

PROJECT REPORT ON

TIC-TAC-TOE

SUBMITTED IN PARTIAL FULFILLMENT OF THE

AWARD OF

CBSE Senior Secondary Certificate

2018-2019

DONE BY

NAME:.....

REGNO:.....

UNDER THE GUIDANCE OF

Mr. SHAJI.M, PGT Computer Science



KENDRIYA VIDYALAYA

EZHIMALA

KANNUR (DIST), KERALA

KENDRIYA VIDYALAYA EZHIMALA



CERTIFICATE

*Certify that the project entitled “**TIC-TAC-TOE**” is a digitized game done by*

Name:

Reg. No:

During the period of 2017 – 2018 of his/her study in this school under the guidance of Mr. M.SHAJI, PGT Computer Science in partial fulfilment of the requirement for the CBSE Senior Secondary Certificate.

Submitted for practical Examination held on

Project Guide

Teacher in-charge

External Examiner

Principal

DECLARATION

I**reg. no.**hereby declare that the project entitled “***TIC-TAC-TOE***” is the result of original work done by me during the period of study in class XI of computer Science, ***KENDRIYA VIDYALAYA, EZHIMALA*** under the supervision and guidance of ***Mr. M.Shaji***, PGT Computer Science.

This project report is submitted in fulfilment of the requirement for the award of **CBSE Senior Secondary Certificate**.

Place:

Signature:

Date:

ACKNOWLEDGEMENT

I express my sincere thanks to **Mr.Muraleedharan T** principal of our vidyalaya, who has created conducive atmosphere to finish my project in time.

A sincere and true word of thanks to Mr. SHAJI.M,
PGT computer science for his valuable suggestion,
guidance and encouragement.

I thank all my sincere friends who helped me for the successful completion of the project.

Last but not the least I would like to thank the almighty and my parents for showering their blessing on me.

ABSTRACT

This project report is about the famous game TIC TAC TOE written in the programming language python. Everyone played this game at least once in their lives. As we were introduced to the concepts of programming, we became curious about digitizing it. As a result of days of hard work we present to you a fully digitized version of the game. By reading this project report you'll be able to understand almost the barebones of the actual game.

By playing this game, you enter our cerebrum and experience a world that we imagine.

Python:

Python is an easy to use, object oriented, high level programming language. The programmer can easily read and interpret the code due to its simplicity.

The program is interpreted using the python interpreter.

INDEX

S.NO	TOPIC	PAGE NUMBER
1	INTRODUCTION	1
2	SYSTEM CONFIGURATION	2
3	DEVELOPMENT MACHINE	3
3	ALGORITHM	5
4	MODULES USED	8
5	CODE WALKTHROUGH	9
6	THE CODE	14
7	HOW TO RUN THE CODE (EXECUTION)	22
8	FURTHER DEVELOPMENT	23
9	CONCLUSION	24
10	BIBLIOGRAPHY	24

INTRODUCTION

TIC TAC TOE is a time passing game played by two players. The player has the objective to get all their markers in a row. That is, the combination of three markers in horizontal, vertical or diagonal of a 3 by 3 grid will decide the victory of the player.

This is not a very challenging game but still it is one of the most played game to pass time.

The player can choose either X or O as their marker and then continue the game as told above.

When a state is reached where both the player could not succeed in making a combination, the game is considered as a tie and then a fresh grid is drawn for retry.

By digitizing the game using python, we created a score board to keep track of the scores of each player which will be updated automatically when the program senses a combination.

The players will be asked to select their marker at the time of starting of the game.

This game can keep us entertained throughout.

SYSTEM CONFIGURATION

The configurations of the development machine:

Hardware:

- Intel Core i7 processor
- 1080p display
- 8GB DDR4 RAM
- 1 TB hard drive

Software:

- Windows 10
- Python 3.6.5 installed
- Visual Studio 2017

However,

MINIMUM SYSTEM REQUIREMENTS:

HARDWARE:

- Any Intel/AMD processor, even ATMEGA will run
- 128 KB RAM
- 10MB hard drive space

SOFTWARE:

- Any windows/Linux OS
- Python 3 or above installed

DEVELOPMENT MACHINE

WINDOWS 10:

Windows 10 is a series of personal computer operating systems produced by Microsoft as part of its Windows NT family of operating systems. It is the successor to Windows 8.1, and was released to manufacturing on July 15, 2015, and broadly released for retail sale on July 29, 2015. Windows 10 receives new builds on an ongoing basis, which are available at no additional cost to users. Mainstream builds of Windows 10 are labelled version YYMM with YY representing the year and MM representing the month of release. For example, Version 1809 for September 2018. There are additional test builds of Windows 10 available to Windows Insiders. Devices in enterprise environments can receive these updates at a slower pace, or use long-term support milestones that only receive critical updates, such as security patches, over their ten-year lifespan of extended support.

