DATA STRUCTURES AND ALGORITHMS (PROJECT COMPONENT)

Connect Four

Dishank Jhaveri (18BCE0991)

Abhishek Kumar Singh (18BCB0130)

Vaibhav Garg (18BCB0090)

Hrithik Dhoka (18BCE0994)

INDEX

ACKNOWLEDGEMENT

I would like to take this opportunity to express my gratitude towards all of those who have helped me complete the project entitled "**Connect Four**" in the limited time frame.

I first of all would like to thank my Data Structures and Algorithms faculty, **Sendhil Kumar** sir for allowing me to implement the game with my own rules in the python framework.

I have also been assisted by my friends and team partners, Vaibhav, Hrithik and Abhishek in the process of completing this project.

We have utilized minimum online resources in the development of this project. The modules used and format of gameplay are unique and thus completely developed by our team to the truest of our knowledge.

-Thank You

INTRODUCTION

In today's world of modern technology and entertainment, games have played an important role in our lives. Millions of games are played on different computers all over the world. Development of software and hardware especially for gaming is not something new. According to many studies, gaming helps in developing mental awareness and improving one's problem solving abilities. Games continue to become popular day by day and the necessity of games only kept increasing. There is an entire industry competing in the business of games. Many consoles like the Play station and Xbox are developed just for gaming purposes. New technologies like VR (virtual reality) are implemented in the recent games to attract gamers worldwide. All this tells us how important games have become to our present society as a form of entertainment and skill building.

Popular video gaming organizations

Python is a high-level programming language for general-purpose programming, created by Guido Van Rossum and first released in 1991. It is an object oriented programming language. It is an interpreted language and the coding design improves the code readability which attracted coders to code in this language. Python implements dynamic typing and provides for a large group of libraries and makes it more user friendly than most of the other high level languages. Python is a very fast growing language and is used in coding many important platforms. Large organizations that make use of python include Yahoo, Google, Wikipedia etc. The popular social networking site Reddit was entirely coded in python.

Connect Four is a two-player connection game where the players choose a color and put discs into 7 x 6 grid with the intention of connecting 4 discs of the same color in any direction. It was first sold in 1974 under the trademark of Connect Four.

Our project was an attempt to bring this popular board game into the digital world enabling millions of gamers all over the world to play this strategic game. The strategy involved in this game is to read all the moves possible and then placing the colored coin in such a way that it accounts for either building up a four connection or prevents the opponent from building one. The player who first connects four wins.

CONNECT FOUR:

This game as mentioned before involves the strategic game play of connecting four discs or coins of the same color in the grid. When the code is run, the user is first prompted with the tutorial of the game and some of the rules on the IDLE screen. The users are then expected to enter their name in alphabets only. Then a menu with options of different colors is displayed from which the users have to select their coin color and their board as well as their background color there by making it fully customized as per user's wish.

A turtle window then pops up with an empty board colored according to user specifications. The user is then expected to click on the row where he wishes the coin to be placed. The player whose turn is being played, is shown at the top of the screen. The game continues until four coins of a particular color is matched and then the winner, time elapsed and other such information is displayed on the IDLE screen. A prompt asking the user whether another game is to be played is displayed on the IDLE screen and according to the user input the program moves forward. In the end the winners of different games of the session is displayed along with their points in ascending order.

MODULES IMPORTED:

Turtle: The code involves the use of an important python module called turtle. This module helps us create an effective GUI for the interaction between the computer and the players during the course of the game. Turtle is part of the python’s standard library that comes along with the installation. Various functions of this module help us create an effective GUI thereby making the code interactive and user friendly.

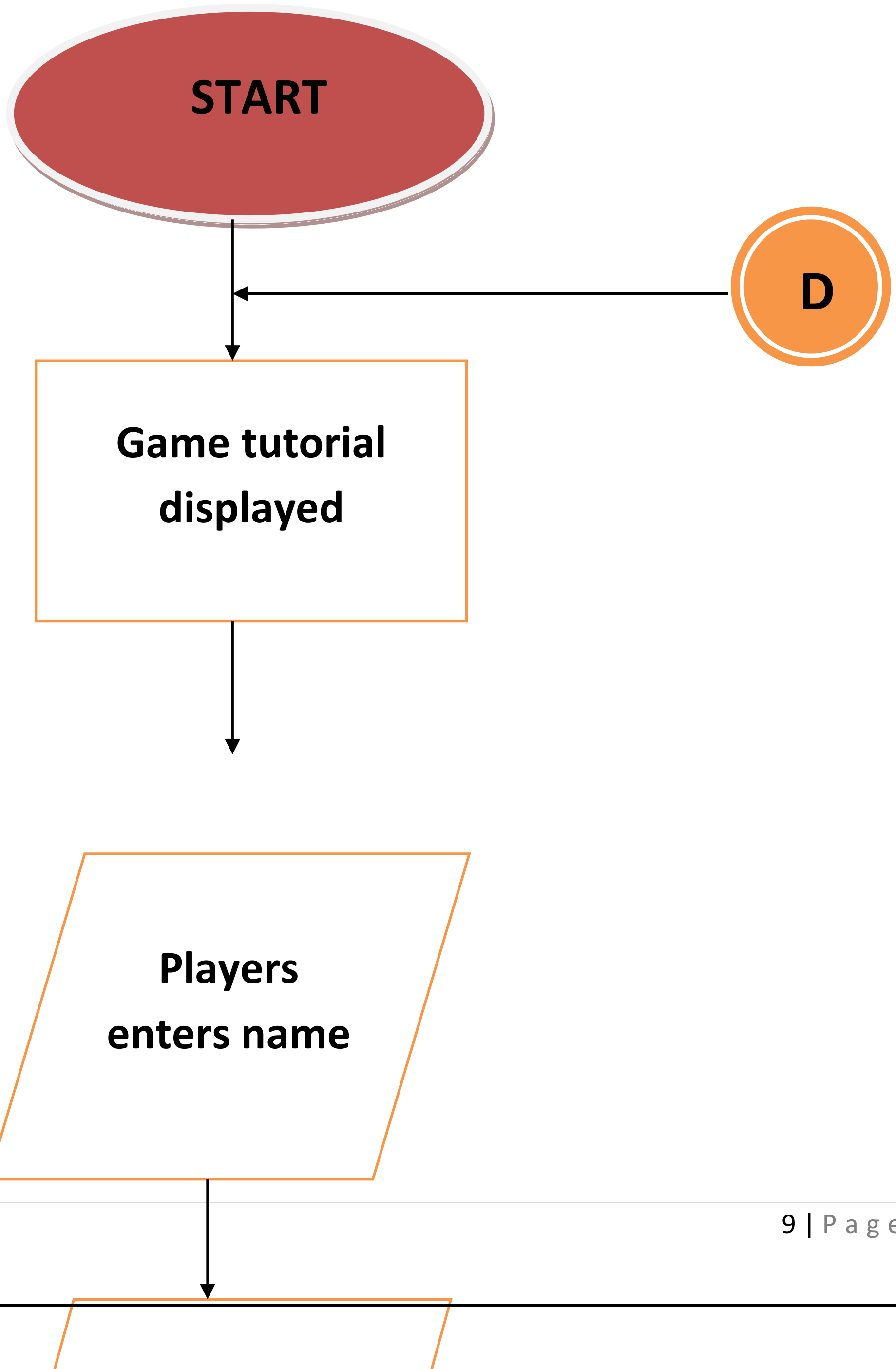
Time: Time module is another module which is imported during run time. It helps the code with getting the time of the system at that moment. It uses this time to find out the time elapsed during the course of the game and hence display this information to the user in the end. Time module is also used to delay the execution of the code for a while.

