VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018, Karnataka, INDIA

A
Python-Mini Project Report (17CS664)
on

"*Tic-Tac-Toe*"

Submitted in partial fulfillment of the requirements for the 6th Semester

Bachelor of Engineering
In
INFORMATION SCIENCE AND ENGINEERING

For the Academic year
2020-2021

BY

ASHWITH ANAND	**1PE17IS019**
ADIL RAUF	**1PE17IS003**
CHAITHANYA P	**1PE18IS400**

Under the Guidance of
Prof. Nivedita Kasturi
Assistant Professor, Dept. of CSE

Department of Information Science and Engineering
PESIT BANGALORE SOUTH CAMPUS
Hosur Road, Bengaluru -560100

PESIT BANGALORE SOUTH CAMPUS

Hosur Road, Bengaluru -560100

Department of Information Science and Engineering

CERTIFICATE

Certified that the Python-Mini Project work entitled ***"Tic-Tac-Toe"*** *bonafide work carried out by* ***Ashwith Anand, Adil Rauf and Chaithanya,*** *bearing USN* ***1PE17IS019, 1PE17IS003 and 1PE18IS400****, student of* ***PESIT Bangalore South Campus*** *in partial fulfillment for the award of* ***6th Semester Bachelor of Engineering*** *in* ***Information Science and Engineering*** *of the* ***Visvesvaraya Technological University****, Belgaum during the year 2020-2021.*

Abstract

Play and create Tic-Tac-Toe, a very renowned game we have all got our hands on since our childhood. To do this task, we will use some in-built libraries of Python namely which are Tkinter and Random. We will code the classic paper-pen game with a nice GUI. We will be coding it in Python and we will be using Tkinter for the interface Tic Tac Toe is a 2 player game where each player has a symbol (either X or O) and plays alternately to mark their symbol on a 3x3 grid. If any player gets their symbol consecutively 3 times in a row, column or diagonal then that player is the winner. At the start we made a 2D array board which is a representation of the actual board. Tkinter is Python's de-facto standard GUI (Graphical User Interface) package.

Table of Contents

INTRODUCTION

Python Tic Tac Toe – Develop a Game in Python

It's no doubt, you must have played Tic Tac Toe in your school days and every one of us loves to play the game. You will be surprised to know that the game of Tic Tac Toe is known to exist since ancient Egypt times.

With this Python project we are going to build an interactive game of Tic Tac Toe where we'll learn new things along the way.

What is Tic Tac Toe

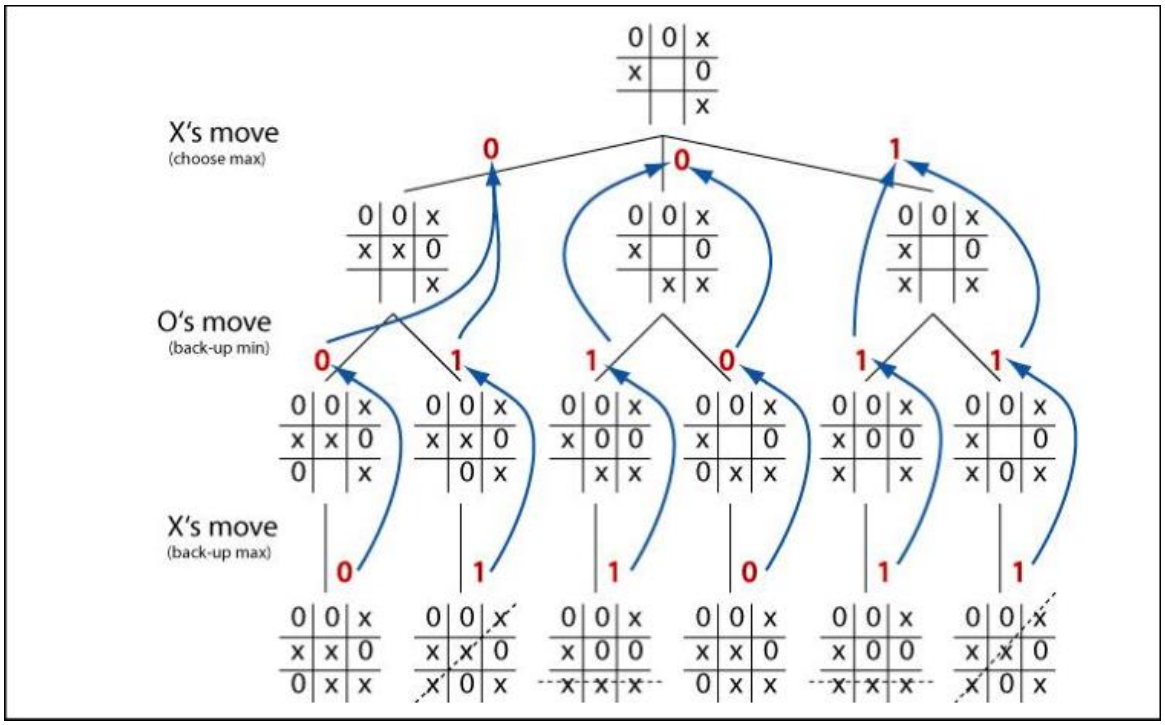
Tic Tac Toe is one of the most played games and is the best time killer game that you can play anywhere with just a pen and paper. If you don't know how to play this game don't worry let us first understand that.

