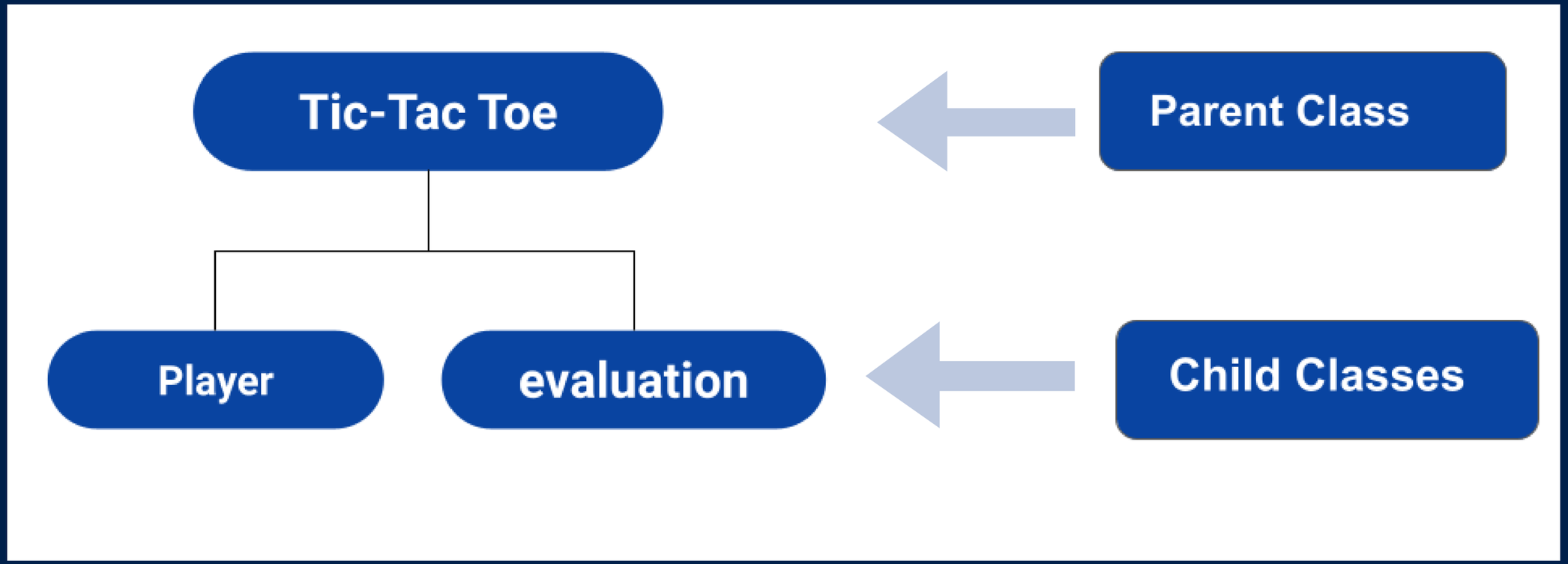
Asip Group

Members

Ashutosh pandey 21/11/EC/039
Shobhit Choudhry 21/11/EC/043
Ishu Malik 21/11/EC/026
Priyanshu Ganwani 21/11/EC/032

Submission to
Dr.Dhirendra Kumar


Tic-Tac Toe
Parent Class
Player
evaluation
Child Classes

TicTac-Toe Class (base class)

Tic Tac Toe class is the base class /parent class that contains the main matrix in which the game is played as well as the evaluation parameters that are later used to evaluate the win,loss,or draw conditions. it also keeps track of number of turns in the game. all of the variables are static so that they can be accessed by other classes but not updated.

```
protected:
    // evaluation factors
    static int diag1, diag2, col[3], row[3], x, y;
    // matrix
    static char arr[3][3];

public:
    static int turn;
```