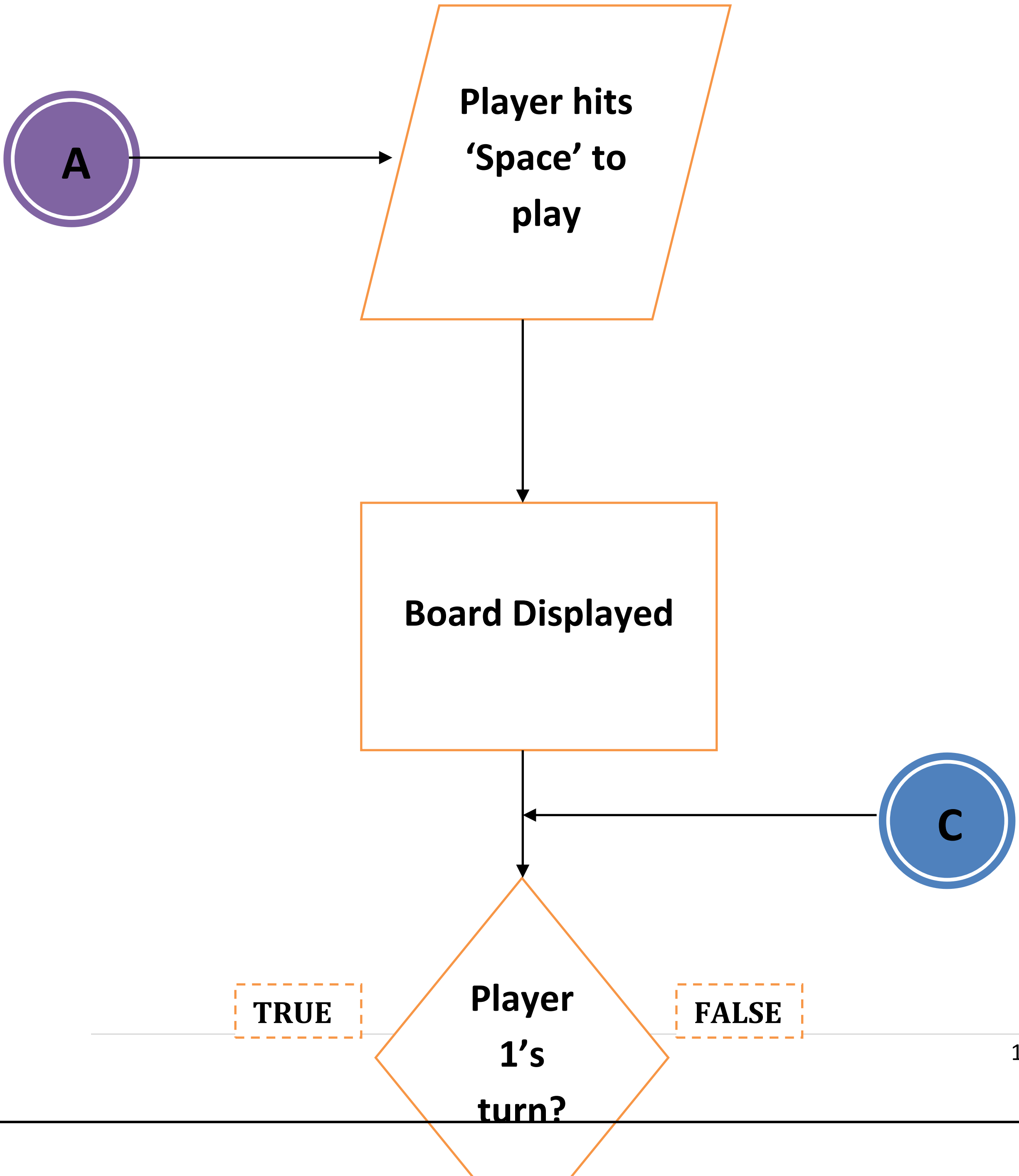
SYSTEM REQUIREMENTS

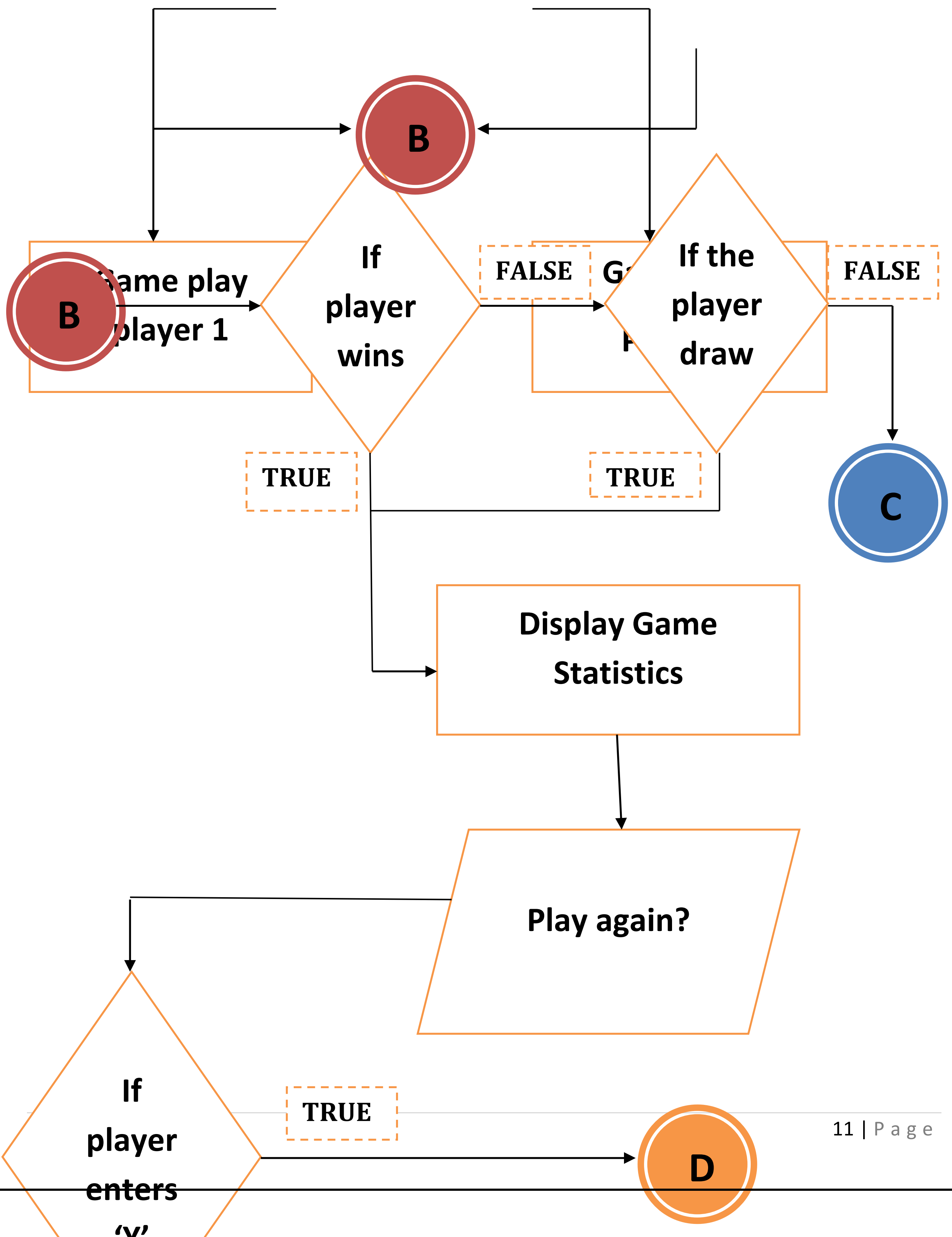
The system requirements for the project to work efficiently are:

- Python 2.7 interpreter and libraries.
- Connect four runner.
- About 20kb free space on hard disk
- Keyboard and other computer peripherals

FLOW CHART




A
Player hits ‘Space’ to play
Board Displayed
C
TRUE
Player 1’s turn?
FALSE


B
Game play
player 1
B
If
player
wins
FALSE
G
If the
player
draw
FALSE
TRUE
TRUE
C
Display Game
Statistics
Play again?
If
player
enters
'Y'
TRUE
D

FALSE

FUNCTION LIS

END

1. User defined

__init__()	Default class method
colorchoice()	Function to allow user to select their preffered colour.
tutorial()	To display the tutorial for the game.
Drawboard()	To draw the gameboard on the turtle screen.
Draw_c1()	To draw player 1's coin on the game board
Draw_c2()	To draw player 2's coin on gameboard.
Getwinner()	To check whether the game is won by either of the players and thus return the winner.
Declare()	To display the winner on turtle screen.
Place1()	To play player 1's desired position.

Place2()	To play player 2's desired position.
Stats()	To display the statistics of the game played.
Place()	To obtain position of the coin to be placed
Main()	To coordinate one game session.
Start()	To input user details and go to the game window.