The game is played by two individuals. First, we draw a board with a 3×3 square grid. The first player chooses 'X' and draws it on any of the square grid, then it's the chance of the second player to draw 'O' on the available spaces. Like this, the players draw 'X' and 'O' alternatively on the empty spaces until a player succeeds in drawing 3 consecutive marks either in the horizontal, vertical or diagonal way. Then the player wins the game otherwise the game draws when all spots are filled.

Topic related to your Mini project
Algorithm/System Architecture
Comparison

In our execution of the Minimax algorithm for solving Tic-Tac-Toe, it works by visualizing all future possible states of the board and constructs it in the form of a tree. When the current board state is given to the algorithm (the root of the tree), it splits into 'n' branches (where n denotes the number of moves that can be chosen by the AI/number of empty cells where the AI can be placed). If any of these new states is a terminal state, no further splits are performed for this state and it is assigned a score the following way:

- score = +1 (if AI wins)
- score = -1 (if AI loses)
- score= 0 (If a draw happens)



- Here is a sample game:

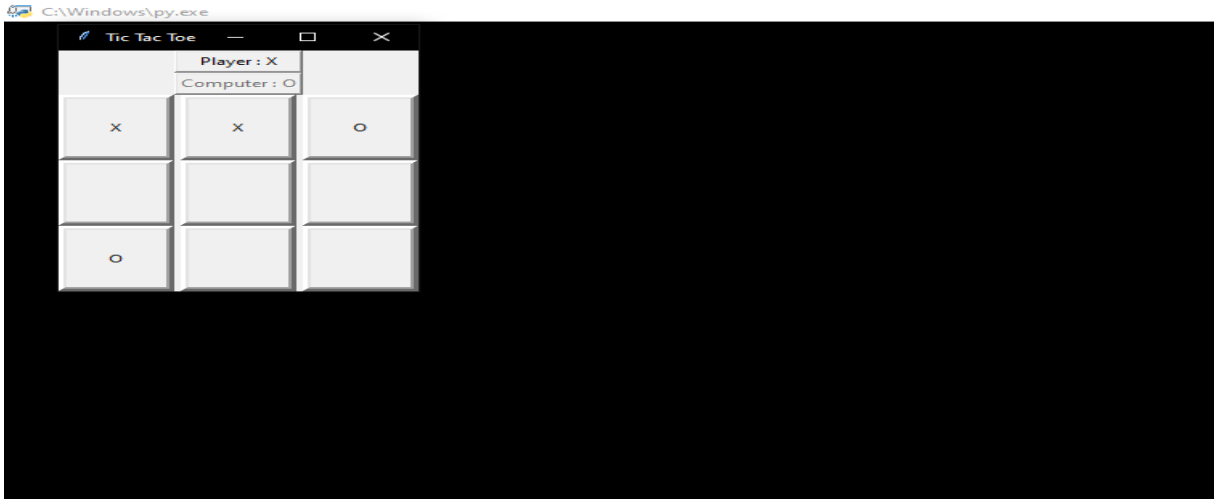
```
New Game!
---------------
|   ||   ||   |
---------------
|   ||   ||   |
---------------
|   ||   ||   |
----------------Choose which player goes first - X (You - the
petty human) or O(The mighty AI): O
AI plays move: 2
---------------

|   || O ||   |
---------------
|   ||   ||   |
---------------
|   ||   ||   |
---------------- Choose where to place (1 to 9): 3
---------------
|   || O || X |
---------------
|   ||   ||   |
---------------
|   ||   ||   |
---------------
AI plays move: 9
---------------
|   || O || X |
---------------
|   ||   ||   |
---------------
|   ||   || O |
---------------- Choose where to place (1 to 9): 5
---------------
|   || O || X |
---------------
|   || X ||   |
---------------
|   ||   || O |
---------------
AI plays move: 7



---------------
|   || O || X |
---------------
|   || X ||   |
---------------
| O ||   || O |
---------------- Choose where to place (1 to 9): 6
---------------
```