One of Windows 10's most notable features is support for universal apps, an expansion of the Metro-style apps first introduced in Windows 8. Universal apps can be designed to run across multiple Microsoft product families with nearly identical code—including PCs, tablets, smartphones, embedded systems, Xbox One, Surface Hub and Mixed Reality. The Windows user interface was revised to handle transitions between a mouse-oriented interface and a touchscreen-optimized interface based on available input devices—particularly

on 2-in-1 PCs, both interfaces include an updated Start menu which incorporates elements of Windows 7's traditional Start menu with the tiles of Windows 8. Windows 10 also introduced the Microsoft Edge web browser, a virtual desktop system, a window and desktop management feature called Task View, support for fingerprint and face recognition login, new security features for enterprise environments, and DirectX 12.

PYTHON:

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Van Rossum led the language community until stepping down as leader in July 2018.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

But how does the code work?

–Algorithm

The first step was to create a grid where we could store markers. This was done by creating three arrays 1 for each column. Then we created a function that prints the board to the console window. Along with the grid, a reference grid was also made for the users to easily talk with the program on where the markers was to be kept.

Now, a player class was created that contained all the data of the player. The Name of the player, Score, Turn, Marker... is being stored in this class.

Now, a function is created that inputs the user details. This is executed at the game start up.

My next intention was to make inputs for the game. For this I used the input function in python. The input was taken from the number pad. After the input was received, a function decides where to put the marker in one of the three lists created.

Before the marker could be placed, the function have to decide whether the place that the user input was occupied by another marker. If it is already occupied, the user is alerted by a small message saying that the position is already occupied by a marker.

After the valid input is received, the marker is added to the array and then the screen is refreshed using the refresh function.

It was hard to change the turn one after the other. A variable is used to store the marker which is currently in use and then changed when the turn is changed.

The player object is created by input of name and preferred markers through the console.

We had to use the OS module to refresh the console by the use of the good old CLS command.

The code uses the f-string available in python 3 or higher. This made it easy to include player name and scores in messages. This was very useful in making a clean code without much noise.

A function called “PlayGame” Controls all the logic of the game. This is the main core of the game. This function controls the turns, valid/invalid moves...

This function is in turn controlled by the initialize function which does all the setup work at boot.

An infinite loop is used to loop through all the player turns. This infinite loop calls the playgame function each time it loops. This makes it easy to control all the variables and change in turns.

Now, as the user input was up and running, it was time to set combinations for success.

A function returns all the success combinations possible in the game. A loop then loops through all the combinations and then accounts for victory.

An interesting function called FlipBool is used. The purpose of the function is to make a true bool to false and false bool to true. This in turn used to change the x turn and o turn alternatively. This provides flexibility and reduces noise.

Now the scorecard was to be prepared. For this we took the score variable from the player class and then pushed it on to the console each time the program refreshed. The use of refresh function made it easy to make changes in the console window during execution of the code.

Now, the mechanism of incrementing the score card was being made. The program when sense the combination, it looks for the marker and the player owning the marker from the player object. A short message is shown about the victory of the player and his score incremented at the time of combination.

MODULES USED

As this game does not have mind blowing graphics or anything, we had to use only one module

OS:

This module in python is used to interact with the command line of the operating system. As python did not provide an easy way to refresh the console window, we had to run the CLS command in command prompt.