This class also contain 2 important functions

1. Print_matrix funtion - As the name suggests its the function that is used to print the tic tac toe pattern on the console after each turn.

```
void print_matrix()
{
    cout << "\n";
    cout << " " << arr[0][0] << "  |  " << arr[0][1] << "  | " << arr[0][2] << endl;
    cout << "___ | "
         << "___ | "
         << "___ \n\n";
    cout << " " << arr[1][0] << "  |  " << arr[1][1] << "  | " << arr[1][2] << endl;
    cout << "___ | "
         << "___ | "
         << "___ \n\n";
    cout << " " << arr[2][0] << "  |  " << arr[2][1] << "  | " << arr[2][2] << endl;
    cout << "    | "
         << "    | "
         << "    \n\n";
}
```

2. Computer_choice - this function is utilised when the player plays in 1 player mode. this functons makes a move on its own in a random manner and on the basis of current game position.

```
int computer_choice()
{
    static vector<int> moves = {1, 2, 3, 4, 5, 6, 7, 8, 9};

    int val;

    for (int k = 0; k < 2; k++)
    {
        (k == 0) ? (val = -2) : (val = 2);
        // checks wheter there is any 2 or -2 in row
        for (int i = 0; i < 3; i++)
        {
            if (row[i] == val)
            {
                for (int j = 0; j < 3; j++)
                {
                    if (arr[i][j] == ' ')
                    {
                        return 3 * i + (j + 1);
                    }
                }
            }
        }

        // checks wheter there is any 2 or -2 in col
        for (int i = 0; i < 3; i++)
        {
            if (col[i] == val)
            {
                for (int j = 0; j < 3; j++)
                {
                    if (arr[j][i] == ' ')
                    {
                        return 3 * j + (i + 1);
                    }
                }
            }
        }
```

```
        // checks wheter there is any 2 or -2 in diag1
        if (diag1 == val)
        {
            for (int i = 0; i < 3; i++)
            {
                if (arr[i][i] == ' ')
                {
                    return 3 * i + i + 1;
                }
            }
        }

        // checks wheter there is any 2 or -2 in diag2
        if (diag2 == val)
        {
            for (int i = 0; i < 3; i++)
            {
                if (arr[i][2 - i] == ' ')
                {
                    return 3 * i + (2 - i) + 1;
                }
            }
        }
    }
    // if there is no 2 or -2 we choose any random element fron the moves vector
    srand(time(0));

    int random = rand() % moves.size();
    val = moves[random];
    moves.erase(moves.begin() + random);

    return val;
```

Evaluation class (child class)

this class is one of the 2 child classes . it inherits the Tic Tac Toe class privately thus is capable of accesing the matrix as well as the evalutaion favtors present in the Tic Tac Toe class.

there are 2 functions in the class -

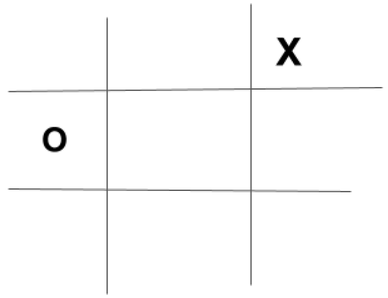
1. evaluate - this funciton is resposible for evealuating the current possition in tjhe matrix and return true if anybody wins or not. it takes 1 input that signifies whether its 1 player mode or 2 palyer mode.

```
public:
    bool evaluate(int mode)
    {
        if (diag1 == 3 or diag1 == -3 or diag2 == 3 or diag2 == -3 or row[x] == 3 or row[x] == -3 or col[y] == 3 or col[y] == -3)
        {
            if (mode == 2)
                (turn % 2 == 0) ? cout << "palyer 1 won!\n\n" : cout << "player 2 won!\n\n";
            else
                (turn % 2 == 0) ? cout << "You won!\n\n" : cout << " computer won!\n\n";
            return true;
        }
        return false;
    }
```

