos(430,40)
                                        t.write('''          has won!
        Congratulations!''',align='center',font=('Calibri',20))
                                        randomct+=1
                                        time.sleep(3)
                                        break

                                    elif (i+1,j) in pd and (i+2,j) in pd and \
                                    (i+3,j) in pd and (i,j) in pd and\

        pd[(i+1,j)]==pd[(i+2,j)]==pd[(i+3,j)]==pd[(i,j)]!='none':
                                    #checking if 4 in a row
                                        winner=str(pd[(i,j)])

        t.write(winner,align='center',font=('Calibri',20))
                                        t.setpos(430,40)
                                        t.write('''                 has won!
        Congratulations!''',align='center',font=('Calibri',20))
                                        randomct+=1
                                        time.sleep(3)
                                        break

                                    elif (i+1,j+1) in pd and (i+2,j+2) in pd
and \
                                    (i+3,j+3) in pd and (i,j) in pd and\
```

```
        pd[(i+1,j+1)]==pd[(i+2,j+2)]==pd[(i+3,j+3)]==pd[(i,j)]!
='none':
                                #checking if 4 in a diagonal – left
down, right up
                                        winner=str(pd[(i,j)])

        t.write(winner,align='center',font=('Calibri',20))
                                        t.setpos(430,40)
                                        t.write('''         has won!
        Congratulations!''',align='center',font=('Calibri',20))
                                        randomct+=1
                                        time.sleep(3)
                                        break


                                elif (i,j) in pd and (i+1,j-1) in pd and \
                                (i+2,j-2) in pd and (i+3,j-3) in pd and\
                                pd[(i,j)]==pd[(i+1,j-1)]==pd[(i+2,j-
2)]==pd[(i+3,j-3)]!='none':
                                #checking if 4 in a diagonal – left
up, right down
                                        winner=str(pd[(i,j)])

        t.write(winner,align='center',font=('Calibri',20))
                                        t.setpos(430,40)
                                        t.write('''         has won!
        Congratulations!''',align='center',font=('Calibri',20))
                                        randomct+=1
                                        time.sleep(3)
                                        break
```

```
                    else:
                        continue

                if randomct==1:
                    break
            if randomct==1:
                break
            ctr+=1

        if m!='q':
```
#only if they havent quit the game, data about the winner is added/displayed
```
            if winner!='tie':
                #updating mysql table

    mycon=sql.connect(host='localhost',passwd='mysql',
user='root',database='class12')
                cursor=mycon.cursor()

                cursor.execute("select name from
connect4;")
                namestuple=cursor.fetchall()
                namesplayed=[]
                for i in namestuple:
                    s=''
                    for j in i:
                        s+=j
```

```
                    namesplayed+=[s.upper()]
                for i in name1,name2:
                    if i in namesplayed:
                        cursor.execute("update connect4
set totalgames=totalgames+1 where name='"+i+"';")
                    else:
                        cursor.execute("insert into
connect4 values('"+i+"',1,0,0);")

                if name1==winner:
                    cursor.execute("update connect4 set
won=won+1 where name='"+name1+"';")
                else:
                    cursor.execute("update connect4 set
won=won+1 where name='"+name2+"';")

                cursor.execute("select totalgames,won
from connect4 where name='"+name1+"';")
                data=cursor.fetchone()
                rname1=data[1]/data[0]
                cursor.execute("select totalgames,won
from connect4 where name='"+name2+"';")
                data=cursor.fetchone()
                rname2=data[1]/data[0]

                cursor.execute("update connect4 set
ratio="+str(rname1)+" where name='"+name1+"';")
                cursor.execute("update connect4 set
ratio="+str(rname2)+" where name='"+name2+"';")
```

```
                mycon.commit()
                mycon.close()

                #updating binary file
                f=open(r'C:\Users\csmaa\Desktop\CS Project\multiconnect4.dat','rb') #assuming it has already been created
                players={}
                try:
                    while True:
                        players.update(pickle.load(f))
                except:
                    f.close()
                for i in players:
                    if name1 in i and name2 in i:
                        score=list(players[i])

                        del players[i]

                        if i[0]==winner:
                            score[0]+=1
                        elif i[1]==winner:
                            score[1]+=1

                        players[i]=tuple(score)
                        break
```

```
                else: #In case dict is empty or if name not
found
                    if name1==winner:
                        players[(name1,name2)]=(1,0)
                    elif name2==winner:
                        players[(name1,name2)]=(0,1)
                #each time w is done so only one pickle
done everytime
                g=open(r'C:\Users\csmaa\Desktop\CS
Project\multiconnect4.dat','wb')
                pickle.dump(players,g)
                g.close()
            #Displaying score board for this game
from binary file
            t.clear()
            t.setpos(0,0)
            t.color('pink')
            f=open(r'C:\Users\csmaa\Desktop\CS Project\
multiconnect4.dat','rb') #assuming it has already been created
            players={}
            try:
                while True:
                    players.update(pickle.load(f))
            except:
                f.close()

