**/\***

**Header Files and Macros Used**

**\*/**

#include <GL/glut.h>

#include <iostream>

#include <math.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#define FONT GLUT\_BITMAP\_HELVETICA\_18

#define PI 3.1415926f

**/\***

**Global Variables**

**board: is a 3X3 matrix (2D array) where we are storing the vlaues of the**

**gameplay and using to check win/draw**

**turn: takes care of alternative turns**

**result: who wins or draw**

**over: takes account if the game is over or not**

**\*/**

int board[3][3]; // board for gameplay

int turn; // current move

int result; // Result of the game

bool over = false; // Is the game Over?

**/\***

**WriteString: Utility function to draw string**

**we use this function to write text in the window in the form of bitmap**

**and take only the font-style, string, position in screen.**

**\*/**

void WriteString(void \*font,const char s[],float x,float y)

{

unsigned int i;

glRasterPos2f(x,y); //Specifies the raster position for pixel operations

for(i=0;i<strlen(s);i++)

{

glutBitmapCharacter(font,s[i]); // renders a bitmap character using OpenGL

}

}

**/\***

**Intialize: Sets the board for Tic Tac Toe and values of the array as**

**initial value of 0, and set First players turn.**

**\*/**

void Intialize()

{

turn=1;

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

board[i][j]=0;

}

}

**/\***

**OnKeyPress: Called when any key from keyboard is pressed.**

**we have used keyboard oprions for repeating the game play.**

**It will wait for (y/n) any other key pressed will be**

**taken to termination of game.**

**\*/**

void OnKeyPress(unsigned char key,int x,int y)

{

switch(key)

{

case 'y':

if(over==true)

{ //new game

over=false;

Intialize();

}

break;

case 'n':

if(over==true)

{

exit(0); //terminate

}

break;

default:

exit(0); //terminate

}

}

**/\***

**OnMouseClick: Called when Mouse is clicked. This function when**

**called get the postion of the mouse pointer and button press.**

**Then depending upon the position it will draw either cross or**

**circle depending upon turn and availability.**

**\*/**

void OnMouseClick(int button,int state,int x,int y)

{

if(over==false && button==GLUT\_LEFT\_BUTTON && state==GLUT\_DOWN)

{

if(turn==1)

{

if(board[(y-50)/100][x/100]==0)

{

board[(y-50)/100][x/100]=1;

turn=2;

}

}

else if(turn==2)

{

if(board[(y-50)/100][x/100]==0)

{

board[(y-50)/100][x/100]=2;

turn=1;

}

}

}

}

**/\***

**DrawLines: Function to draw up the horizontal and vertical lines**

**at initialization. This is the structure of the visible board UI.**

**Here we used glBegin(GL\_LINES) which Treats each pair of consiqutive**

**vertices as an independent line segment.**

**\*/**

void DrawLines()

{

glBegin(GL\_LINES);

glColor3f(0,0,0);

glVertex2f(100,50);

glVertex2f(100,340);

glVertex2f(200,340);

glVertex2f(200,50);

glVertex2f(0,150);

glVertex2f(300,150);

glVertex2f(0,250);

glVertex2f(300,250);

glEnd();

}

**/\***

**DrawCircle: Utility function to draw the circle.**

**It will take the centre of the circle, radius, smoothness count**

**to draw a circle at that position.**

**Here we used the glBegin(GL\_LINE\_LOOP) that Draws a connected group**

**of line segments from the first vertex to the last, then back to**

**the first.**

**\*/**

void DrawCircle(float cx, float cy, float r, int num\_segments)

{

glBegin(GL\_LINE\_LOOP);

for (int i = 0; i < num\_segments; i++)

{

float theta = 2.0f \* PI \* float(i) / float(num\_segments);//get the current angle

float x = r \* cos(theta);//calculate the x component

float y = r \* sin(theta);//calculate the y component

glVertex2f(x + cx, y + cy);//output vertex

}

glEnd();

}

**/\***

**DrawXO: Function to draw the cross and circle of Tic Tac Toe.**

**This funtion draws the correct symbol depending upon turn and**

**position predefined.**

**Here we used glBegin(GL\_LINES) to draw cross,**

**and the function DrawCircle to draw round.**

**\*/**

void DrawXO()

{

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

{

if(board[i][j]==1)

{

glBegin(GL\_LINES);

glVertex2f(50 + j \* 100 - 25, 100 + i \* 100 - 25);

glVertex2f(50 + j \* 100 + 25, 100 + i \* 100 + 25);

glVertex2f(50 + j \* 100 - 25, 100 + i \* 100 + 25);

glVertex2f(50 + j \* 100 + 25, 100 + i \* 100 - 25);

glEnd();

