INTRODUCTION

Our project entitled "**Tic-Tac-Toe game"** is a paper-and-pencil game for two players, *X* and *O*, who take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.

Players soon discover that the best play from both parties leads to a draw. Hence, tic-tac-toe is most often played by young children.

Because of the simplicity of tic-tac-toe, it is often used as a pedagogical tool for teaching the concepts of good sportsmanship and the branch of artificial intelligence that deals with the searching of game trees. It is straightforward to write a computer program to play tic-tac-toe perfectly or to enumerate the 765 essentially different positions (the state space complexity) or the 26,830 possible games up to rotations and reflections (the game tree complexity) on this space.

The game can be generalized to an m, n, k -game in which two players alternate placing stones of their own color on an $m \times n$ board, with the goal of getting k of their own color in a row. Tic-tac-toe is the (3, 3, 3)-game. Harary's generalized tic-tac-toe is an even broader generalization of tic-tac-toe. It can also be generalized as a n^d game. Tic-tac-toe is the game where n equals 3 and d equals 2. If played properly, the game will end in a draw, making tic-tac-toe a futile game.

AIM OF THE APPLICATION

This project **"Tic-Tac-Toe game"** was with the aim to prepare the following

- ✓ To reduce man power to play the game.
- ✓ To reduce the wastage of materials.
- ✓ To make the user to be versatile with the game.

SYSTEM REQUIREMENTS

Hardware Requirements

1. RAM **:** 4 GB

2. Memory **:** 300 GB

3. Processor **:** Intel Core Processor

Software Requirements

1. Operating System **:** Windows 8.1 Pro

2. Compiler **:** Turbo C++ version 3.0

PROBLEM DESCRIPTION

1. This game utilizes more ink.
2. This game consumes more spaces in a paper

The project **"Tic-Tac-Toe game"** is an application through which one can easily overcome the above issues.

OBJECTIVE AND PROPOSED SYSTEM

"Tic-Tac-Toe game", is an application beneficial for the user to use and operate the application and acquire various benefits from it.

It is a simple application, which is generated to ease the way to play the game called "Tic-Tac-Toe".

LIMITATIONS OF THE EXISTING SYSTEM

WASTAGE OF RESOURCES

When we play using pen and paper, more ink and papers turn into a wastage. We also use more ink to make the Tic-Tac-Toe board and to update the scores.

ADVANTAGES OF PROPOSED SYSTEM

AVOIDS WASTAGE OF RESOURCES

Lots of papers and ink can be saved by playing this game in a computer.

USER FRIENDLY

This game is made more and more user-friendly to avoid making mistakes while playing the game

AVOIDS MISTAKES

This game is made more precisely so that it avoids the user to make invalid moves.

MODULE DESCRIPTION

1. Rules Screen

This module helps the end-user to understand the rules and regulations of Tic-Tac-Toe game.

2. User Details

This module helps the compiler to get the following details from the players

- ✓ Name of the player
- ✓ Age of the player
- ✓ City of the player

3. Game Screen

This module helps the end-user to play the game successfully by the following ways:

- ✓ Ensuring the user to make their respective mark on the respective boxes.
- ✓ Checking whether the user has won the game after each move.
- ✓ Predicting the winner of the game more accurately.
- ✓ Eliminating the user to make invalid moves.

CONCLUSION

The project entitled "**Tic-Tac-Toe"** game is successfully designed and implemented. The main objective of this application is to play the game more easily without the wastage of resources. It reduces the manual work done by the user, which is to draw the game board of the game. It saves time to do the required arrangements for the making the game board.

This application provides a user friendly approach towards the system. The system has been developed and is found to satisfy all the requirements. This hopes that the system will be utilised its maximum and will do good performance in a long run. The main resource required, which helps us to make all it easy is the modern computerized technology.

FUTURE ENHANCEMENTS

The program would be updated with recognisable features, such as the following

- Increase in dimensions of the board.
- Increase in graphical display.
- Increase in the number of players.

The new system is designed such that those enhancements can be integrated with current modules easily with less integration work.

SOURCE CODE

//Program to create “Tic-Tac-Toe” game.