            for i in players:
                if name1 in i and name2 in i:
```

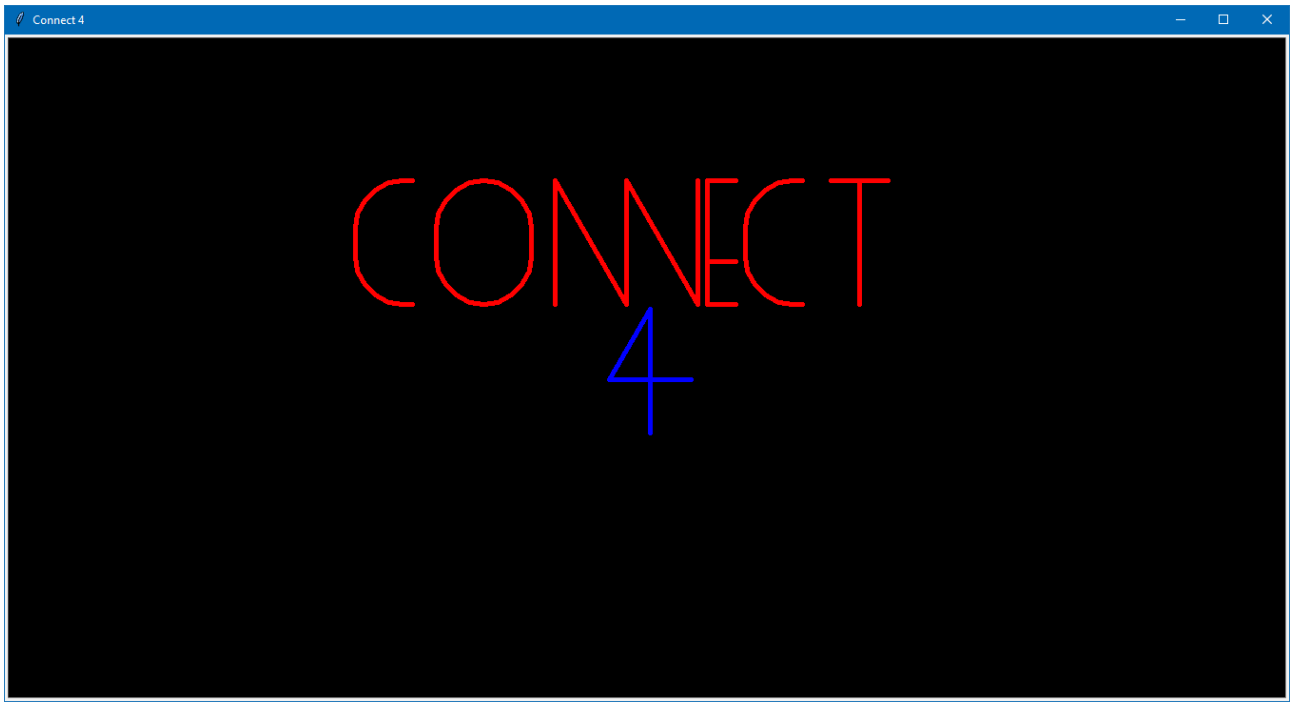
```
                        if i[0]==name1:
                            name1score=players[i][0]
                            name2score=players[i][1]
                        elif i[1]==name1:
                            name1score=players[i][1]
                            name2score=players[i][0]
                    #case of no entry cannot be present since addwinner() has already been done before this
                    #Output of SCORE BOARD
                    output='''SCORE BOARD\n
Total games:    '''+str(name1score+name2score)+'''
Won by '''+name1+''':    '''+str(name1score)+'''
Won by '''+name2+''':    '''+str(name2score)
                    t.write(output,align="center", font=("Calibri", 25))
                    time.sleep(5)
                #displaying leaderboard from mysql table

    mycon=sql.connect(host='localhost',passwd='mysql',user='root',database='class12')
                cursor=mycon.cursor()
                cursor.execute("select * from connect4 order by ratio desc,totalgames desc")
                alldata=cursor.fetchall()
                mycon.close()

                p1=str(alldata[0][0])
                p2=str(alldata[1][0])
```

```
                p3=str(alldata[2][0])
                r1=str(alldata[0][2])+'/'+str(alldata[0][1])
                r2=str(alldata[1][2])+'/'+str(alldata[1][1])
                r3=str(alldata[2][2])+'/'+str(alldata[2][1])

                t.clear()
                t.setpos(0,10)
                t.color('Orange')
                t.write('''LEADERBOARD\n
Name - Win Ratio\n
1 - '''+p1+'''  ('''+r1+''')
2 - '''+p2+'''  ('''+r2+''')
3 - '''+p3+'''  ('''+r3+''')''',align="center", font=("Calibri",20))
                time.sleep(7)
                #asking whether they want to play again with same person, different person or quit the game
                t.clear()
                t.setpos(0,10)
                t.color('red')
                t.write('''Type in the Textbox\n
1. Play again with the same person
2. Play with a different person
3. Quit''', align="center", font=("Calibri", 20))