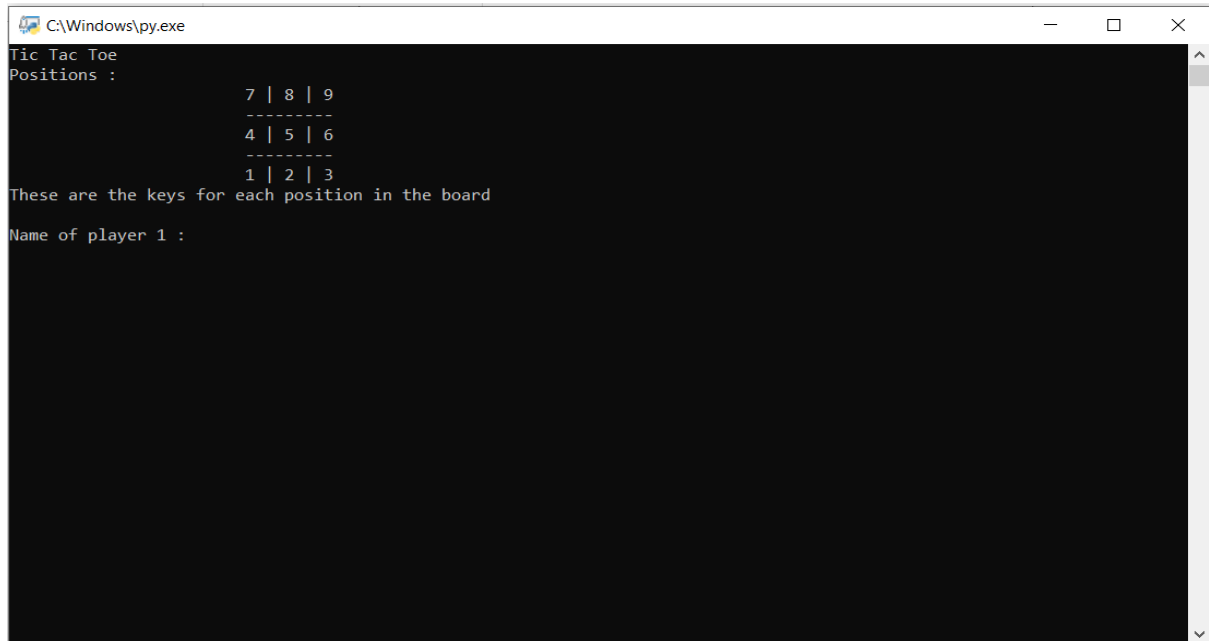
This command can be run using

```
os.system('cls')
```

The `os.system()` function takes an argument which is passed on to the command prompt. As we used the CLS command, we could refresh the screen without further complications

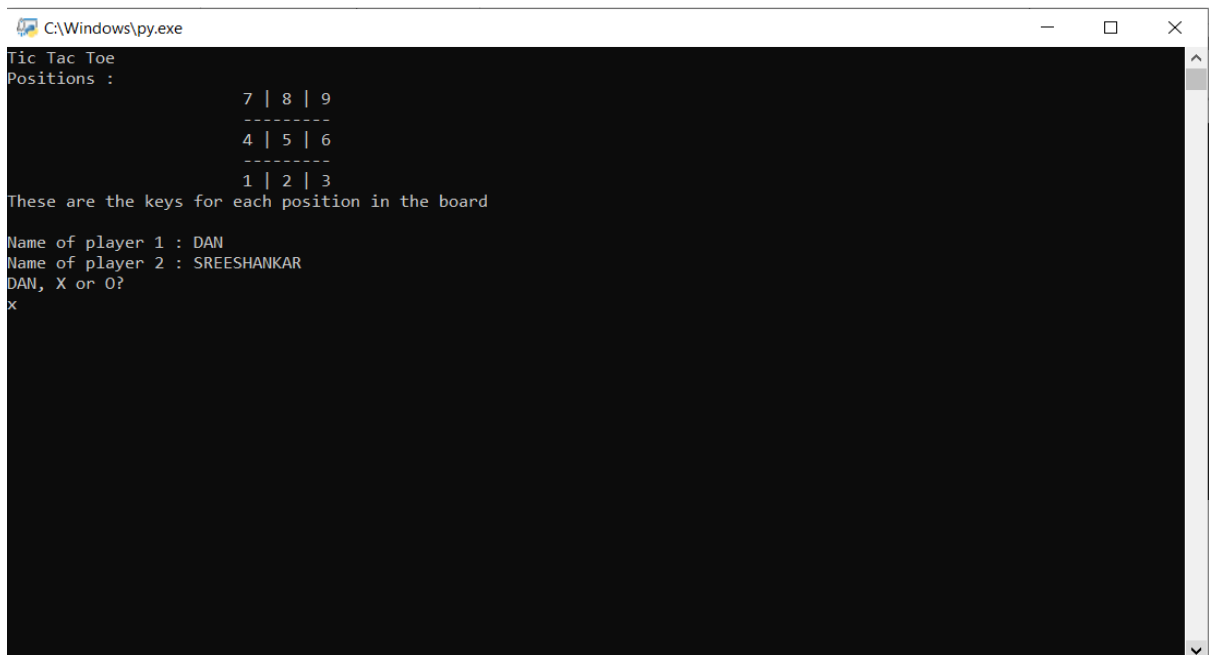
CODE WALKTHROUGH

At the start of the game, we see a grid of numbers (The reference grid). The input cursor blinks for the input of player names