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<process.h>
#include<dos.h>
#include<string.h>

char square[10] = {'0','1','2','3','4','5','6','7','8','9'};
char play_n[3][100],play_city[3][100];
char load[42]={'-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' ',' ','-',' '};
int play_age[3];


void start_screen();
void rules_screen();
void player_1();
void player_2();
void loading();
int game_screen();
int checkwin();
void board();
```

```
// main function
int main()
 {
   start_screen();
   rules_screen();
   player_1();
   player_2();
   loading();
   game_screen();
   return 0;
 }


//Funtion to show the Starting Screen
void start_screen()
 {
   clrscr();
   cout<<"\n\n\t\t\t  Welcome to TIC TAC TOE ";
   cout<<"\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\t\t\t  Press any key to continue";
   getch();
 }
```

//Function to display the rules

```
void rules_screen()
 {
   clrscr();
   cout<<"\n\n\t\t\t   Rules and regulations";
   cout<<"\n\n\n 1) The game is played on a grid that's 3 squares by 3 squares.";
   cout<<"\n\n 2) Player 1 is X, Player 2 is O.";
   cout<<"\n\n 3) Players take turns putting their marks in numbered squares.";
   cout<<"\n\n 4) The first player to get 3 of her marks in a row is the winner.";
   cout<<"\n\n 5) When all 9 squares are full, the game ends in a tie.";
   cout<<"\n\n\n\n\n\n\n\t\t\t  Press any key to continue";
   getch();
 }
```

//Function to get the details of player 1

```
void player_1()
 {
   clrscr();
   cout<<"\n\n\t\t\t   Enter the details of Player 1";
   cout<<"\n\n\n";

   cout<<" Enter your name (max 100 characters): ";
   gets(play_n[1]);

   cout<<"\n\n Enter your age: ";
   cin>>play_age[1];
```

```
  if(play_age[1]>100||play_age[1]<1)
    {
      cout<<"\n\n\n\n\n\n\n\n\t\t          Your age is not a valid age";
      cout<<"\n\n\t\t\t      Press any key to exit";
      getch();
      exit(0);
    }

  cout<<"\n\n Enter your city (max 50 characters): ";
  gets(play_city[1]);

  cout<<"\n\n\n\n\n\n\n\n\t\t\t  Press any key to continue";
  getch();
 }
```

//Function to get the details of player 2

```
void player_2()
 {
  clrscr();
  cout<<"\n\n\t\t\t   Enter the details of Player 2";
  cout<<"\n\n\n";

  cout<<" Enter your name (max 100 characters): ";
  gets(play_n[2]);
```

```
  cout<<"\n\n Enter your age: ";
  cin>>play_age[2];

  if(play_age[2]>100||play_age[2]<1)
    {
      cout<<"n\n\n\n\n\n\n\t\t          Your age is not a valid age";
      cout<<"\n\n\t\t\t      Press any key to exit";
      getch();
      exit(0);
    }

  cout<<"\n\n Enter your city (max 50 characters): ";
  gets(play_city[2]);

  cout<<"\n\n\n\n\n\n\n\t\t\t  Press any key to continue";
  getch();
 }
```

//Function to show the loading screen

```
void loading()
 {
  clrscr();

  for(int k=0;k<=4;k++)
     {
        for(int i=0;i<41;i++)
```

```
        {
                clrscr();
            cout<<"\n\n\n\n\n\n\n\n\n\t\t\t\t    LOADING\n\n";
            cout<<"\t\t     ";
            for(int j=0;j<=i;j++)
              {
                cout<<load[j];
              }
                delay(20);
        }
    }

  cout<<"\n\n\n\n\n\n\t\t\t       Your game is ready";
  cout<<"\n\n\t\t\t    Press any key to continue";
  getch();
 }
```

//Function to show the game screen

```
int game_screen()
 {
  clrscr();
  int player = 1,i,choice;
  char mark;

  do
    {
```

```
clrscr();
board();
player=(player%2)?1:2;

cout << play_n[player]<<  ", please enter a number:  ";
cin >> choice;

mark=(player == 1) ? 'X' : '0';

if (choice == 1 && square[1] == '1')
        square[1] = mark;
else if (choice == 2 && square[2] == '2')
        square[2] = mark;
else if (choice == 3 && square[3] == '3')
        square[3] = mark;
else if (choice == 4 && square[4] == '4')
        square[4] = mark;
else if (choice == 5 && square[5] == '5')
        square[5] = mark;
else if (choice == 6 && square[6] == '6')
        square[6] = mark;
else if (choice == 7 && square[7] == '7')
        square[7] = mark;
else if (choice == 8 && square[8] == '8')
        square[8] = mark;
else if (choice == 9 && square[9] == '9')
        square[9] = mark;
```