                userchoice=inputval('Your Choice','1,2 or 3?',
('1','2','3'))
                buttonsound()
                t.clear()
```

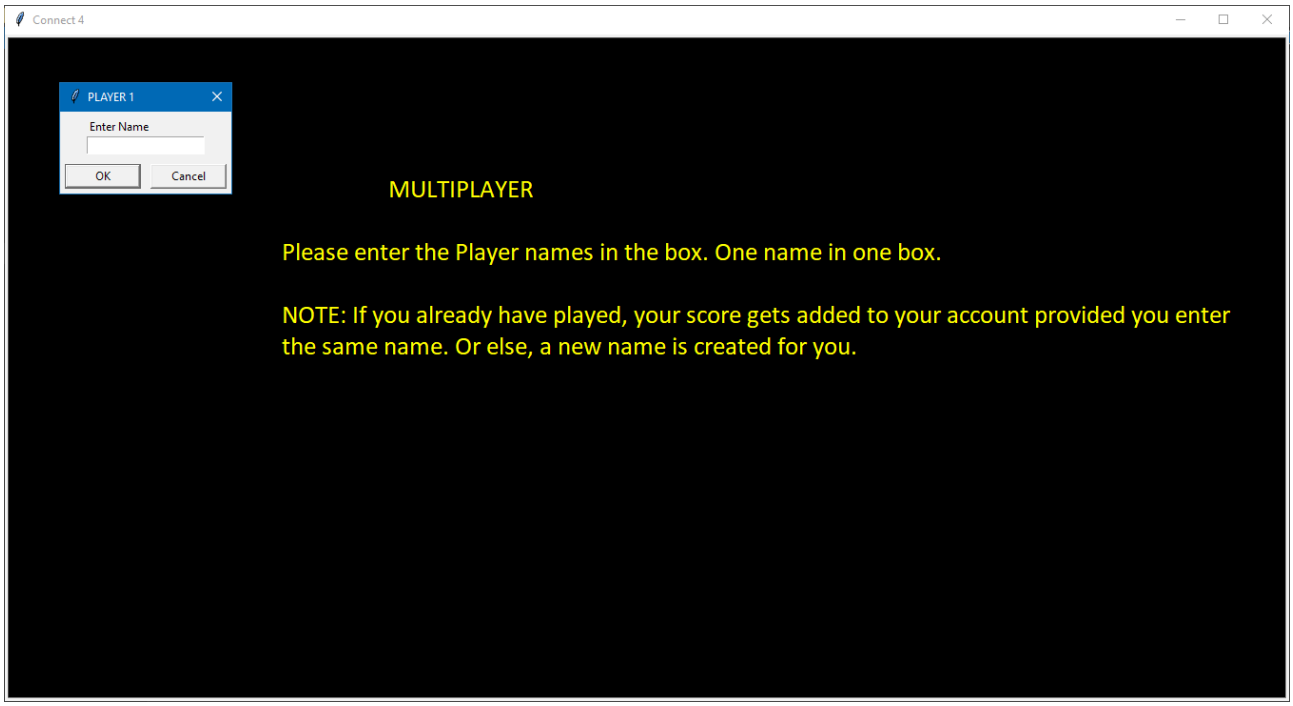
```
                t.rt(90) #to make grid back to normal alignment
                if userchoice=='2':
                    break
                elif userchoice=='3':
                    break
            if userchoice=='3':
                break