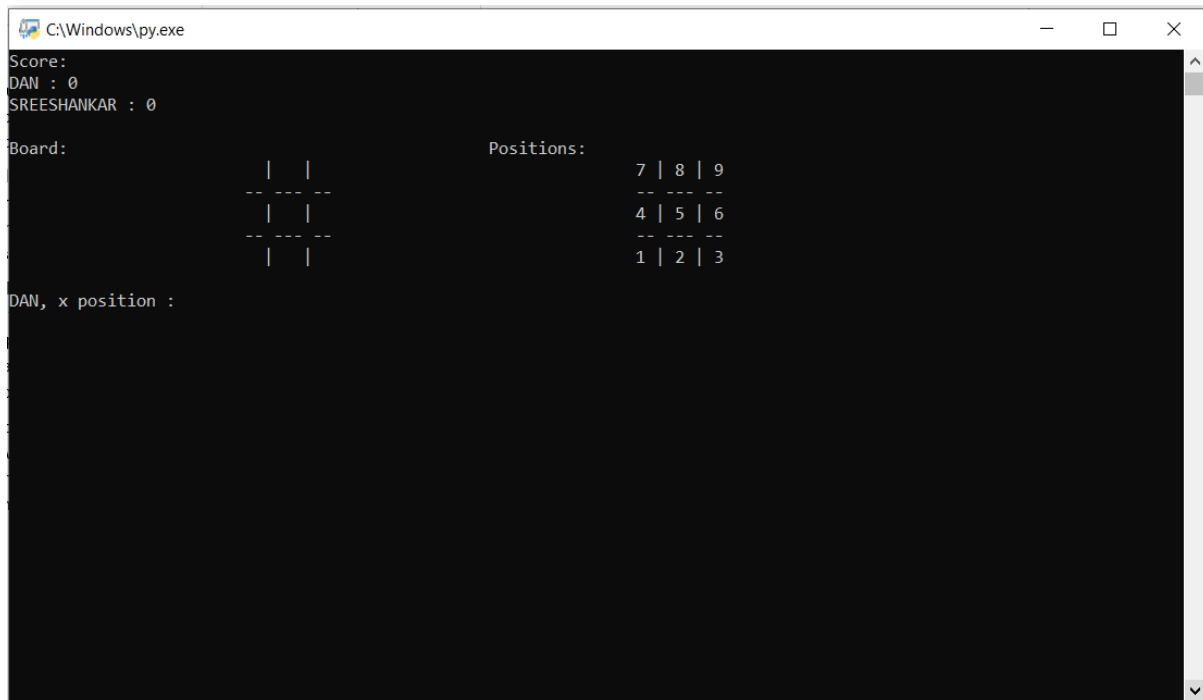
```
C:\Windows\py.exe
Tic Tac Toe
Positions :
      7 | 8 | 9
      -----
      4 | 5 | 6
      -----
      1 | 2 | 3
These are the keys for each position in the board
Name of player 1 :
```

When the player names are entered, the game asks for the first player for the marker.



```
C:\Windows\py.exe
Tic Tac Toe
Positions :
      7 | 8 | 9
      -----
      4 | 5 | 6
      -----
      1 | 2 | 3
These are the keys for each position in the board
Name of player 1 : DAN
Name of player 2 : SREESHANKAR
DAN, X or O?
x
```

Now we see the screen refresh and a fresh board is presented to us. Also, the score board is visible where the scores read zeroes.



```
C:\Windows\py.exe
Score:
DAN : 0
SREESHANKAR : 0

Board:
      | |
  ---|---
      | |
  ---|---
      | |

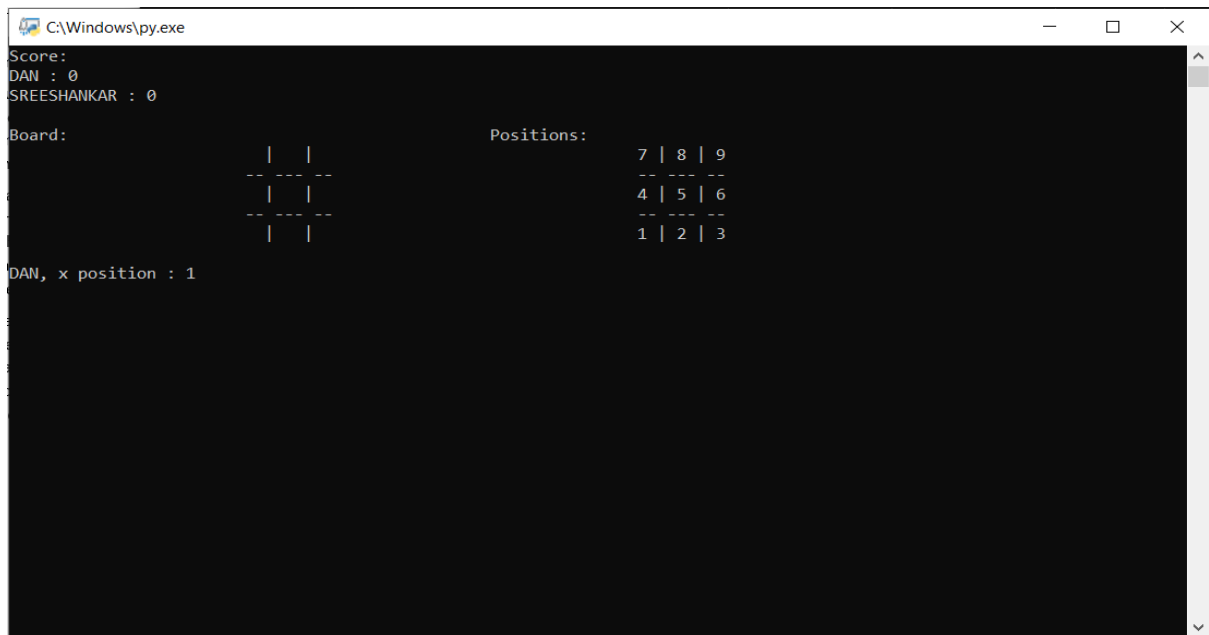
Positions:
      7 | 8 | 9
  ---|---
      4 | 5 | 6
  ---|---
      1 | 2 | 3

DAN, x position :
```