2. Built in functions

Sort()	To sort the list in ascending order.
Str()	To convert the argument into string data type.
Turtle()	Function to create a turtle object.
ht()	To hide the turtle object from the screen.
Speed()	To define movement speed of turtle object.
Title()	To define title name to turtle window.
Setup()	To setup the charecterstics of

	the turtle window.
Bgcolor()	To define the background color.
Pencolor()	To define pen color.
Pu(),pd(),rt(),lt()	Turtle commands to move the turtle on the screen.
Onkey()	To check for user click.
Listen()	To listen to user input.
Mainloop()	To control and run the turtle window until it is destroyed.
Isalpha()	To check whether argument is string input.
Time()	To obtain system time.
Clear()	To clear contents from turtle screen.
Pensize()	To define the size of the turtle drawing.
Write()	To write text on the turtle screen.
Bye()	To destroy the turtle window.
Append()	To insert the argument into the list.
Int()	To convert the argument into integer data type.
Reset()	To reset the contents of the

	turtle screen to initial condition.
Sleep()	To stop the execution of the code for the given time period.
Fillcolor()	To fill color in the hollow drawing.
Begin_fill()	To start filling the color in the drawing .
End_fill()	To stop fi.lling color inside the hollow drawings.
Circle()	To draw a circle of the given radius on the screen.
Game()	Default object creating method.

SOURCE CODE

```
#START
from turtle import *
from time import *
class game:
    res = []
    def __init__(self):
        self.turn = 0
        self.board = [[None] * 6 for i in range(7)]
        self.count = [0] * 7
        self.colour = ['','','','']
#color1,color2,bgcolor,boardcolor
        self.name1 = ''
        self.name2 = ''
        self.winner = ''
        self.moves = 0
        self.timestart = 0
        self.timeend = 0
        self.points1 = 0
        self.points2 = 0
    def colorchoice(self):
        cl =
['black','gray','navy','red','green','yellow','maroon',
'white']
        i=0
        print '\n\t1.BLACK        2.GRAY       3.NAVY
4.RED'
        print   '\t5.GREEN        6.YELLOW     7.MAROON
8.WHITE'
        print '\nFROM THE ABOVE COLOR CODES:'
        while i!=4:
                l = [self.name1+', Enter your
coin',self.name2+', Enter your coin','Enter
background','Enter board']
                n = input(l[i]+' color code:')
                try:
                    n = int(n)
                    if 0<n<9 and (cl[n-1] not in
self.colour):
                        self.colour[i] = cl[n-1]
                        i+=1
                    else:
                        print 'MAKE SURE THE CODE IS
CORRECT AND IS UNUSED!\n'
                except:
                    print 'ENTER THE COLOUR CODE
CONSISTING OF NUMERIC VALUES ONLY.\n'

    def tutorial(self):
        print '\n',('*'*70)
        print '\nWELCOME TO OUR CONNECT FOUR GAME.WE ARE GLAD YOU CHOSE US.\n'
        print 'Connect 4 is a game involving a 7x6 grid where a winner is the player who matches four coins first:horizontally,vertically or diagonally.'
        print '\nHere are a few rules:'
        print '1.The grid consists of 7 columns and 6 rows.A coin gets placed at the bottom of each column considering there is no coin in that row.In case a coin is present in the column,the input coin is placed above the coin present.'
        print '2.Wait for your turn before placing the coin in order to avoid faulty placement of coins.'
        print '3.The looser of the game gets 10 points by default while the winner gets a minimum of 50 points which increases with lesser amoiunt of moves.'
        print '4.Please follow the instructions at every step.'
        print '\nHope you enjoy the game.Have fun.'
        print '*'*70
        print '\n'
        sleep(5)
    def drawboard(self):
        pencolor(self.colour[3])
        pu()
        setpos(-245,270)
        pd()
        rt(90)
        fd(540)
        lt(90)
```

```
fd(490)
lt(90)
fd(540)
lt(90)
fd(490)
pu()
setpos(-175,270)
pd()
lt(90)
fd(540)
pu()
setpos(-105,270)
pd()
fd(540)
pu()
setpos(-35,270)
pd()
fd(540)
pu()
setpos(35,270)
pd()
fd(540)
pu()
setpos(105,270)
pd()
fd(540)
pu()
setpos(175,270)
```

```
        pd()
        fd(540)
        pu()
        setpos(245,270)
        pd()
        fd(540)
        pu()
        setpos(-245,180)
        lt(90)
        pd()
        fd(490)
        pu()
        setpos(-245,90)
        pd()
        fd(490)
        pu()
        setpos(-245,0)
        pd()
        fd(490)
        pu()
        setpos(-245,-90)
        pd()
        fd(490)
        pu()
        setpos(-245,-180)
        pd()
        fd(490)
    def draw_c1(self):
        pencolor(self.colour[0])
        fillcolor(self.colour[0])
        begin_fill()
        circle(10)
        end_fill()
    def draw_c2(self):
        pencolor(self.colour[1])
        fillcolor(self.colour[1])
        begin_fill()
        circle(10)
        end_fill()
    def getwinner(self):
        # Check rows for winner
        for y in range(6):
            for x in range(4):
                if self.board[x][y]==self.board[x+1][y]==
                self.board[x+2][y]==self.board[x+3][y] and
                self.board[x][y]!=None:
                    return self.board[x][y]

        # Check columns for winner
        for x in range(7):
            for y in range(3):
                if self.board[x][y] == self.board[x][y+1]
                == self.board[x][y+2]==self.board[x][y+3]
                and self.board[x][y] !=None:
                    return self.board[x][y]
```

```
# Check diagonal (bottom-left to top-right) for
winner
for row in range(4):
   for col in range(3,6):
if (self.board[row][col] == self.board[row +
1][col - 1]  == self.board[row + 2][col - 2]
==\
self.board[row + 3][col - 3]) and
(self.board[row][col] != None):
     return self.board[row][col]



# Check diagonal (top-left to bottom-right) for
winner
for row in range(4):
   for col in range(3):
     if (self.board[row][col] == self.board[row
     + 1][col + 1] == self.board[row + 2][col +
     2] ==\
     self.board[row + 3][col + 3]) and
     (self.board[row][col] != None):
          return self.board[row][col]
l=1
for row in self.board:
     for i in row:
          if i==None:
               l=0
          if l:
               return 'D'
```

```
         # No winner: return the empty string
         return False
    def declare(self,code):
        sleep(10)
        self.timeend = time()
        ht()

        speed(100)
        pencolor(self.colour[3])
        pu()
        setpos(-200,100)
        pd()
        if code == 'X':
            write("THE GAME IS WON BY "+self.name1,font
= ('Arial',21,"bold"))
            self.winner = self.name1
            self.points2 = 10
            self.points1=50+(50-self.moves)
        elif code == 'Y':
            write("THE GAME IS WON BY "+self.name2,font
= ('Arial',21,"bold"))
            self.winner = self.name2
            self.points2 = 50+(50-self.moves)
            self.points1 = 10
        else:
            write("THE GAME IS DRAWN",font =
('Arial',21,"bold"))
            self.winner = 'Drawn'
            self.points2 = 10
            self.points1 = 10
        pu()
        setpos(-200,-200)
        pd()
        write("CLICK TO EXIT",font =
('Arial',18,"bold"))
        self.cont = False
    def place1(self,row):
        self.board[row][self.count[row]] = 'X'
        pu()
        setpos(-210+(row*70),-230+(self.count[row]*90))
        pd()
        self.draw_c1()
        self.count[row]+=1
        w = self.getwinner()
        pu()
        setpos(0,310)
        pd()
        self.draw_c2()
        if w == 'X':

            self.screen.reset()
            self.declare('X')
        elif w=='D':

            self.screen.reset()
            self.declare('D')
```