```
        else
            {
                cout<<"You have given an invalid move ";
                player--;
                cin.ignore();
                cin.get();
            }
        i=checkwin();
        player++;
        }
        while(i==-1);
  clrscr();
  board();

  if(i==1)
    {
        cout<<"==> Congrats "<<play_n[--player]<<" , you win the game ";
        cout<<"\n==> Your prize will be given to "<<play_city[player]<<" city
centre";
        cout<<"\n==> Say your real age "<<play_age[player]<<" and collect it!!";
    }
  else
        cout<<"\n  ==> Oops , the Game drawn";
  cout<<"\n\n\n\t\t\t      Press any key to exit";
  getch();
  return 0;
 }
```

//Function to check whether the game is won by the player

```
int checkwin()
 {
        if (square[1] == square[2] && square[2] == square[3])
                return 1;
        else if (square[4] == square[5] && square[5] == square[6])
                return 1;
        else if (square[7] == square[8] && square[8] == square[9])
                return 1;
        else if (square[1] == square[4] && square[4] == square[7])
                return 1;
        else if (square[2] == square[5] && square[5] == square[8])
                return 1;
        else if (square[3] == square[6] && square[6] == square[9])
                return 1;
        else if (square[1] == square[5] && square[5] == square[9])
                return 1;
        else if (square[3] == square[5] && square[5] == square[7])
                return 1;
        else if (square[1] != '1' && square[2] != '2' && square[3] != '3' && square[4]
!= '4' && square[5] != '5' && square[6] != '6' && square[7] != '7' && square[8] !=
'8' && square[9] != '9')
                return 0;
        else
                return -1;
 }
```

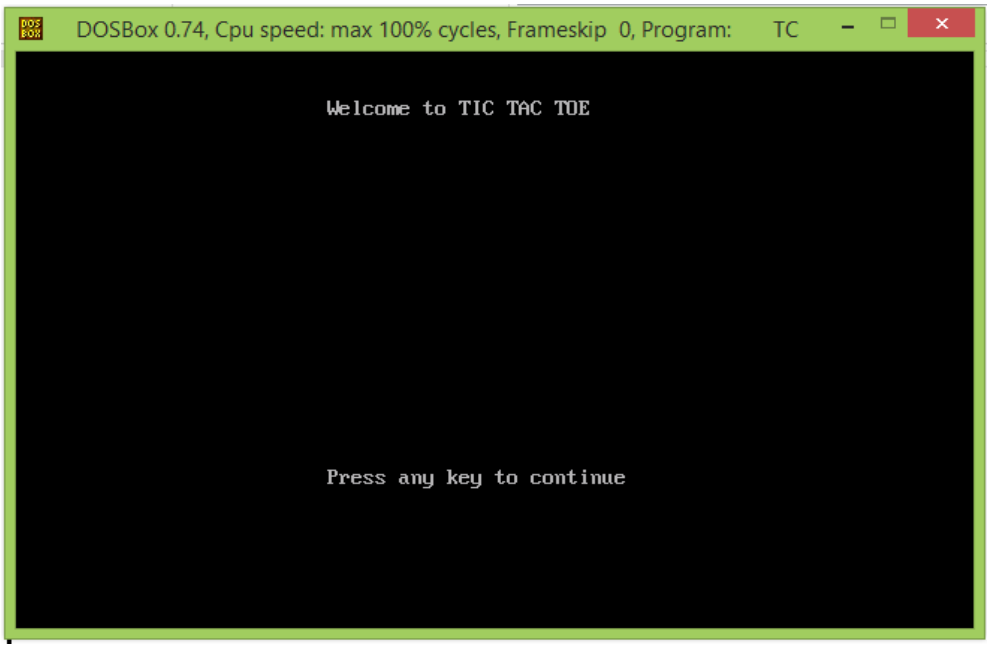
//Function to show the Tic-Tac-Toe board