Note the reference grid shown in the right side

Now the game asks the first player to set his marker. As I selected X as the marker for the first player, it asks for X position.

The next two images will show how the game enters the data into the grid.



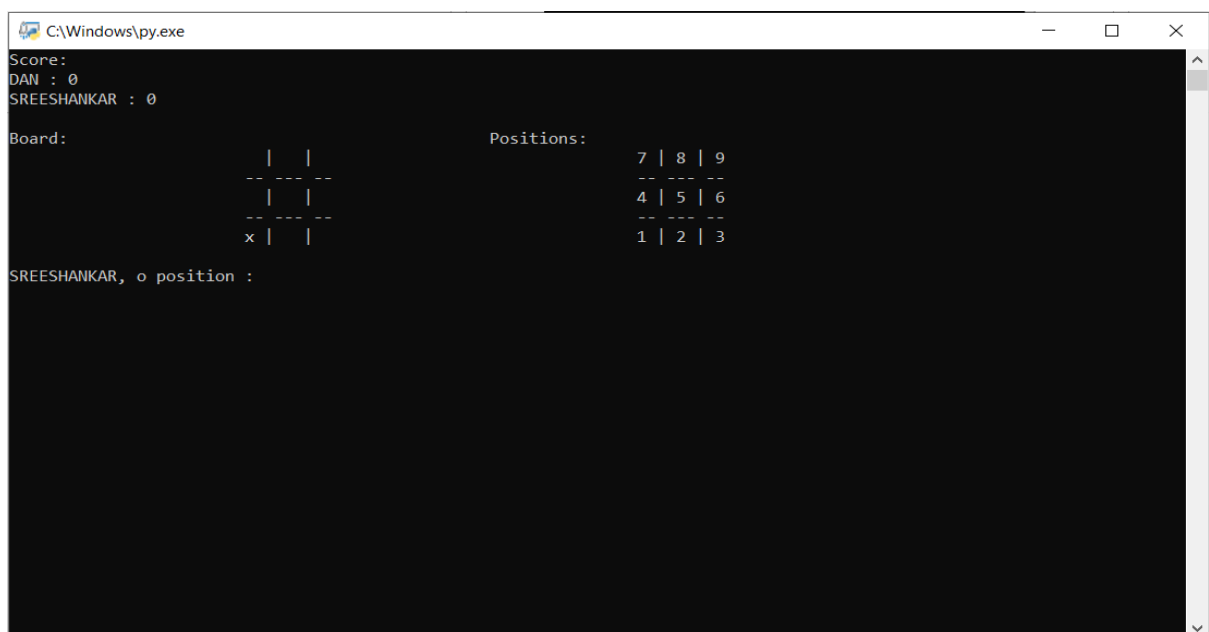
```
C:\Windows\py.exe
Score:
DAN : 0
SREESHANKAR : 0

Board:
      | |
  ---|---
      | |
  ---|---
      | |

Positions:
  7 | 8 | 9
  ---|---
  4 | 5 | 6
  ---|---
  1 | 2 | 3

DAN, x position : 1
```