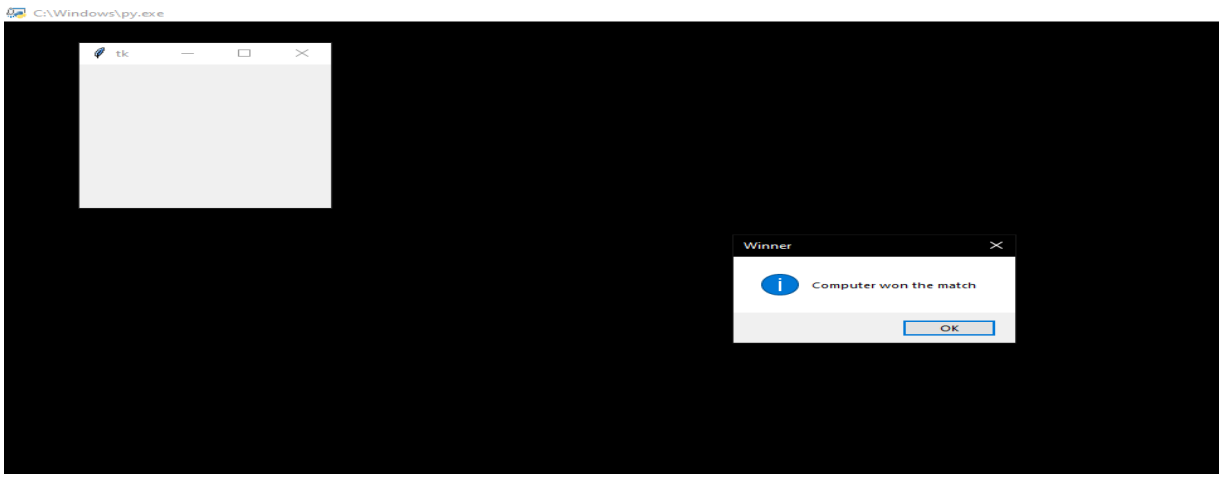
```
|   || O || X |
---------------
|   || X || X |
---------------
| O ||   || O |
---------------
AI plays move: 4
---------------
|   || O || X |
---------------
| O || X || X |
---------------
| O ||   || O |
--------------- Choose where to place (1 to 9): 1
---------------
| X || O || X |
---------------
| O || X || X |
---------------
| O ||   || O |
---------------
AI plays move: 8
---------------
| X || O || X |
---------------
| O || X || X |
---------------
| O || O || O |
---------------
Draw!
```