```
void board()
 {
   cout << "\n\n\t\t\t\t  Tic Tac Toe\n\n";
   cout << endl;
   cout << "\t\t\t           |     |     " << endl;
   cout << "\t\t\t        " << square[1] << "  |  " << square[2] << "  |  " << square[3] <<
endl;
   cout << "\t\t\t       _____|_____|_____" << endl;
   cout << "\t\t\t           |     |     " << endl;
   cout << "\t\t\t        " << square[4] << "  |  " << square[5] << "  |  " << square[6] <<
endl;
   cout << "\t\t\t       _____|_____|_____" << endl;
   cout << "\t\t\t           |     |     " << endl;
   cout << "\t\t\t        " << square[7] << "  |  " << square[8] << "  |  " << square[9] <<
endl;
   cout << "\t\t\t           |     |     " << endl << endl;
   cout << play_n[1] <<" -(X) \n"<< play_n[2] <<" -(0)" << endl << endl;
 }
```

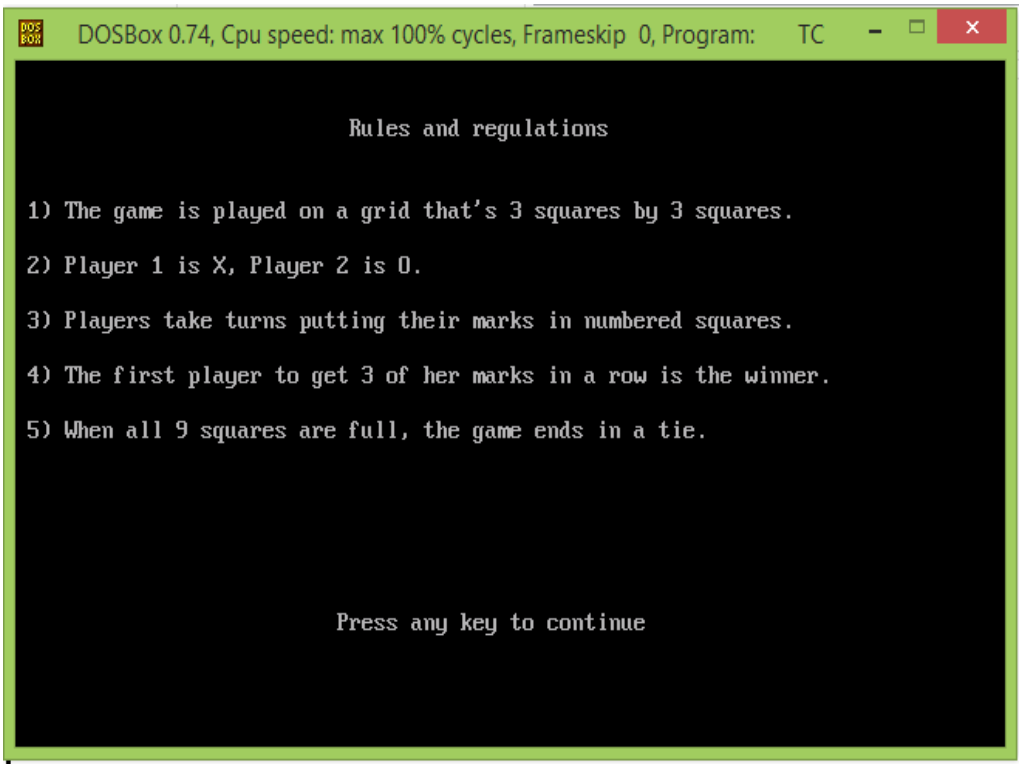
//End of the program for “Tic-Tac-Toe” game

SCREENSHOTS

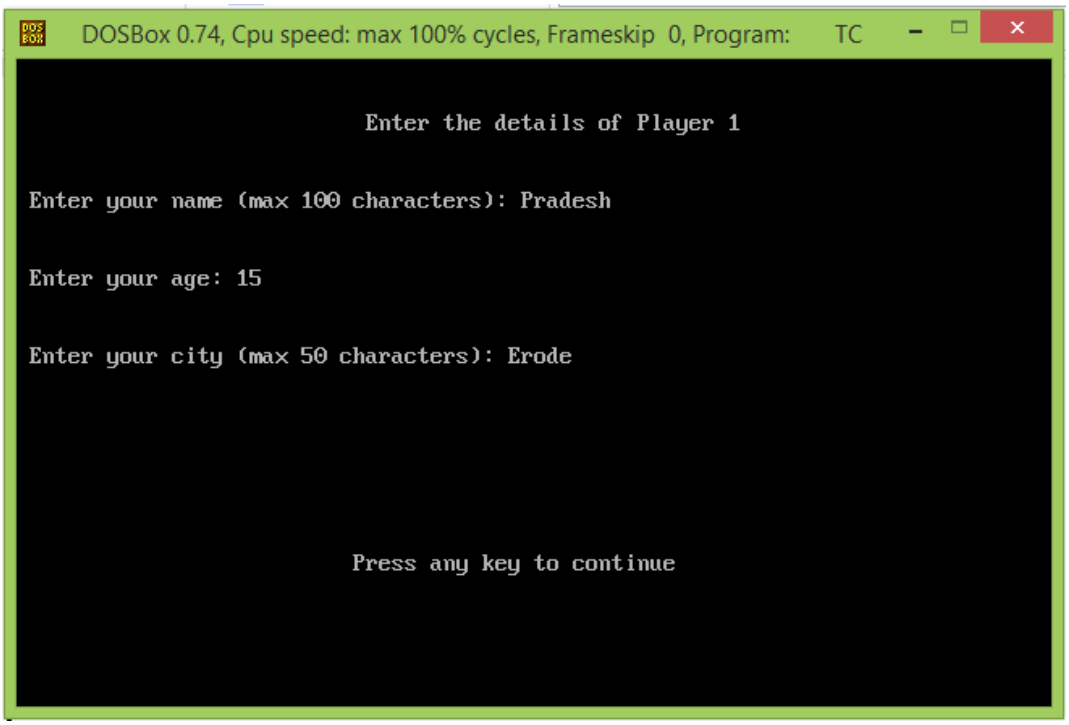
1. Home page



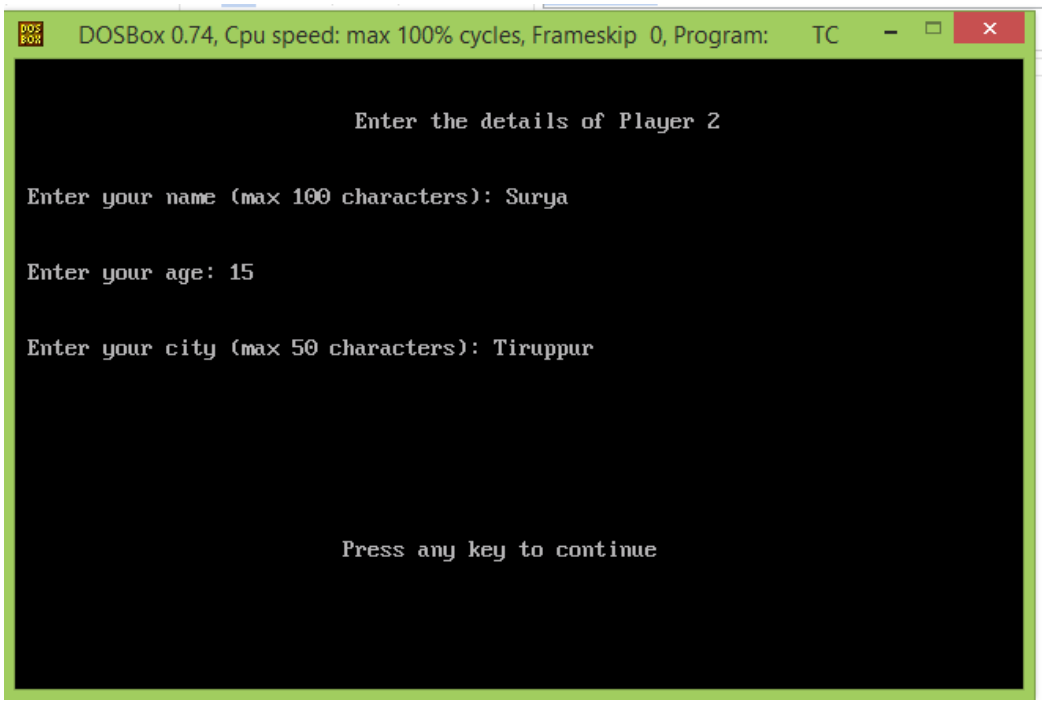
2. Rules Window