And when entered...



```
C:\Windows\py.exe
Score:
DAN : 0
SREESHANKAR : 0

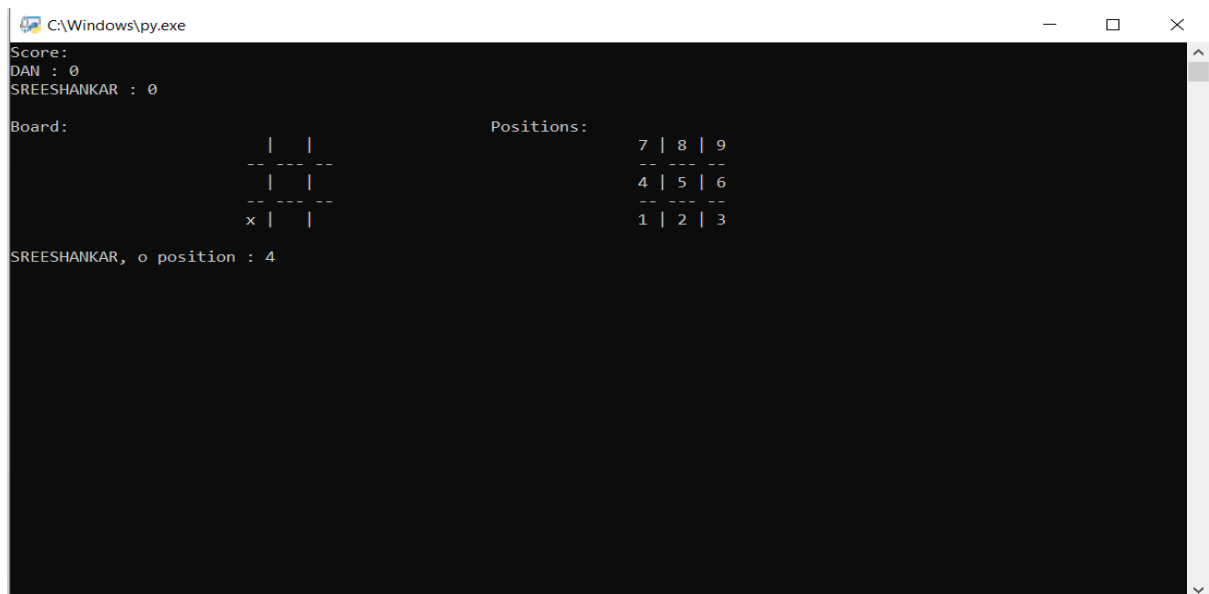
Board:
      | |
  ---|---
      | |
  ---|---
  x  | |

Positions:
  7 | 8 | 9
  ---|---
  4 | 5 | 6
  ---|---
  1 | 2 | 3

SREESHANKAR, o position :
```

We see that now the game asks the O position to the second player

The next two images shows the data entry of the second player



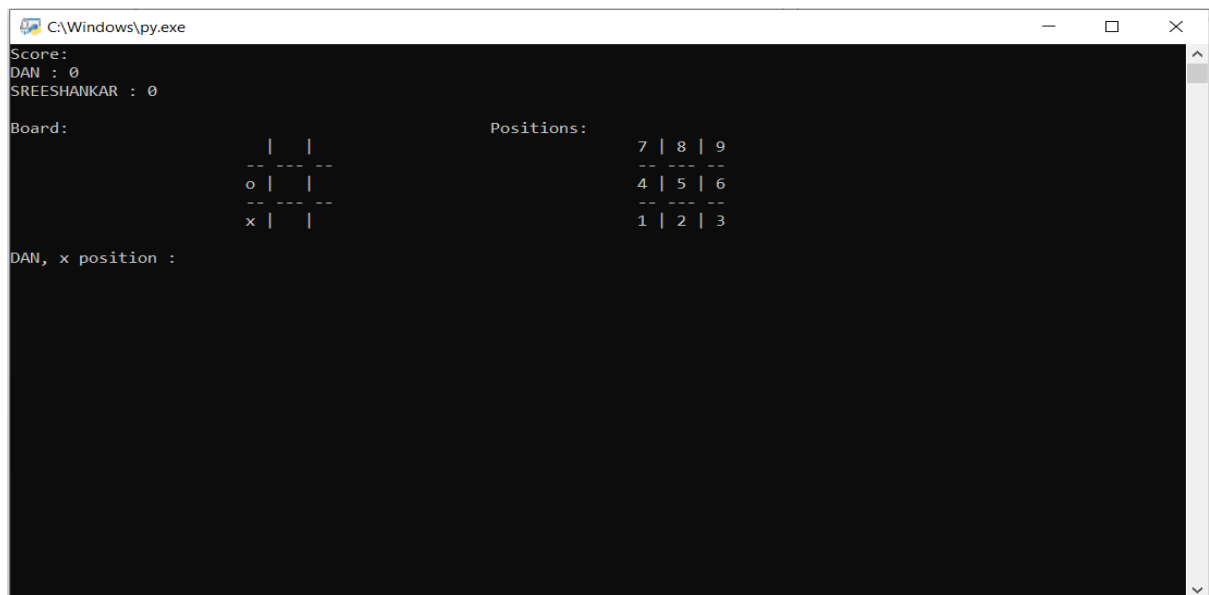
```
C:\Windows\py.exe
Score:
DAN : 0
SREESHANKAR : 0

Board:
      | |
      - - -
      | |
      - - -
    x | |

Positions:
      7 | 8 | 9
      - - -
      4 | 5 | 6
      - - -
      1 | 2 | 3

SREESHANKAR, o position : 4
```