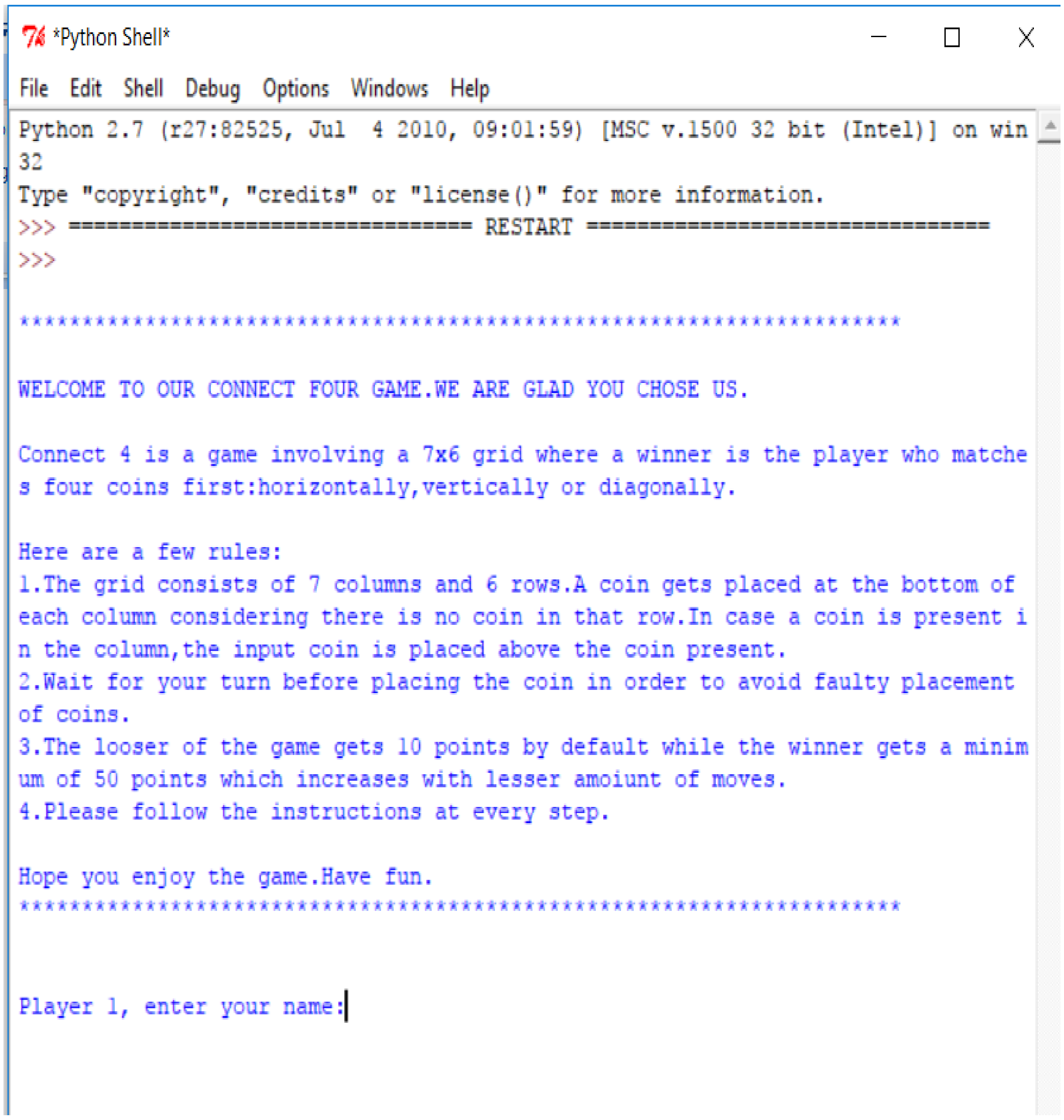
```
def place2(self,row):
    speed(1000)
    self.board[row][self.count[row]] = 'Y'
    pu()
    setpos(-210+(row*70),-230+(self.count[row]*90))
    pd()
    self.draw_c2()
    self.count[row]+=1
    w = self.getwinner()
    pu()
    setpos(0,310)
    pd()
    self.draw_c1()
    if w == 'Y':

        self.screen.reset()
        self.declare('Y')
    elif w=='D':

        self.screen.reset()
        self.declare('D')
def stats(self):
    print '*'*70
    print
    if self.winner!='Drawn':
        print 'Winner:',self.winner
        print 'Moves played entirely:',self.moves
```

```
            print 'Points scored
by',self.name1,':',self.points1
            print 'Points scored
by',self.name2,':',self.points2
            print 'Time elapsed:',int(self.timeend-
self.timestart),'secs'
        else:
            print 'MATCH IS DRAWN.Well played both of
you!'
            print 'Time elapsed:'
        print '*'*70
        print

game.res.append([max(self.points1,self.points2),self.wi
nner])
    def place(self,x,y):
        row = 0
        if self.cont == False:
            bye()
            self.stats()
        elif 270>y>-270:
            if -240<x<-180:
                row =1
            elif -170<x<-110:
                row=2
            elif -100<x<-40:
                row =3
            elif -30<x<30:
                row =4
            elif 40<x<100:
                row=5
            elif 110<x<170:
                row=6
            elif 180<x<240:
                row=7
            if row!=0 and self.count[row-1]<6:
                self.moves+=1
                if self.turn == 0:
                    self.place1(row-1)
                    self.turn = 1
                else:
                    self.place2(row-1)
                    self.turn=0
    def main(self):
        self.timestart = time()
        self.screen.clear()
        bgcolor(self.colour[2])
        pensize('5')
        ht()
        speed(200)
        self.drawboard()
        self.cont = True
        pu()
        setpos(-100,310)
        pd()
        write('TURN:\t',font=('Arial',17,'bold'))
        pu()
        setpos(0,310)
        pd()
        self.draw_c1()
        pu()
        self.screen.onclick(self.place)

    def choice(self):
        self.tutorial()
        i=0
        l = [None,None]
        while i<2:
            l[i] = raw_input('Player '+str(i+1)+',
enter your name:')
            if l[i].isalpha():
                i+=1
            else:
                print 'Make sure you only use alphabets
while entering your name!\n'
        self.name1 = l[0]
        self.name2 = l[1]
        self.colorchoice()
        t = Turtle()
        ht()
        t.ht()
        t.speed(200)
        speed(200)
        setup(height= 700, width = 700)
        self.screen = Screen()
        title('Connect-4')
        bgcolor('black')
        pencolor('red')
        pu()
        setpos(-175,0)
        pd()
        write('CLICK \'SPACE\' TO
PLAY..',font=('Arial',17,'bold'))
        self.screen.onkey(self.main,'space')
        self.screen.listen()
        mainloop()
count = 0
while True:
    obj = game()
    obj.choice()
    cho = raw_input('Enter \'Y\' to play again:')
    if cho != 'Y':
        break
print '*'*70
print 'Session stats:\n'
game.res.sort()
for i in range(len(game.res)):
    if game.res[i][1]!= 'Drawn':
        print str(i+1)+'.',game.res[i][1]+':
',game.res[i][0]
print '\nTHANK YOU FOR PLAYING WITH US.HOPE YOU ENJOYED
:)!'
```