}

else if(board[i][j]==2)

{

DrawCircle(50 + j\*100 , 100 + i\*100 , 25 , 15);

}

}

}

}

**/\***

**CheckWinner: Function to check if there is any winner at any point.**

**This drives simply checking the horizontal, vertical and diagonal**

**grids for a series. And declear a winner if any.**

**\*/**

bool CheckWinner()

{

int i, j;

// horizontal check

for(i=0;i<3;i++)

{

for(j=1;j<3;j++)

{

if(board[i][0]!=0 && board[i][0]==board[i][j])

{

if(j==2)

{

return true;

}

}

else

break;

}

}

// vertical check

for(i=0;i<3;i++)

{

for(j=1;j<3;j++)

{

if(board[0][i]!=0 && board[0][i]==board[j][i])

{

if(j==2)

return true;

}

else

break;

}

}

// Diagonal check

if((board[0][0]!=0 && board[0][0]==board[1][1] && board[0][0]==board[2][2])

|| (board[2][0]!=0 && board[2][0]==board[1][1] && board[2][0]==board[0][2]))

return true;

return false;

}

**/\***

**IsDraw: function to check if there is draw.**

**If all the position are filled and still our CheckWinner return no**

**winner then IsDraw declears the game draw.**

**\*/**

bool IsDraw()

{

int i, j;

//bool draw;

for(i=0;i<3;i++)

{

for(j=0;j<3;j++)

{

if(board[i][j]==0)

return false;

}

}

return true;

}

**/\***

**Display: Function to display up everything. This function takes care**

**of the rpeated display upon each event. And maintains the dynamic**

**nature of the game.**

**Here we write strings using the WriteString function.**

**Draw Line of board everytime using DrawLines function.**

**Draw X and O upon an appropriate event using DrawXO function.**

**Then Check for winner everytime using CheckWinner function.**

**Finally decide Draw or winner using IsDraw function.**

**\*/**

void Display()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glClearColor(1, 1, 1, 1);

glColor3f(0, 0, 0);

if(turn == 1)

WriteString(FONT, "Player1's turn", 100, 30);

else

WriteString(FONT, "Player2's turn", 100, 30);

DrawLines();

DrawXO();

if(CheckWinner() == true)

{

if(turn == 1)

{

over = true;

result = 2;

}

else

{

over = true;

result = 1;

}

}

else if(IsDraw() == true)

{

over = true;

result = 0;

}

if(over == true)

{

WriteString(FONT, "Game Over", 100, 160);

if(result == 0)

WriteString(FONT, "It's a draw", 110, 185);

if(result == 1)

WriteString(FONT, "Player1 wins", 95, 185);

if(result == 2)

WriteString(FONT, "Player2 wins", 95, 185);

WriteString(FONT, "Do you want to continue (y/n)", 40, 210);

}

}

**/\***

**Reshape: Function to reshape callback. This is called when we**

**or create or modify the window. This is usefull in maintaining the**

**user behaviour and quickly revert back to our working zone.**

**Everytime we are maintaining the viewport and matrixmode.**

**\*/**

void Reshape(int x, int y)

{

glViewport(0, 0, x, y);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(0, x, y, 0, 0, 1);

glMatrixMode(GL\_MODELVIEW);

}

**/\***

**Main Driver Function: This Do the basic initialization and calls**

**the approprite function to start their basic utility.Finally it runs**

**the Game Loop which keeps on runnig till some one terminate it from**

**inside function or forcebly.**

**\*/**

int main(int argc, char \*\*argv)

{

Intialize();

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_RGB|GLUT\_SINGLE);

glutInitWindowPosition(550,200);

glutInitWindowSize(300,350);

glutCreateWindow("LETS PLAY: Tic Tac Toe");

glutReshapeFunc(Reshape); //sets the reshape callback for the current window

glutDisplayFunc(Display);

glutKeyboardFunc(OnKeyPress); //sets the keyboard callback for the current window

glutMouseFunc(OnMouseClick); //sets the mouse callback for the current window

glutIdleFunc(Display);

glutMainLoop();

return 0;

}

**National Institute of Technology Silchar**

**Project On OpenGL**

**Tittle: Tic Tac Toe Game based on OpenGL using C++ Language.**

**Submitted by:**

**Kumar Gaurav**

**Sagar Kar**

**Arujit Pradhan**