And when entered...



```
C:\Windows\py.exe
Score:
DAN : 0
SREESHANKAR : 0

Board:
      | |
      - - -
    o | |
      - - -
    x | |

Positions:
      7 | 8 | 9
      - - -
      4 | 5 | 6
      - - -
      1 | 2 | 3

DAN, x position :
```

Now, the o marker is set in place and the game in turn changes the turn to the net player

Now, let's see what happens when DAN wins the game...

```
C:\Windows\py.exe
Score:
DAN : 1
SREESHANKAR : 0

Board:
      o |  | 
      ---
      o |  | 
      ---
      x | x | x

Positions:
      7 | 8 | 9
      ---
      4 | 5 | 6
      ---
      1 | 2 | 3

Bingo DAN, your score : 1
Do you want to play again?(y/n)
```

We can see the short message saying that DAN won the game. Also notice the score card where the new score is added. The game asks the players if they wish to continue the game or stop it.

If no combination is achieved, the game is declared a tie and no player is given a score

```
C:\Windows\py.exe
Score:
DAN : 1
SREESHANKAR : 1

Board:
      o | o | x
      ---
      x | o | x
      ---
      o | x | o

Positions:
      7 | 8 | 9
      ---
      4 | 5 | 6
      ---
      1 | 2 | 3

Game Over
Do you want to play again?(y/n)
```

The Code

```
import os
```

```
r1=[' ',' ',' ']
```

```
r2=[' ',' ',' ']
```

```
r3=[' ',' ',' ']
```

```
def flipBool(x):
```

```
    if x:
```

```
        return False
```

```
    return True
```

```
def setBoard():
```

```
    global r1,r2,r3
```

```
    r1=[' ',' ',' ']
```

```
    r2=[' ',' ',' ']
```

```
    r3=[' ',' ',' ']
```

```
def setPlayerChoice():
```

```
    Player1.setChoice(input(f"{Player1.Pname}, X or O?\n").lower())
```

```
    if Player1.Pchoice == 'x' :
```

```
        Player2.setChoice("o")
```

else:

Player2.setChoice("x")

def SuccessCombination(r1,r2,r3):

return [(r1[0],r1[1],r1[2]),

(r2[0],r2[1],r2[2]),

(r3[0],r3[1],r3[2]),

(r1[0],r2[0],r3[0]),

(r1[1],r2[1],r3[1]),

(r1[2],r2[2],r3[2]),

(r1[0],r2[1],r3[2]),

(r1[2],r2[1],r3[0])]

class Player():

Pname=None

Pchoice=None

Pscore=0

def __init__(self,name):

self.Pname=name

def IncScore(self):

self.Pscore+=1

```
def setChoice(self,choice):
    if choice in ['x','o']:
        self.Pchoice=choice
    else:
        print("Invalid Choice")
```

```
def Initialize():
```

```
setPlayerChoice()  
setBoard()  
xturn=True
```

```
def PrintBoard():
```

```
os.system('cls')

print(f"Score:\n{Player1.Pname} :
{Player1.Pscore}\n{Player2.Pname} :
{Player2.Pscore}\n\nBoard:\t\t\t\t\t Positions: \n\t\t\t\t\t {r1[0]} |
{r1[1]} | {r1[2]} \t\t\t\t\t 7 | 8 | 9\n\t\t\t\t\t -- --- --\t\t\t\t\t -- --- --
\n\t\t\t\t\t {r2[0]} | {r2[1]} | {r2[2]}\t\t\t\t\t 4 | 5 | 6\n\t\t\t\t\t -- --- --\t\t\t\t\t -- --- --
\n\t\t\t\t\t {r3[0]} | {r3[1]} | {r3[2]}\t\t\t\t\t 1 | 2 | 3\n")
```

```

def entl(x,position):
    global xturn
    if position in [1,2,3]:
        if r3[position-1]==' ':
            r3[position-1]=x
        else:
            print("That position is already occupied")
            xturn=flipBool(xturn)
    elif position in [4,5,6]:
        if r2[position-4]==' ':
            r2[position-4]=x
        else:
            print("That position is already occupied")
            xturn=flipBool(xturn)
    elif position in [7,8,9]:
        if r1[position-7]==' ':
            r1[position-7]=x
        else:
            print("That position is already occupied")
            xturn=flipBool(xturn)
    else:
        print("Invalid input")