#END

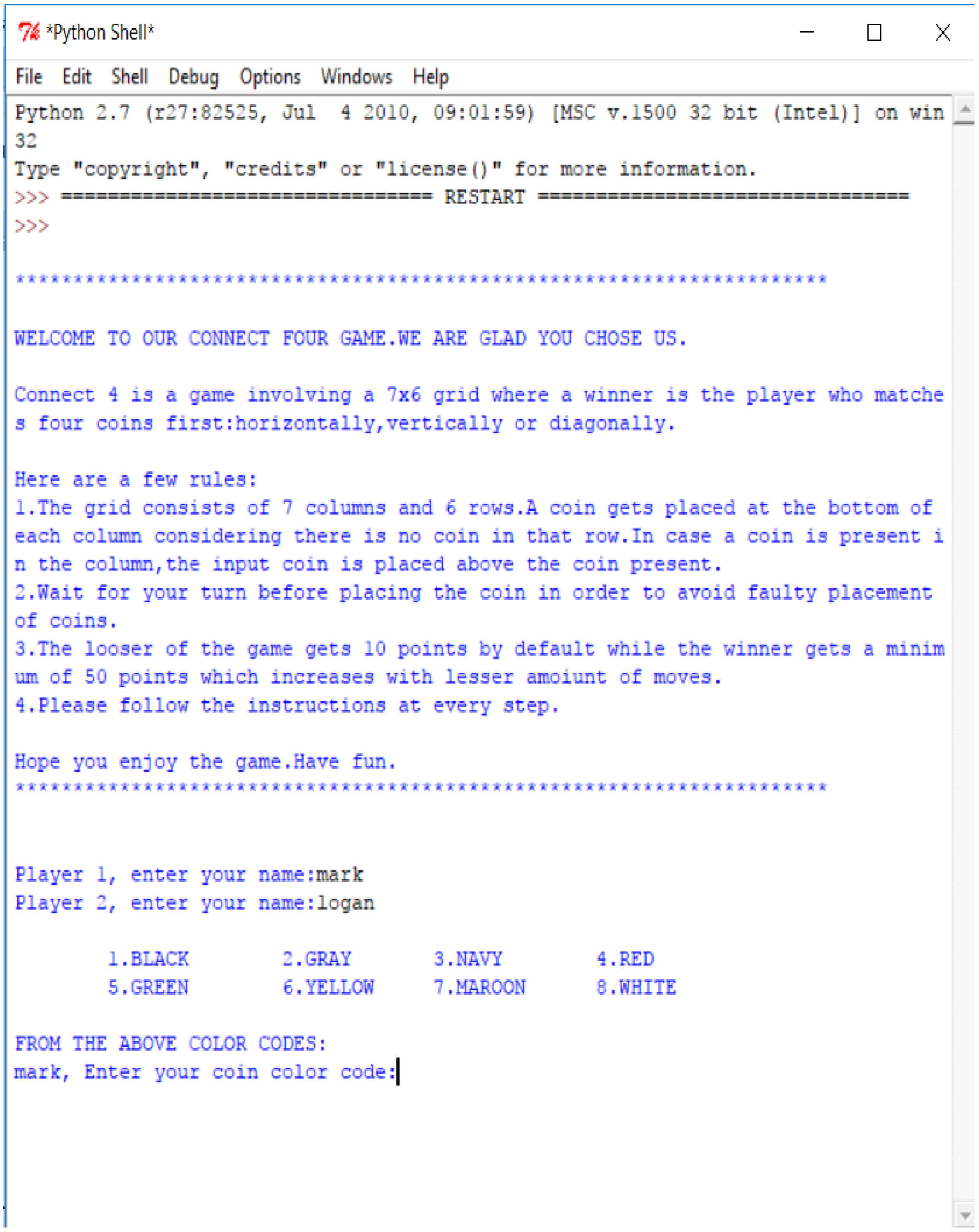
SAMPLE INPUT AND OUTPUT

1. **Starting page**



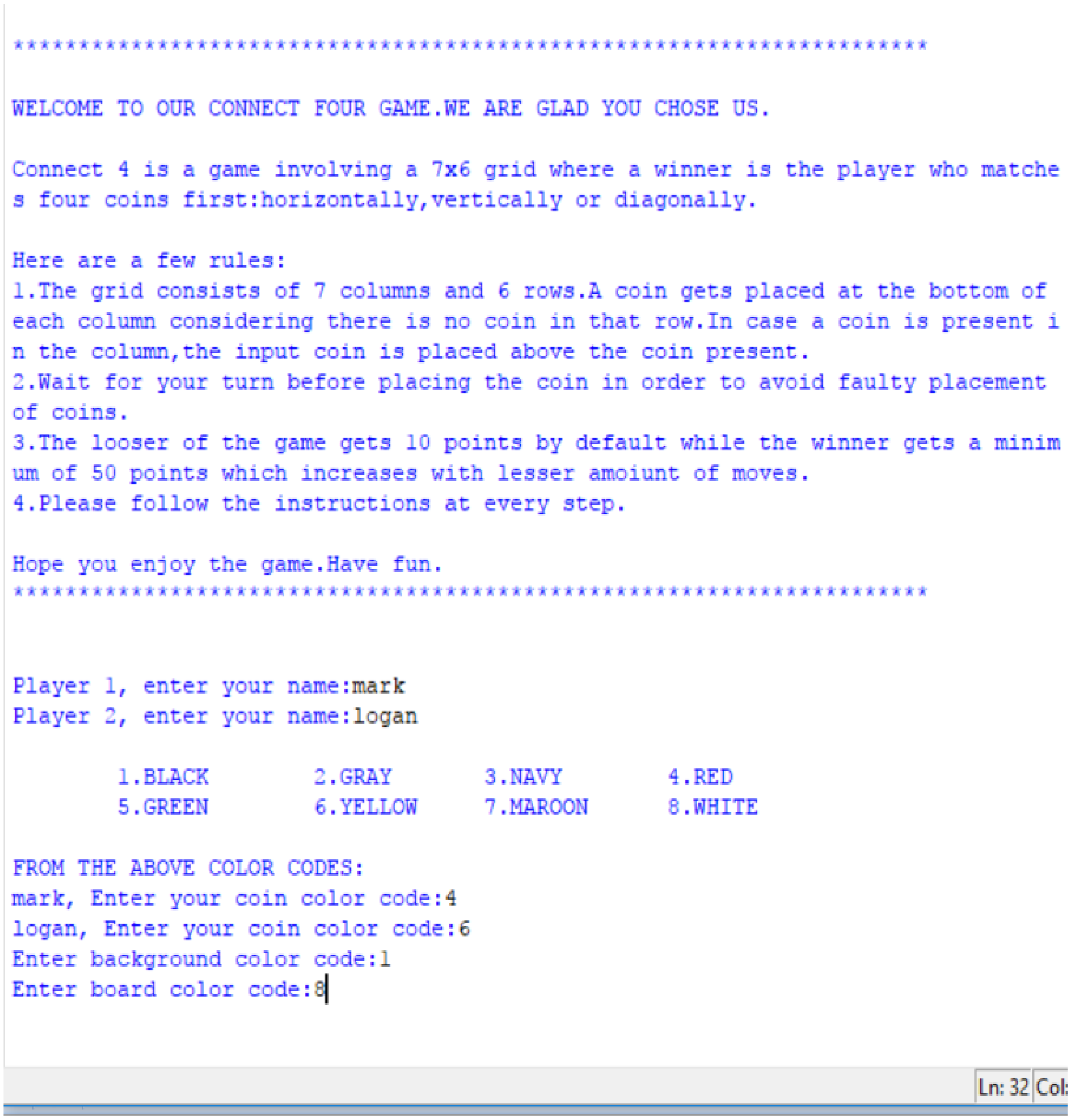
```
Python 2.7 (r27:82525, Jul  4 2010, 09:01:59) [MSC v.1500 32 bit (Intel)] on win
32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>

**********************************************************************

WELCOME TO OUR CONNECT FOUR GAME.WE ARE GLAD YOU CHOSE US.

Connect 4 is a game involving a 7x6 grid where a winner is the player who matche
s four coins first:horizontally,vertically or diagonally.

Here are a few rules:
1.The grid consists of 7 columns and 6 rows.A coin gets placed at the bottom of
each column considering there is no coin in that row.In case a coin is present i
n the column,the input coin is placed above the coin present.
2.Wait for your turn before placing the coin in order to avoid faulty placement
of coins.
3.The looser of the game gets 10 points by default while the winner gets a minim
um of 50 points which increases with lesser amoiunt of moves.
4.Please follow the instructions at every step.

Hope you enjoy the game.Have fun.
**********************************************************************


Player 1, enter your name:
```