Simulation Results/Case Study summary

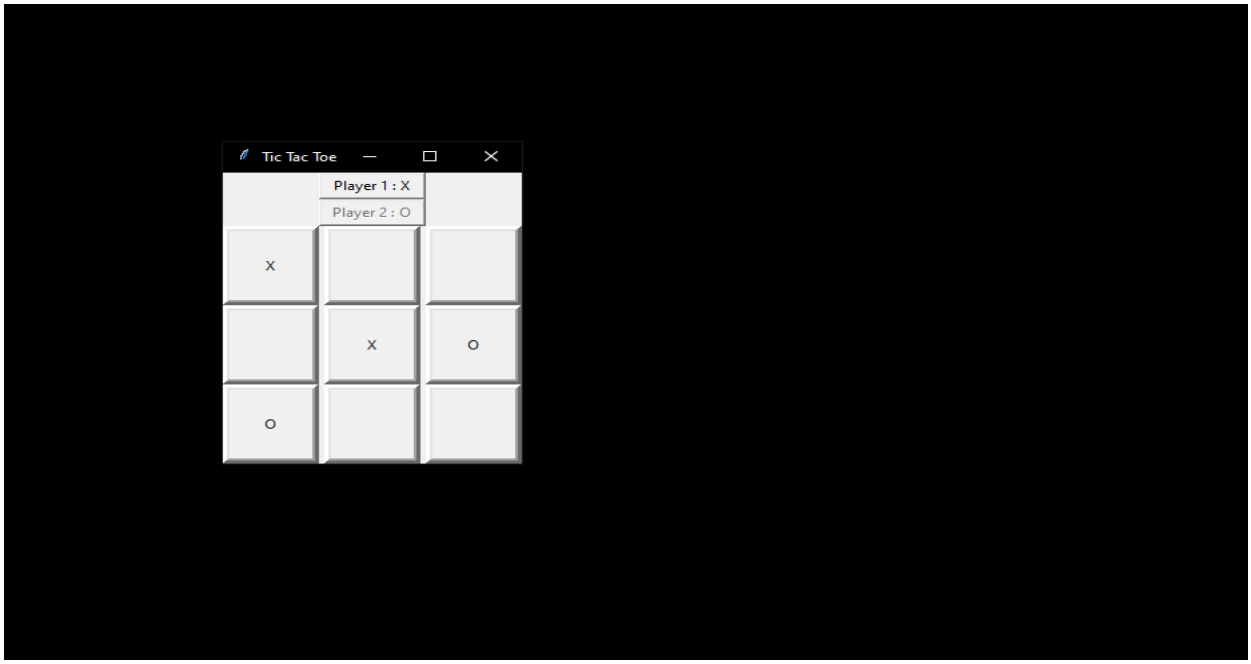
Home screen

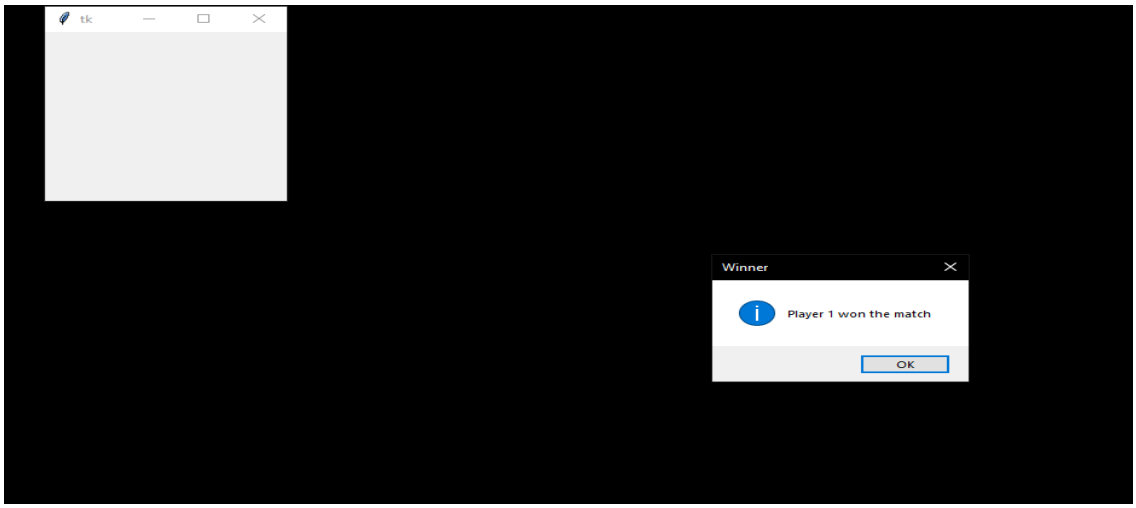
Single Player





Multiplayer





Conclusion

We have successfully devloped the Tic-Tac-Toe game project in python. We use tkinter module for rendering graphics on a display window. We learn how to create buttons and config text on buttons.

In this way we successfully made a Tic-Tac-Toe game python project. We hope you enjoyed building tic-tac-toe game project.

References

https://data-flair.training/blogs/python-tic-tac-toe/

https://towardsdatascience.com/lets-beat-games-using-a-bunch-of-code-part-1-tic-tac-toe-1543e981fec1

https://realpython.com/pygame-a-primer/

Appendix
(copy important module code)

```
# Decide the next move of system
def pc():
    possiblemove = []
    for i in range(len(board)):
        for j in range(len(board[i])):
            if board[i][j] == ' ':
                possiblemove.append([i, j])
    move = []
    if possiblemove == []:
        return
    else:
        for let in ['O', 'X']:
            for i in possiblemove:
                boardcopy = deepcopy(board)
                boardcopy[i[0]][i[1]] = let
                if winner(boardcopy, let):
                    return i
        corner = []
        for i in possiblemove:
            if i in [[0, 0], [0, 2], [2, 0], [2, 2]]:
                corner.append(i)
        if len(corner) > 0:
            move = random.randint(0, len(corner)-1)
            return corner[move]
        edge = []
        for i in possiblemove:
            if i in [[0, 1], [1, 0], [1, 2], [2, 1]]:
```

```
            edge.append(i)
        if len(edge) > 0:
            move = random.randint(0, len(edge)-1)
            return edge[move]


# Configure text on button while playing with system
def get_text_pc(i, j, gb, l1, l2):
    global sign
    if board[i][j] == ' ':
        if sign % 2 == 0:
            l1.config(state=DISABLED)
            l2.config(state=ACTIVE)
            board[i][j] = "X"
        else:
            button[i][j].config(state=ACTIVE)
            l2.config(state=DISABLED)
            l1.config(state=ACTIVE)
            board[i][j] = "O"
        sign += 1
        button[i][j].config(text=board[i][j])
    x = True
    if winner(board, "X"):
        gb.destroy()
        x = False
        box = messagebox.showinfo("Winner", "Player won the match")
    elif winner(board, "O"):
        gb.destroy()
        x = False
```

```
    box = messagebox.showinfo("Winner", "Computer won the match")
elif(isfull()):
    gb.destroy()
    x = False
    box = messagebox.showinfo("Tie Game", "Tie Game")
if(x):
    if sign % 2 != 0:
        move = pc()
        button[move[0]][move[1]].config(state=DISABLED)
        get_text_pc(move[0], move[1], gb, l1, l2)
```