        if userchoice=='3':
            break
turtle.bye()
```

SCREENSHOTS OF PYTHON PLATFORM

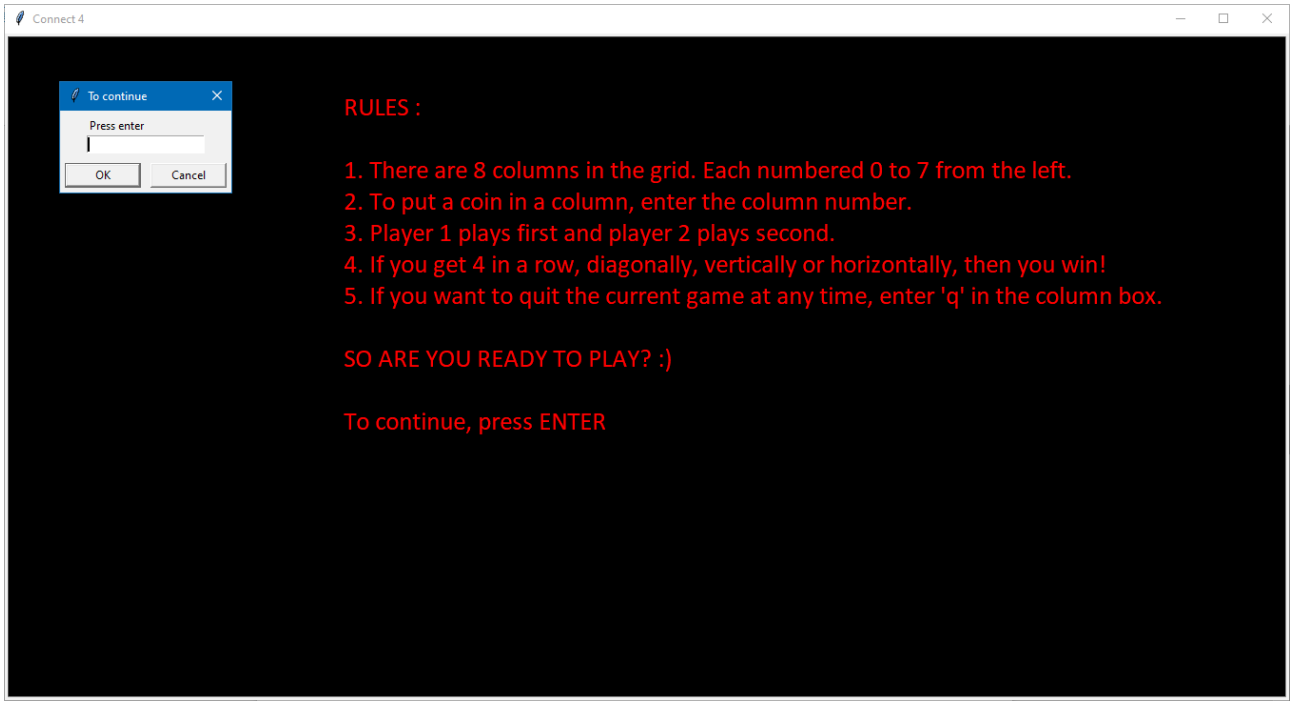
Initial Animation of Connect4



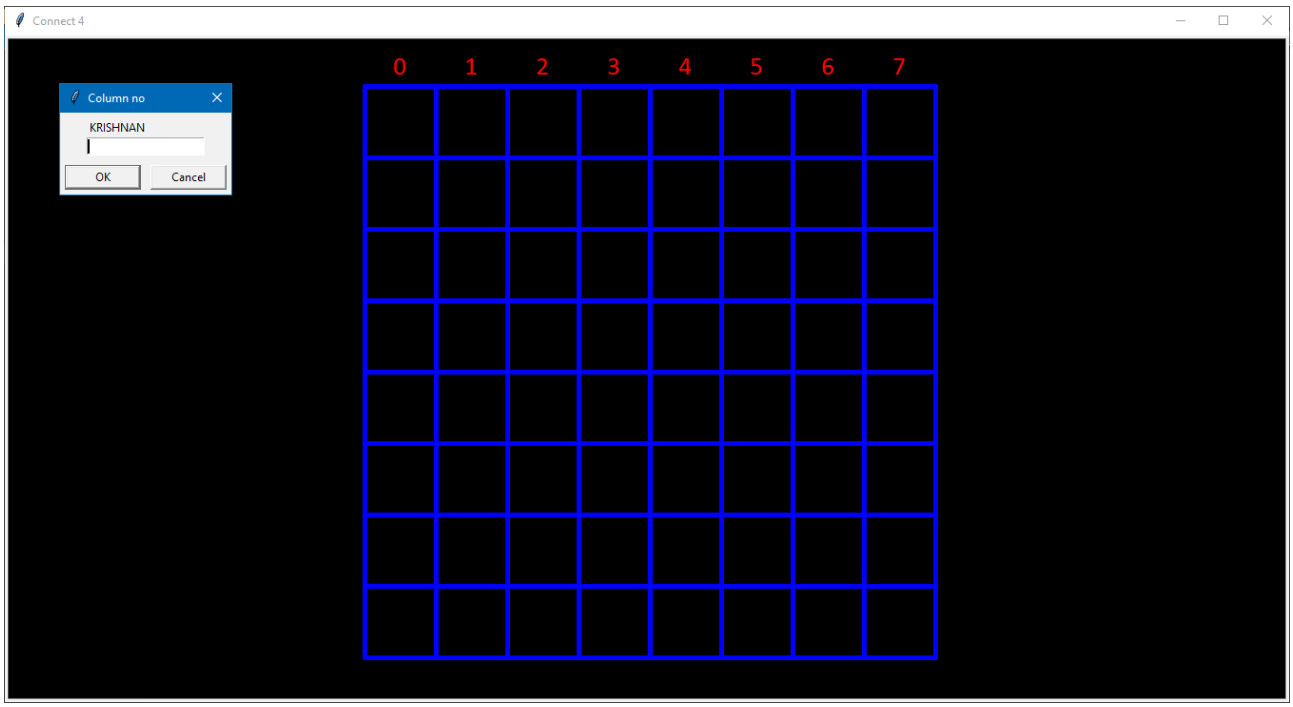
Players entering their names – first textbox displayed here. Once the name of player 1 is entered, the second textbox is shown.



Rules displayed

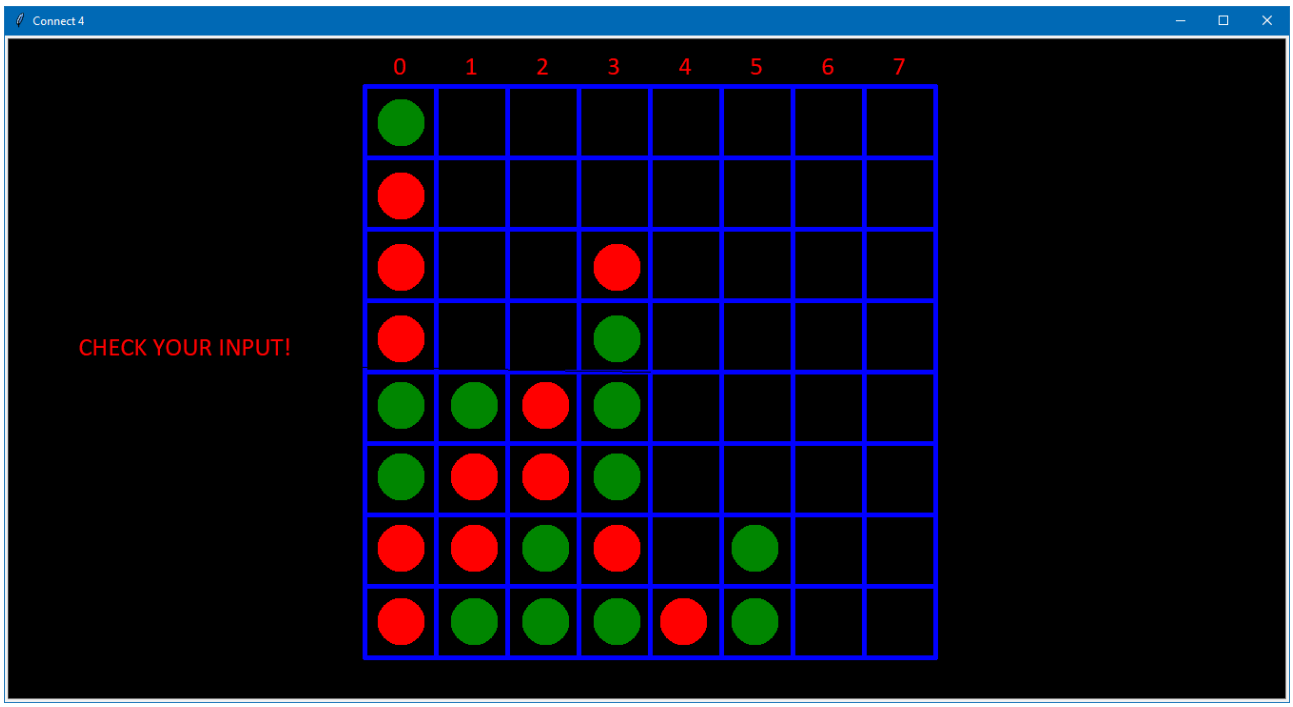