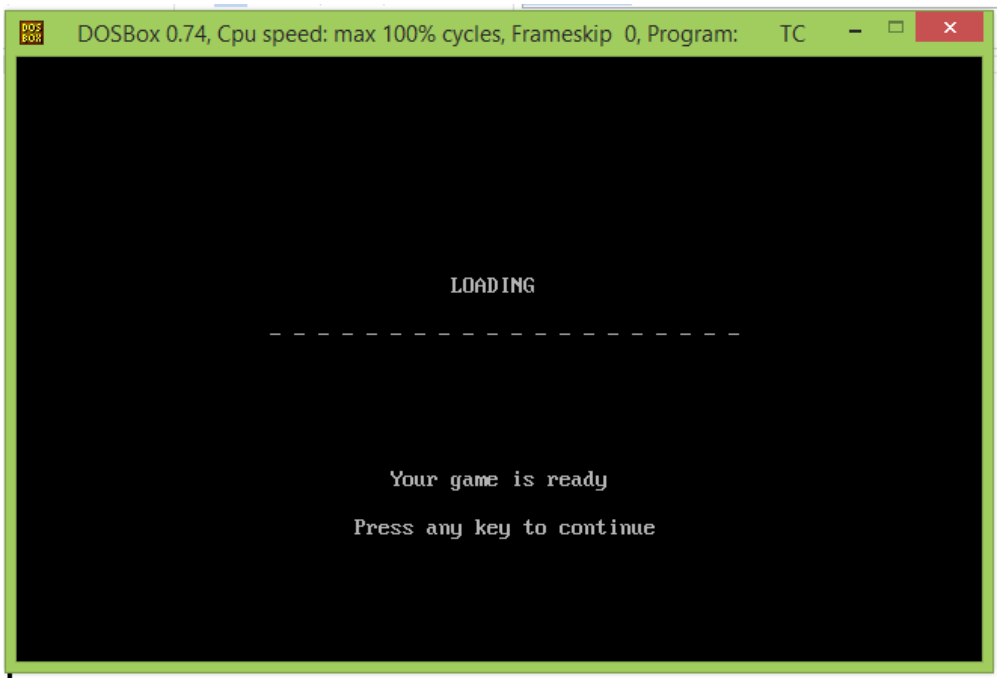
3. Details of Player 1



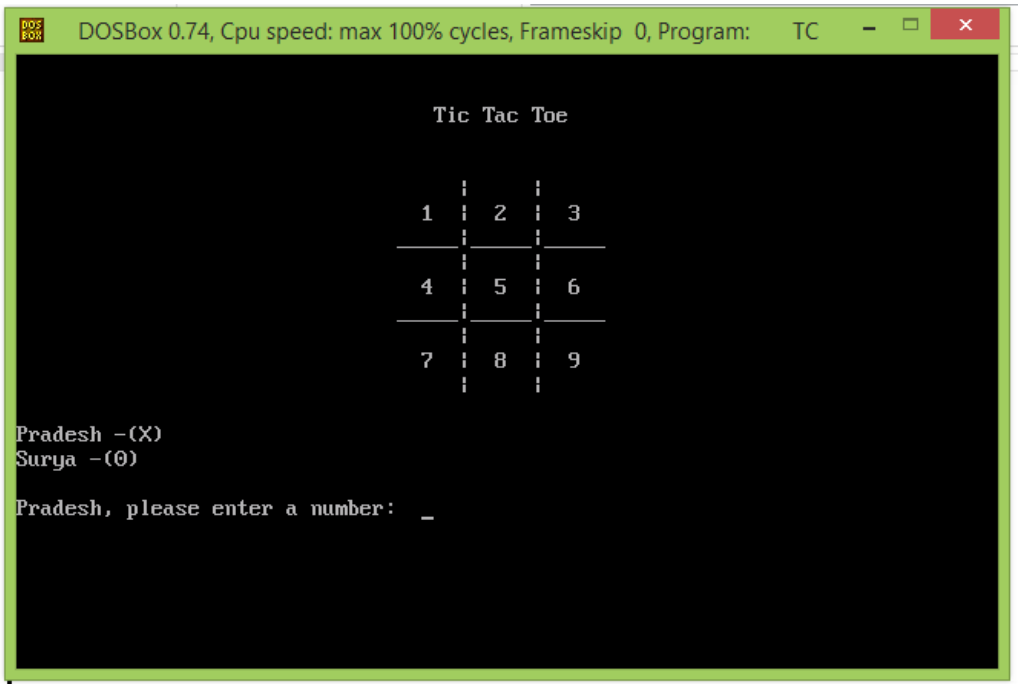
4. Details of Player 2



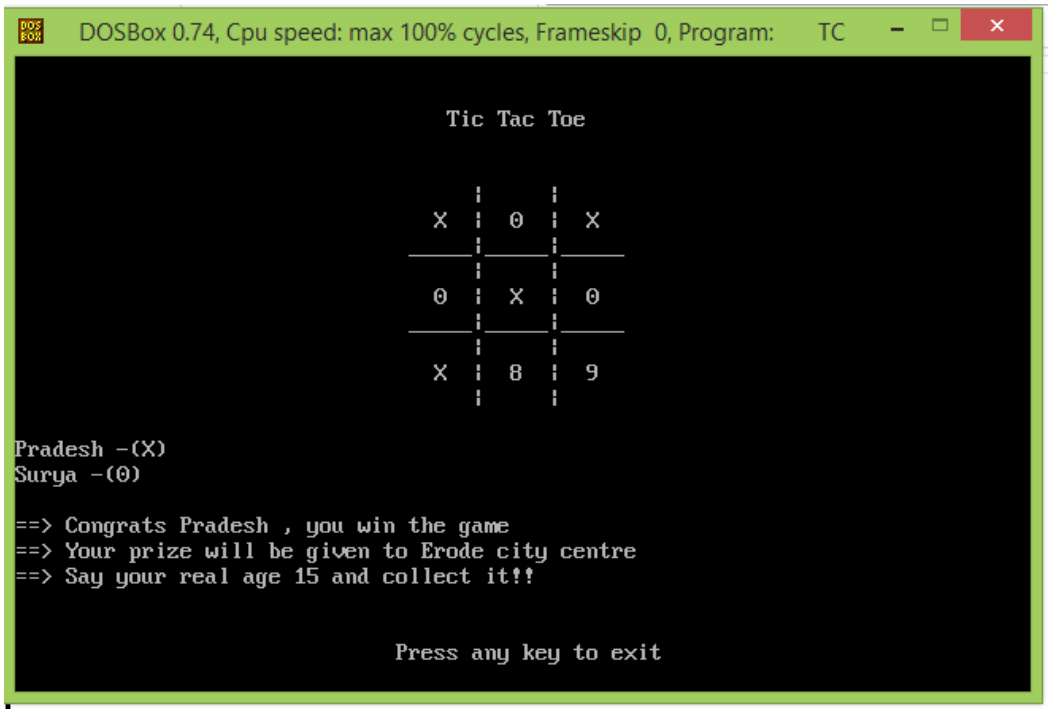
5. Loading Screen



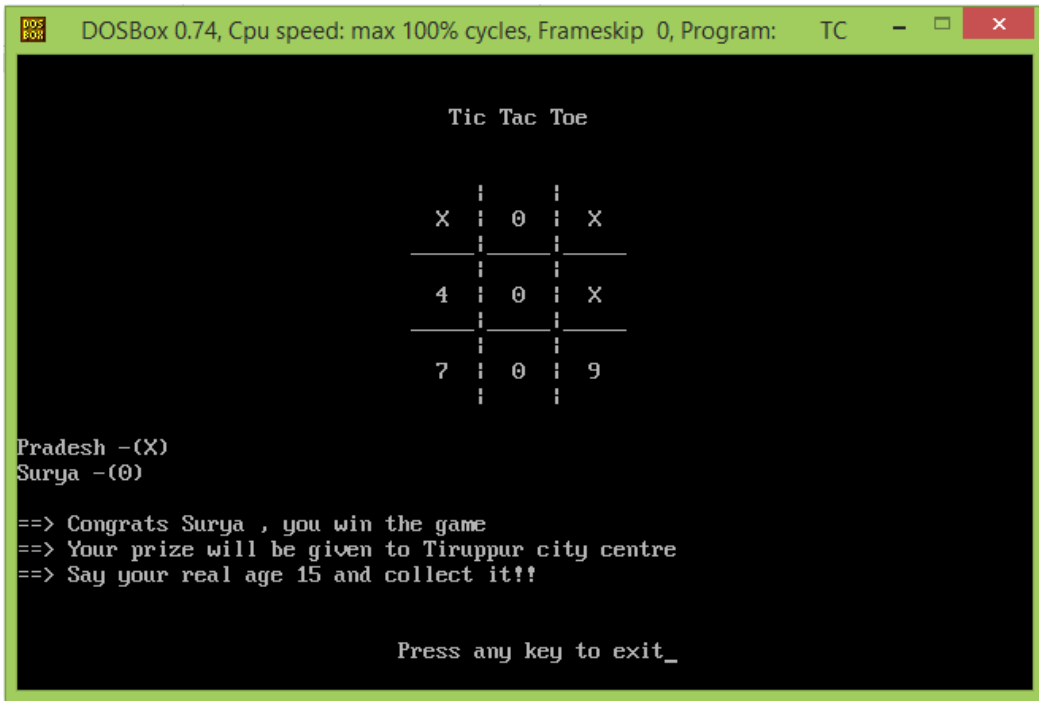
6.Game Screen



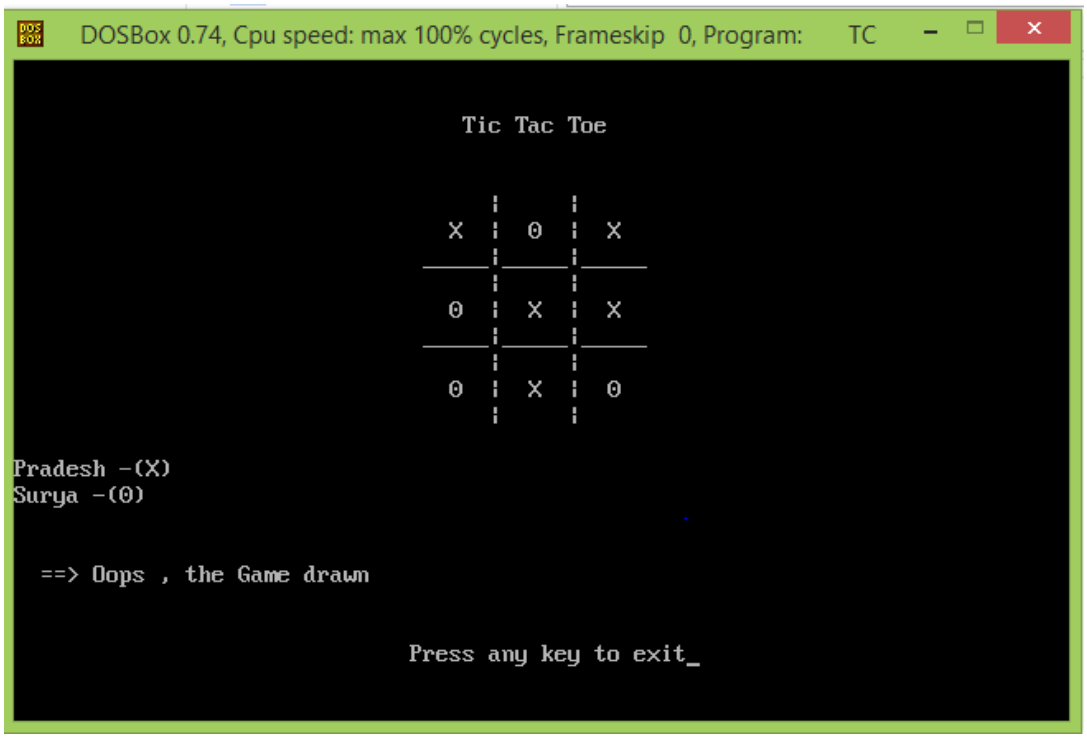
7. Player 1 won



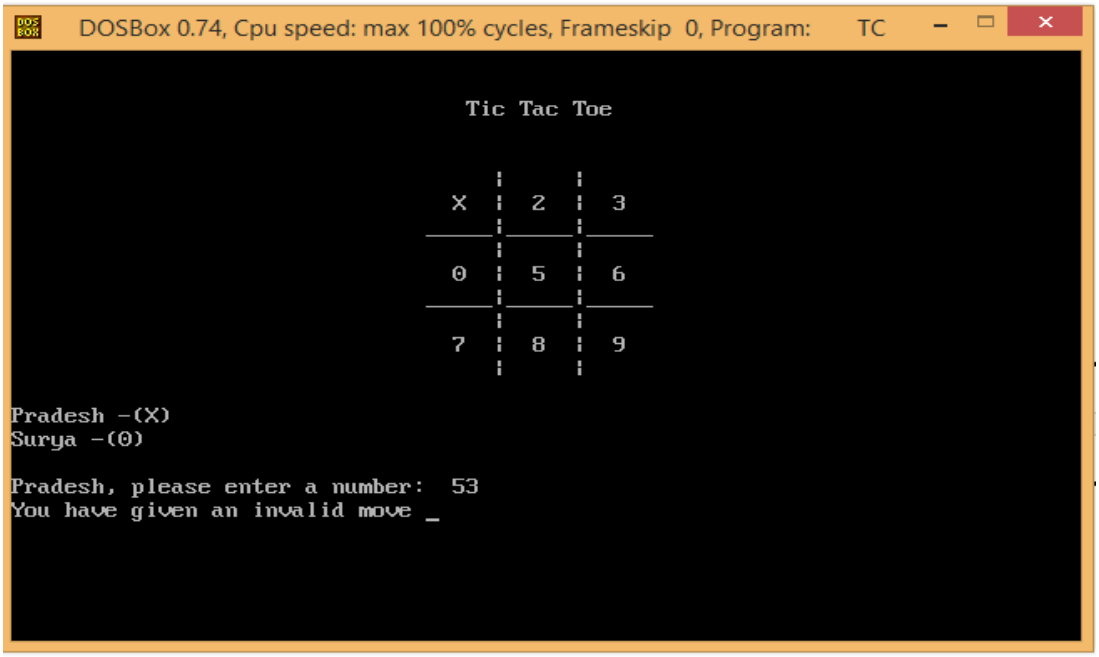
8. Player 2 won



9. Game Drawn



10. Invalid Move



REFERENCES

- ✓ Computer Science with C++ - Sumita Arora
- ✓ www.slideshare.net
- ✓ www.cppforschool.com
- ✓ www.cplusplus.com
- ✓ www.wikipedia.org
- ✓ www.isocpp.org
- ✓ www.codeacademy.com
- ✓ www.w3schools.com
- ✓ www.youtube.com
- ✓ www.programiz.com
- ✓ www.reddit.com