2. Enter details



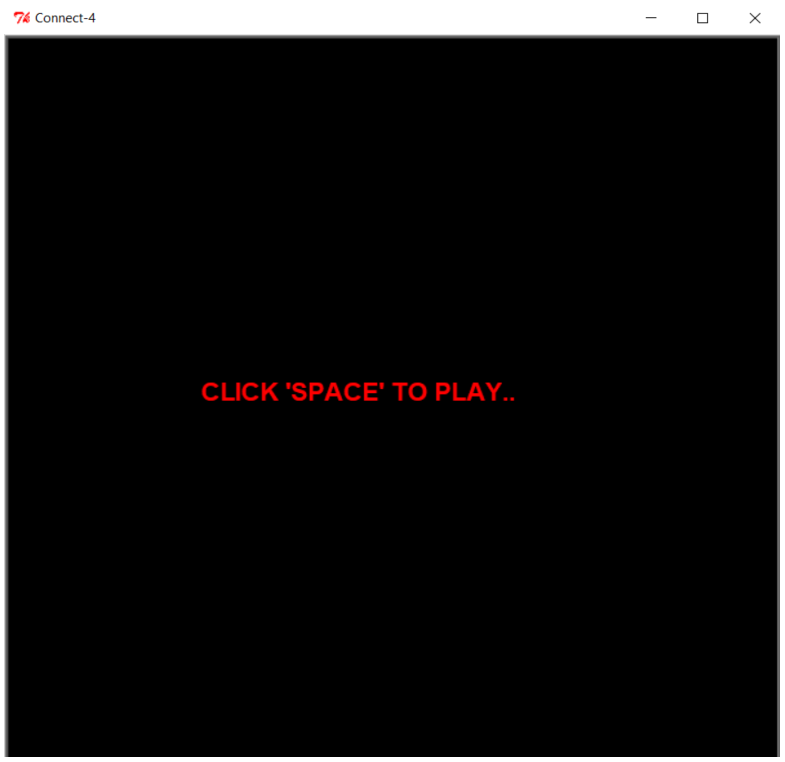
```
Python 2.7 (r27:82525, Jul  4 2010, 09:01:59) [MSC v.1500 32 bit (Intel)] on win
32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>

**********************************************************************

WELCOME TO OUR CONNECT FOUR GAME.WE ARE GLAD YOU CHOSE US.

Connect 4 is a game involving a 7x6 grid where a winner is the player who matche
s four coins first:horizontally,vertically or diagonally.

Here are a few rules:
1.The grid consists of 7 columns and 6 rows.A coin gets placed at the bottom of
each column considering there is no coin in that row.In case a coin is present i
n the column,the input coin is placed above the coin present.
2.Wait for your turn before placing the coin in order to avoid faulty placement
of coins.
3.The looser of the game gets 10 points by default while the winner gets a minim
um of 50 points which increases with lesser amoiunt of moves.
4.Please follow the instructions at every step.

Hope you enjoy the game.Have fun.
**********************************************************************


Player 1, enter your name:mark
Player 2, enter your name:logan

        1.BLACK        2.GRAY       3.NAVY        4.RED
        5.GREEN        6.YELLOW     7.MAROON      8.WHITE

FROM THE ABOVE COLOR CODES:
mark, Enter your coin color code:
```

3. Enter game specifications



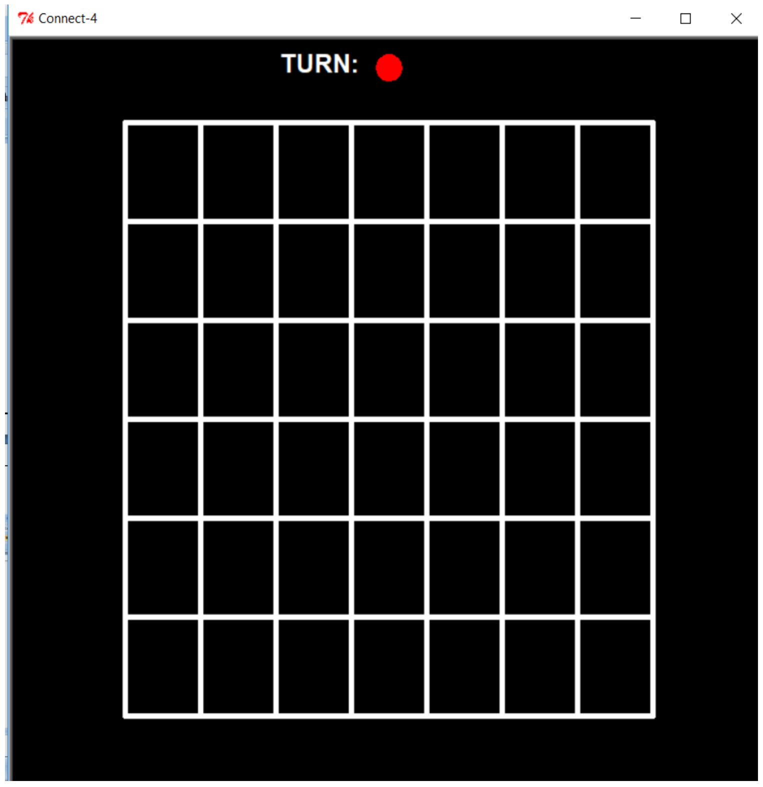
```
**********************************************************************

WELCOME TO OUR CONNECT FOUR GAME.WE ARE GLAD YOU CHOSE US.

Connect 4 is a game involving a 7x6 grid where a winner is the player who matche
s four coins first:horizontally,vertically or diagonally.

Here are a few rules:
1.The grid consists of 7 columns and 6 rows.A coin gets placed at the bottom of
each column considering there is no coin in that row.In case a coin is present i
n the column,the input coin is placed above the coin present.
2.Wait for your turn before placing the coin in order to avoid faulty placement
of coins.
3.The looser of the game gets 10 points by default while the winner gets a minim
um of 50 points which increases with lesser amoiunt of moves.
4.Please follow the instructions at every step.

Hope you enjoy the game.Have fun.
**********************************************************************



Player 1, enter your name:mark
Player 2, enter your name:logan

        1.BLACK        2.GRAY       3.NAVY       4.RED
        5.GREEN        6.YELLOW     7.MAROON     8.WHITE

FROM THE ABOVE COLOR CODES:
mark, Enter your coin color code:4
logan, Enter your coin color code:6
Enter background color code:1
Enter board color code:8
```

4. Game startup window

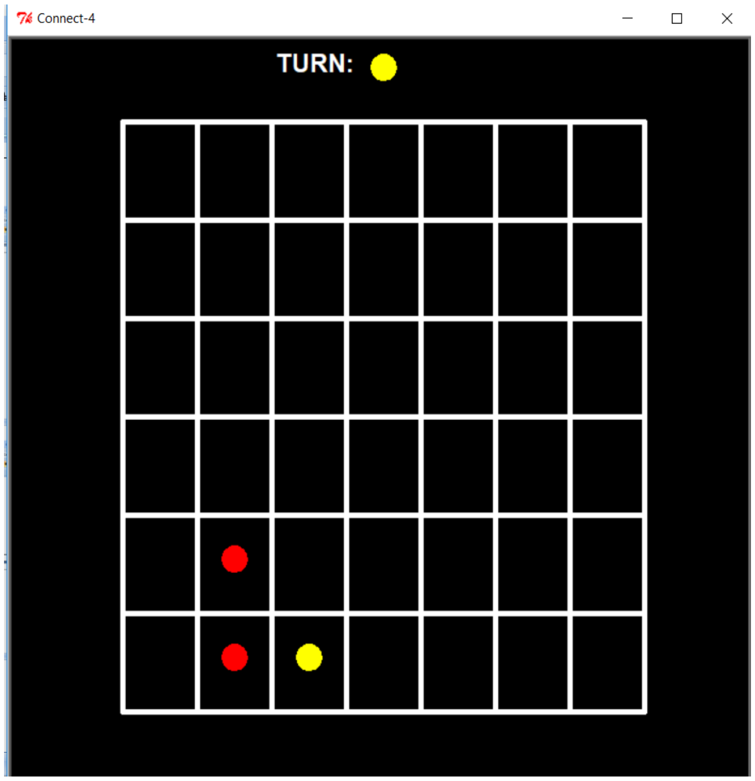


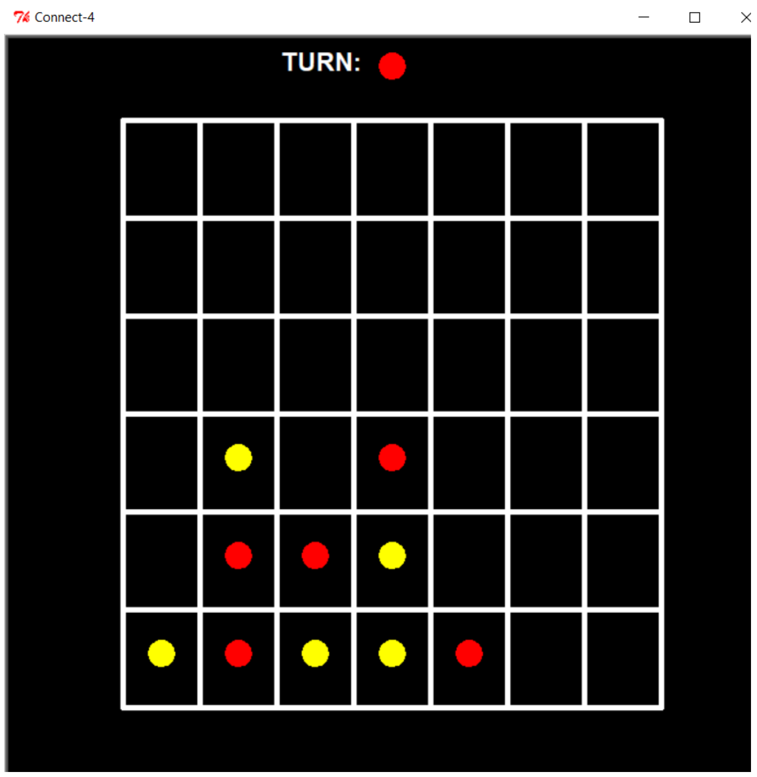
CLICK 'SPACE' TO PLAY..