Grid displayed and asking the Player1 to enter the column number

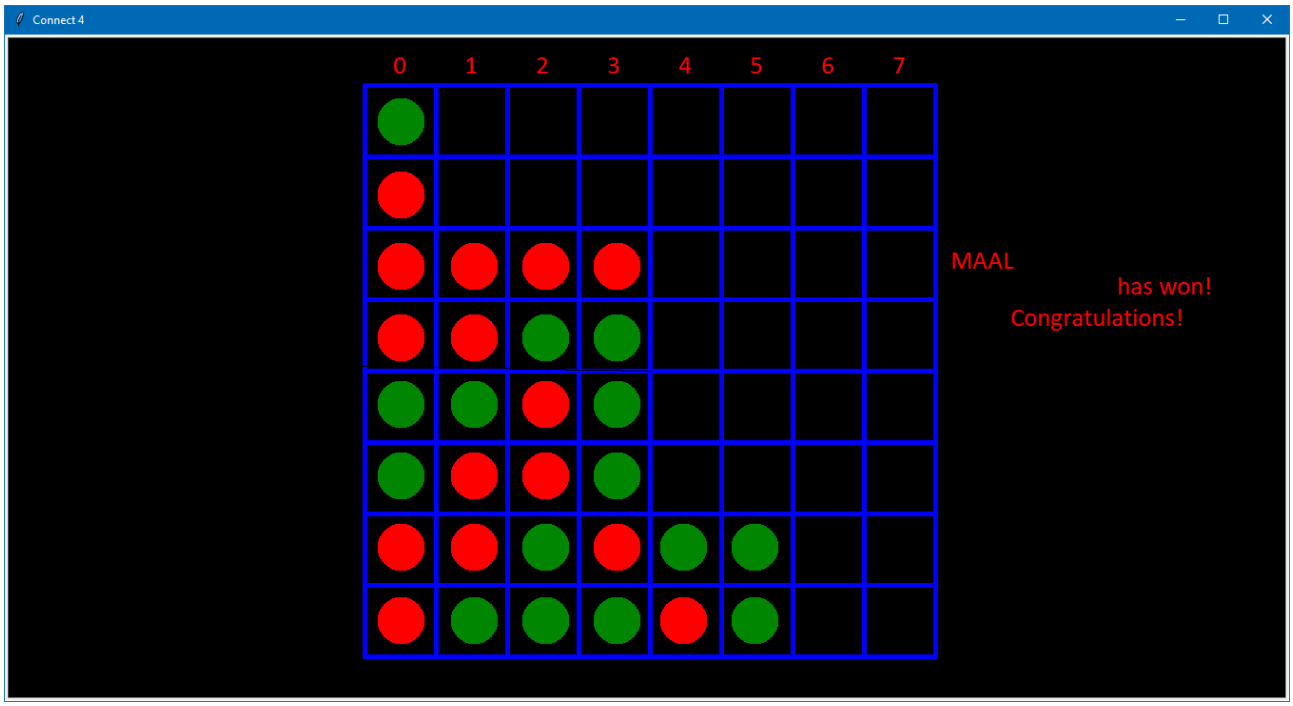


When a column number already full is input – Error message!

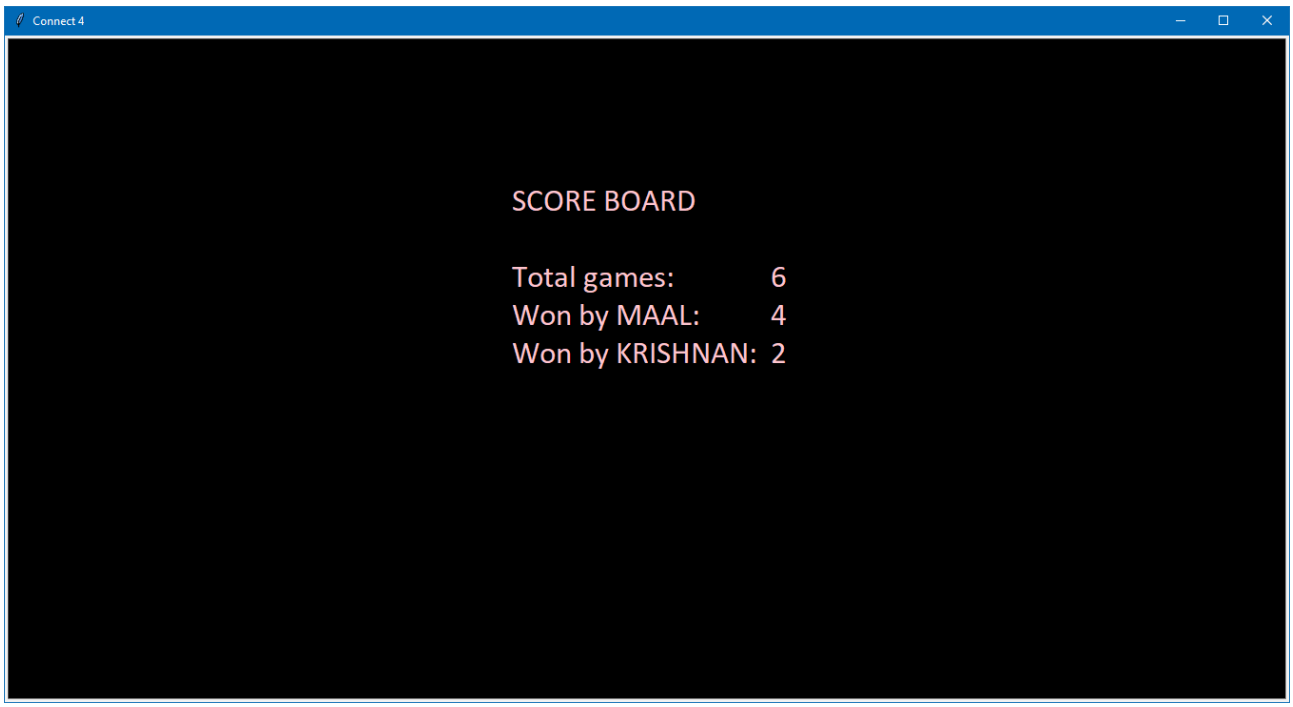
(Similar message shown when invalid inputs are made)



Player2 has won this game! 4 red coins in a row!

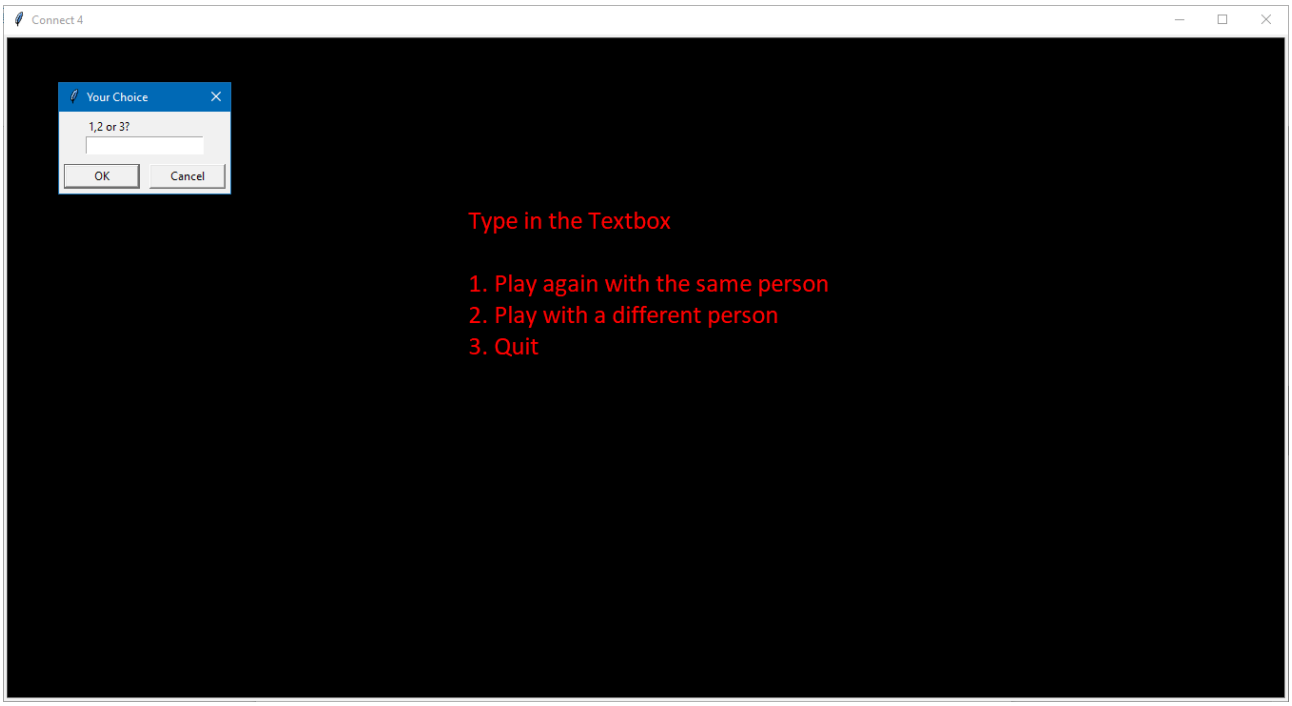


Scoreboard displayed



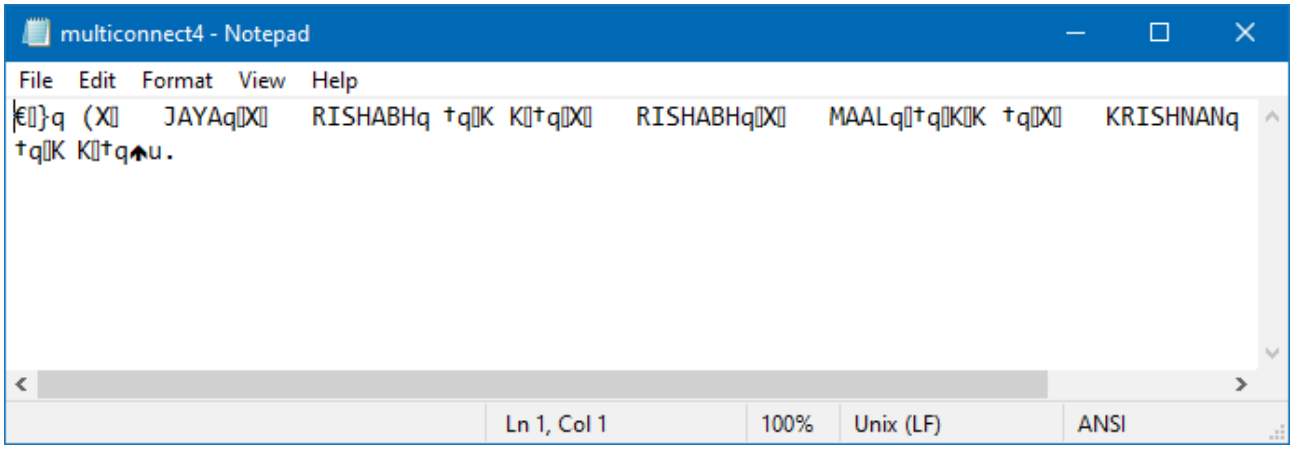
Leader board displayed