```

```

def playgame(p1,p2):
    global xturn,GameRst
    if p1.Pchoice=='x':
        xplayer=p1
        oplayer=p2
    else:
        xplayer=p2
        oplayer=p1

    if xturn:
        choice=int(input(f'{xplayer.Pname}, x position : '))
        xturn=False
        entl('x',choice)
    else:
        choice=int(input(f'{oplayer.Pname}, o position : '))
        xturn=True
        entl('o',choice)

    for a,b,c in SuccessCombination(r1,r2,r3):
        if a==b and b==c and a==c and a in ['o','x'] and b in ['o','x'] and c
in ['o','x']:
            if a=='x':
                xplayer.Pscore+=1

```



```

        PrintBoard()
        print(f"\nBingo {xplayer.Pname}, your score :
{xplayer.Pscore}\n")
    else:
        oplayer.Pscore+=1
        PrintBoard()
        print(f"\nBingo {oplayer.Pname}, your score :
{oplayer.Pscore}\n")
    GameRst=True
    pass

check_game_over=True

for r1_,r2_,r3_ in zip(r1,r2,r3):
    if r1_==' ' or r2_==' ' or r3_==' ':
        check_game_over=False

if check_game_over==True:
    PrintBoard()
    print("Game Over")
    GameRst=True

```

GameRst=False

Playgame=True

```
print("Tic Tac Toe\nPositions :\n\t\t\t\t7 | 8 | 9\n\t\t\t\t-----\n\t\t\t\t4  
| 5 | 6\n\t\t\t\t-----\n\t\t\t\t1 | 2 | 3\nThese are the keys for each  
position in the board\n")
```

Player1 = Player(input("Name of player 1 : "))

Player2 = Player(input("Name of player 2 : "))

while True:

 if Player1.Pname==Player2.Pname:

 print("\nName of both players cannot be the same\n")

 Player2 = Player(input("Name of player 2 : "))

 else:

 break

Initialize()

xturn=True

```
while Playgame:
```

```
    os.system('cls')
```

```
    PrintBoard()
```

```
    playgame(Player1,Player2)
```

```
if GameRst:
```

```
    c=input("Do you want to play again?(y/n)").lower()
```

```
    if c=="n":
```

```
        break
```

```
    print("Resetting game...")
```

```
    os.system('cls')
```

```
    GameRst=False
```

```
    Initialize()
```

HOW TO RUN THIS CODE?

At first, we need to download the python package from

www.python.org

Further steps:

1. **Open IDLE,**

This can be done by searching IDLE on the windows taskbar.

2. **File>New File,**

As the IDLE brings you by default to the interactive shell, it is necessary to open the script mode present in python package

3. **Typing the code,**

The code is typed into the new window which will show syntax highlighting.

4. **Run>Run Module,**

This is the best part... The code will now run on the IDLE window and the output can be seen. The game is now playable.

5. **Enjoy,**

This is the most important step. Make sure you enjoy the game with your friends. Play this game to pass time and enjoy.

FURTHER DEVELOPMENT

Just as nobody is perfect, no program is perfect. There is always scope for improvement. If you are an expert in python, try to read and understand the code. It is not hard as *simplicity* is Python's one of the strong points. After understanding the code, try to make changes the game. After you've successfully mastered the code, try to customize the game.

SOME TIPS FOR IMPROVEMENT:

- The game can be made more challenging by making it a TIC TAC TOE championship. This will add more thrill to the game.
- Refer to the *socket* module and try to make the game a multiplayer game to run the game across a network. This gives an idea on how networking works and how computers talk to each other.

This code has been written in a flexible manner so that any changes can be easily made to the code to account for an individual's need.

CONCLUSION

The game TIC TAC TOE has been created in python on a windows machine with python 3.6.5 installed and is ready to be played. The game has been tested and retested by our team mates to minimize bugs. The code is flexible and user friendly. A project report is created so as to understand the code better.

Teamwork, development process, debugging... were the goals achieved by the project.

BIBLIOGRAPHY

- Class XI text SUMITA ARORA about programming with python.
- www.tutorialspoint.com
- www.stackoverflow.com
- www.wikipedia.org