5. Initial game setup

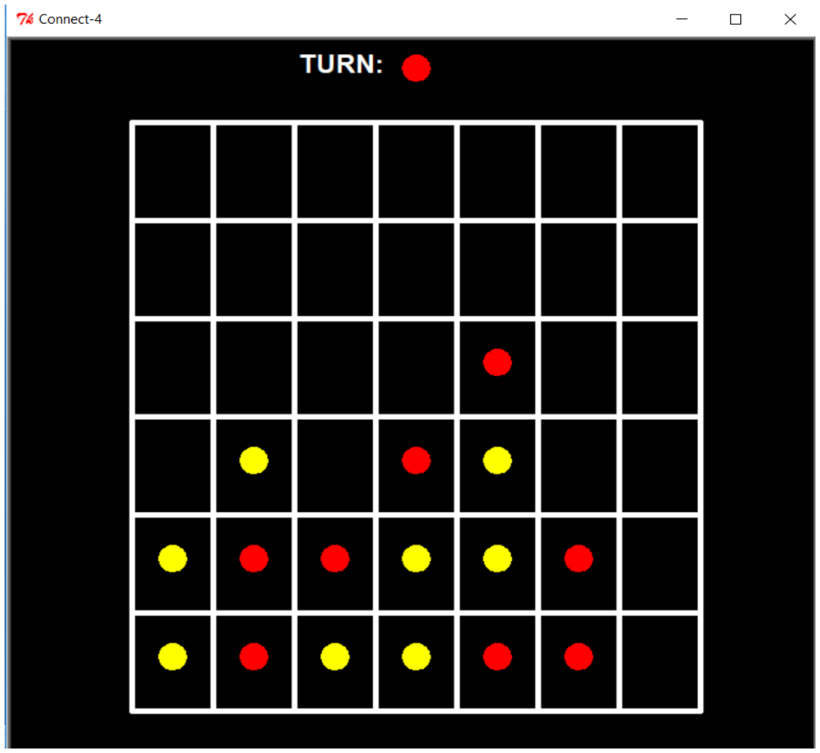


6. Game play:

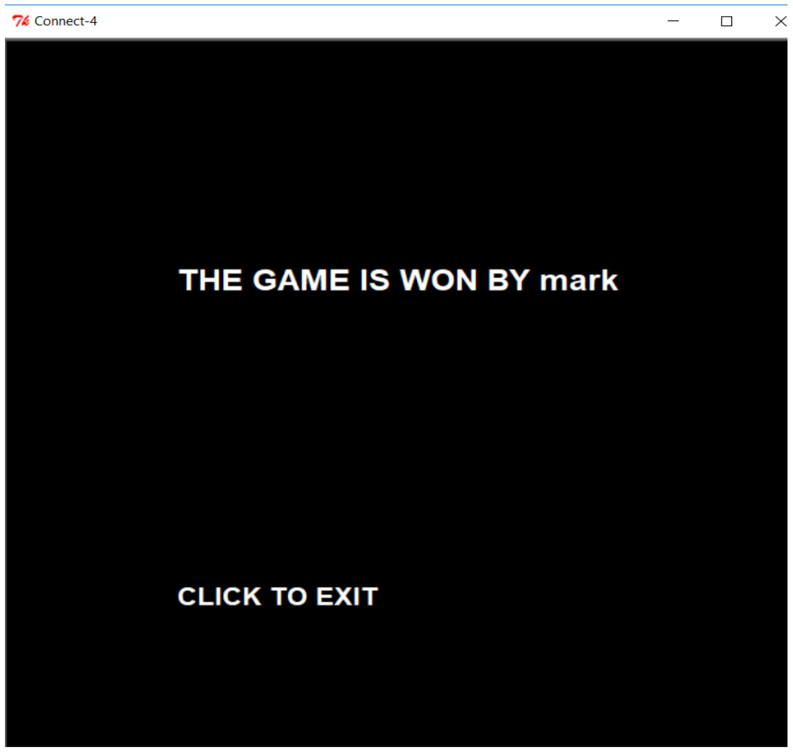




7. Winning move



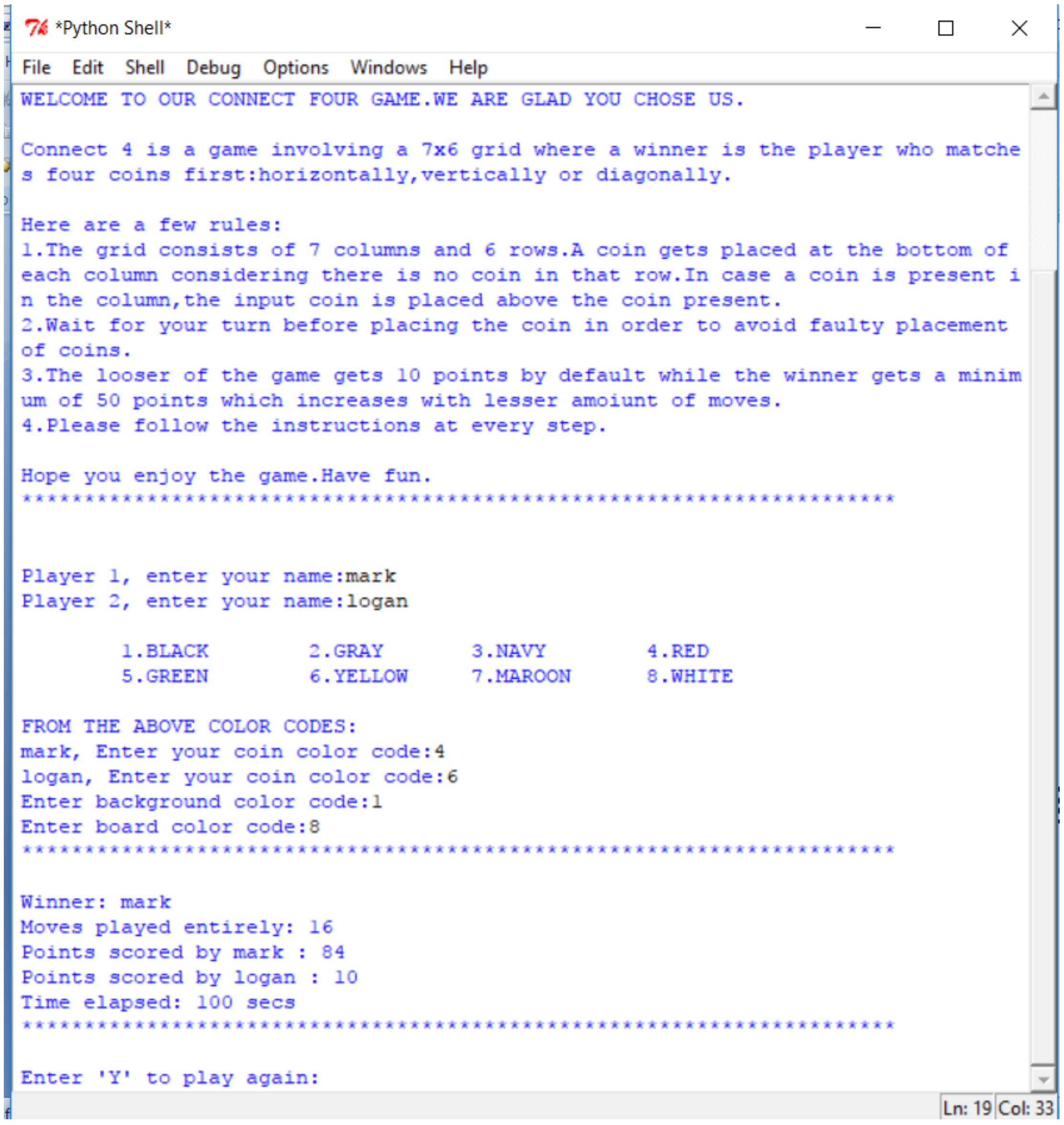
8. Winner display



THE GAME IS WON BY mark

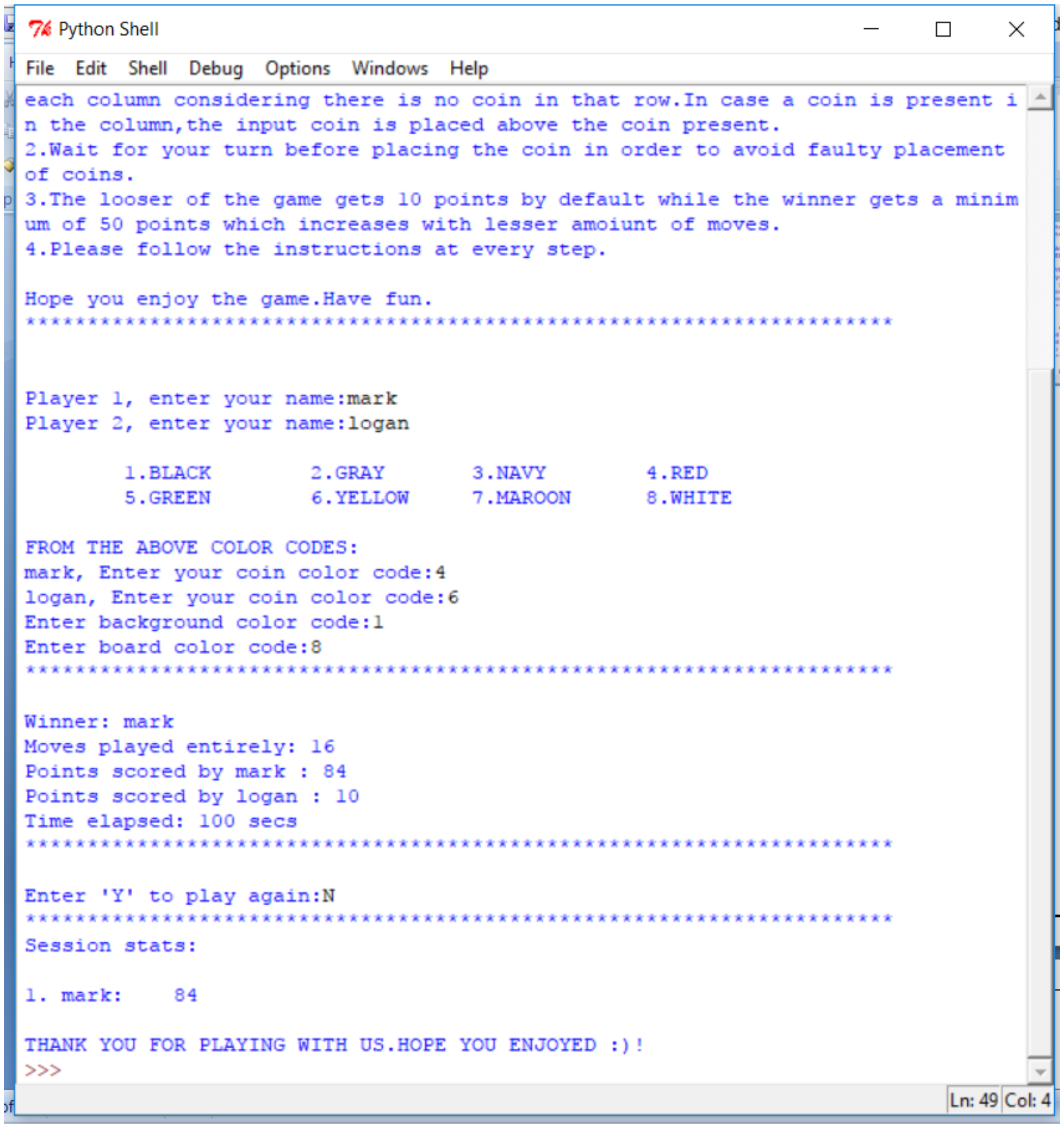
CLICK TO EXIT

9. Game statistics display



```
WELCOME TO OUR CONNECT FOUR GAME.WE ARE GLAD YOU CHOSE US.

Connect 4 is a game involving a 7x6 grid where a winner is the player who matche
s four coins first:horizontally,vertically or diagonally.

Here are a few rules:
1.The grid consists of 7 columns and 6 rows.A coin gets placed at the bottom of
each column considering there is no coin in that row.In case a coin is present i
n the column,the input coin is placed above the coin present.
2.Wait for your turn before placing the coin in order to avoid faulty placement
of coins.
3.The looser of the game gets 10 points by default while the winner gets a minim
um of 50 points which increases with lesser amoiunt of moves.
4.Please follow the instructions at every step.

Hope you enjoy the game.Have fun.
**********************************************************************


Player 1, enter your name:mark
Player 2, enter your name:logan

        1.BLACK        2.GRAY       3.NAVY        4.RED
        5.GREEN        6.YELLOW     7.MAROON      8.WHITE

FROM THE ABOVE COLOR CODES:
mark, Enter your coin color code:4
logan, Enter your coin color code:6
Enter background color code:1
Enter board color code:8
**********************************************************************

Winner: mark
Moves played entirely: 16
Points scored by mark : 84
Points scored by logan : 10
Time elapsed: 100 secs
**********************************************************************

Enter 'Y' to play again:
```

10. Session statistics and quitting the game



```
each column considering there is no coin in that row.In case a coin is present i
n the column,the input coin is placed above the coin present.
2.Wait for your turn before placing the coin in order to avoid faulty placement
of coins.
3.The looser of the game gets 10 points by default while the winner gets a minim
um of 50 points which increases with lesser amoiunt of moves.
4.Please follow the instructions at every step.

Hope you enjoy the game.Have fun.
**********************************************************************


Player 1, enter your name:mark
Player 2, enter your name:logan

        1.BLACK        2.GRAY       3.NAVY        4.RED
        5.GREEN        6.YELLOW     7.MAROON      8.WHITE

FROM THE ABOVE COLOR CODES:
mark, Enter your coin color code:4
logan, Enter your coin color code:6
Enter background color code:1