Final options displayed after each game

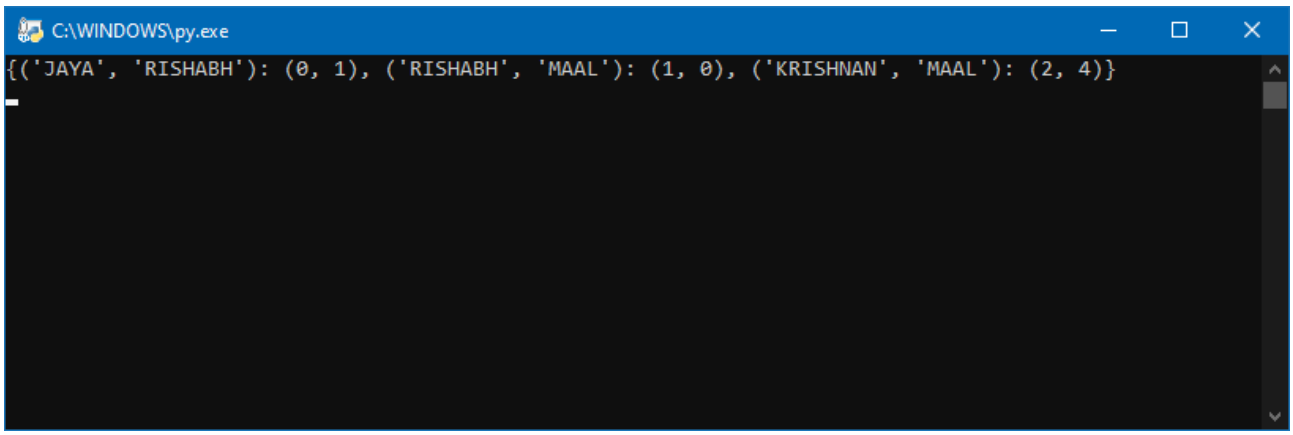


SCREENSHOTS OF FILE HANDLING

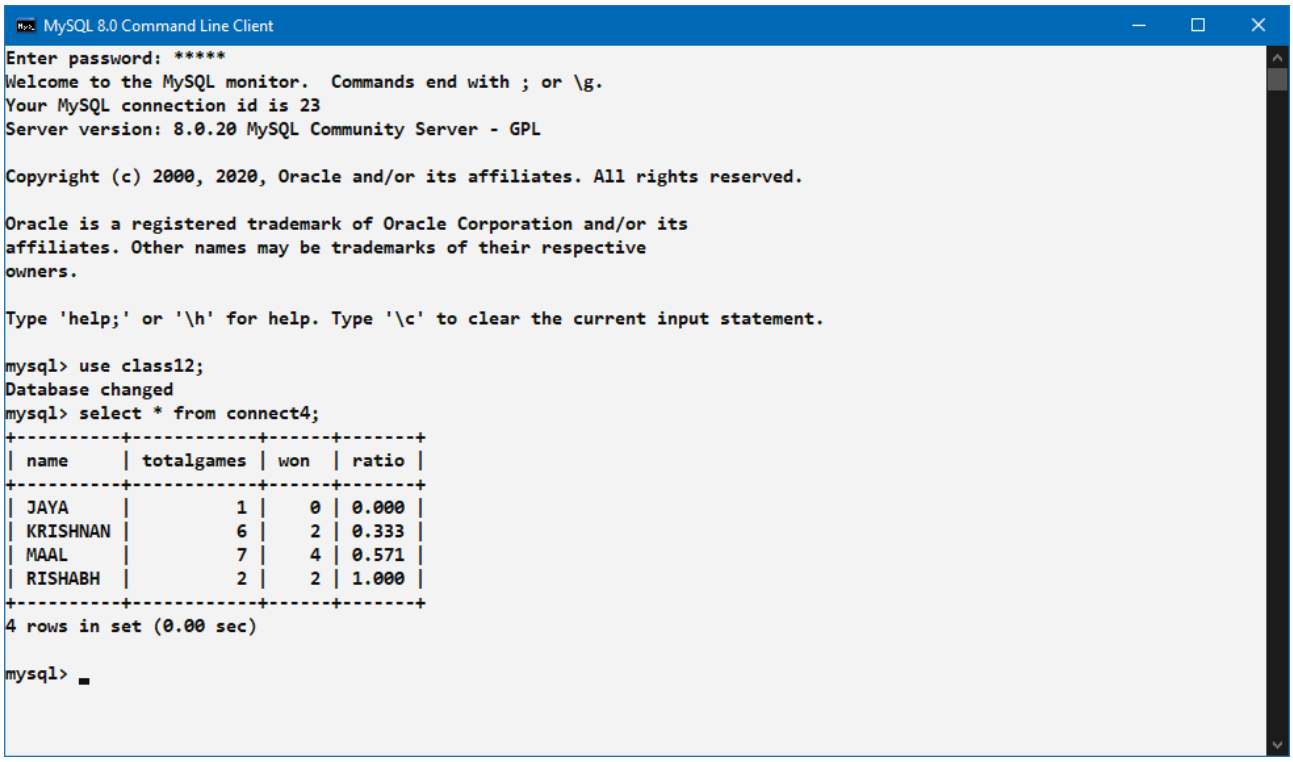
Actual binary file



Binary file in Human readable format



SCREENSHOT OF MYSQL



FUTURE IMPROVEMENTS

- **Single player Connect4**

  A game where one challenges the computer, instead of a fellow person.

- **Challenging friends on the internet**

  Making the game available on the internet and providing a method to play with friends remotely.

- **Providing a chatting panel along with the game**
  A way to play and catch up with your friends simultaneously.

BIBLIOGRAPHY

- **Learning about Python and its features:**
  Computer Science with Python (Textbook XII) By Sumita Arora

- **Learning more about Python's Turtle module:**
  https://docs.python.org/3/library/turtle.html#:~:text=Turtle%20graphics%20is%20a%20popular%20way%20for%20introducing%20programming%20to%20kids.&text=The%20turtle%20module%20is%20an,)%20100%25%20compatible%20with%20it.

- **Learning about Python's Winsound module:**

  https://docs.python.org/3/library/winsound.html#:~:text=The%20winsound%20module%20provides%20access,includes%20functions%20and%20several%20constants.&text=Call%20the%20underlying%20PlaySound(),%2Dlike%20object%2C%20or%20None%20.