Enter board color code:8
**********************************************************************

Winner: mark
Moves played entirely: 16
Points scored by mark : 84
Points scored by logan : 10
Time elapsed: 100 secs
**********************************************************************

Enter 'Y' to play again:N
**********************************************************************
Session stats:

1. mark:    84

THANK YOU FOR PLAYING WITH US.HOPE YOU ENJOYED :)!
>>>
```

ADVANTAGES OF THE GAME

The game has the following advantages:

- Two player game: Making an interactive and competitive atmosphere for the users.
- Customized graphics: Users get to choose a color as per their wish not only for the coin but also for the background and the board color.
- Interactive: The game provides an user friendly interface with the help of turtle.
- Fault tolerant: The game is roughly fault tolerant and handles errors in user input and in many other places.
- Post game analysis: The game provides a post game analysis of the game providing information like the total amount of moves played, time elapsed and the winner of the game.
- Easy to understand: The game provides an easy to understand interface for the user to play. Besides it provides a tutorial which helps the user to understand the rules and hence play the game efficiently and enjoy doing the same.
- Platform Independent: As it is coded using python, any Operating system with python 2.7 version can run the game efficiently.

CONSTRAINTS AND SCOPE

The application is very much fault tolerant but it does have some constraints attached to it.

1. The game cannot fit in today's real world competition as it utilizes only the basic elements of coding. In today's games a lot of new technology including virtual reality is implemented in games and this game does not stand a chance against them.
2. It does not hold record of the players that play the game which is one of the essential elements of the games releasing throughout the world right now.
3. The game makes use of one of the most basic GUI windows and thus is not very attractive compared to the other games that come out.
4. The game does not take very long to finish as it is a strategy based game comprising of two players and any game can be won in 42 moves unlike many games that have many levels to it and takes at least a few weeks for the game to end providing thrill and entertainment to the user all along.

BIBLIOGRAPHY

- www.Python.org
- www.wikipedia.org
- www.StackOverflow.com
- Youtube tutorial videos
- Computer Science with Python for class XII by Sumitha Aurora
- NCERT Computer Science with python textbook for class XII