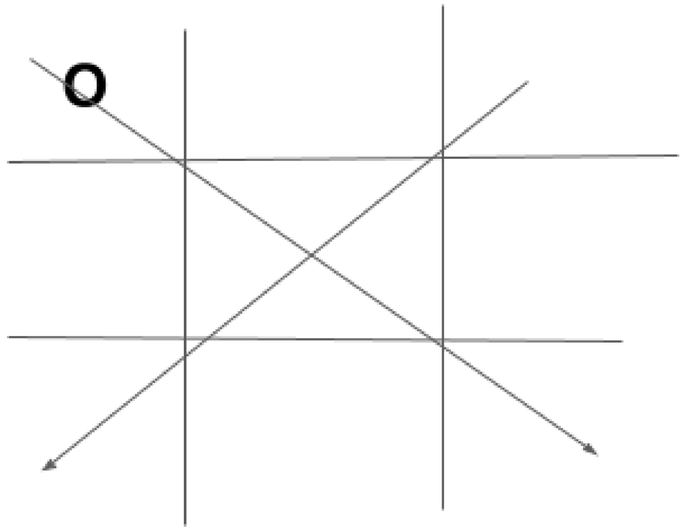
(const char [16])"player 2 won!\n\n"

COLUMNS

1
-1
0



ROWS	-1	0	1



Diag 2=0

Diag 1=1

2. update - this function updates all the evaliation factors accodingly.

```
// updates the evaluation factors
void update()
{
    if (y == x)
    {
        (turn % 2 == 0) ? diag1++ : diag1--;
    }
    if (x + y == 2)
    {
        (turn % 2 == 0) ? diag2++ : diag2--;
    }
    (turn % 2 == 0) ? col[y]++ : col[y]--;
    (turn % 2 == 0) ? row[x]++ : row[x]--;
}
```

Player class (Child class)

Like the other class it also privately inherits the Tic Tac Toe class to get access to the matrix.

```
public:
    void set_symbol(char c)
    {
        symbol = c;
    }
```

player class have a private variable named symbol that stores the sybmol that belong to a particular player (mainly X and O).

this class contains 2 functions

1. set symbol - this sets the symbol variable.

2. make_move - there are 2 make move functions.

1st make_move function donot take any input and it is called when the game is played in 2 player mode.

```
void make_move()
{
start2:
    int val;
    cout << "Enter box number - ";
    cin >> val;

    x = (val - 1) / 3;
    y = (val - 1) % 3;

    if (val > 9 or arr[x][y] == 'X' or arr[x][y] == 'O')
    {
        cout << "Retry!\n";
        cout << "Either the box is filled or you entered wrong number\n";
        goto start2;
    }
    char c = symbol;

    // updating matrix
    arr[x][y] = c;
}
```

2nd make_move function take one input and it is called when the game is played in 1 player mode.

```
void make_move(int mode)
{
start:
    int val = computer_choice();

    x = (val - 1) / 3;
    y = (val - 1) % 3;

    if (arr[x][y] != ' ')
    {
        goto start;
    }

    char c = symbol;

    // updating matrix
    arr[x][y] = c;
}
```

Objects

In the main function we create the following objects

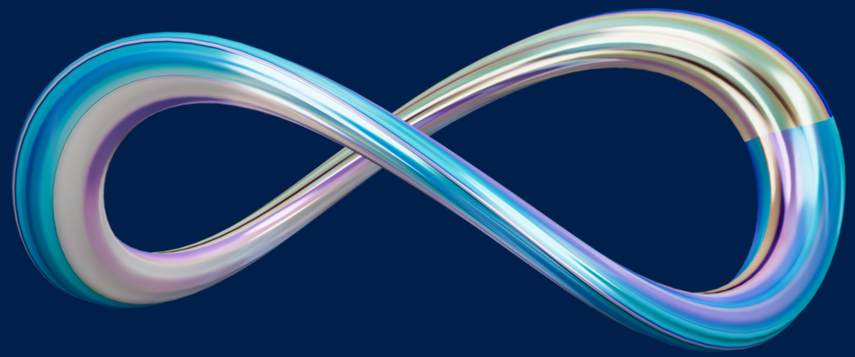
```
tic_tac_toe game;
player p1, p2;
evaluation eval;
```

game - contains the main matrix,evaluation factors and to keep track of number of turns in the game.

p1,p2 are the 2 palyers

eval object is used to call update and evaluate function during the game

Game loop



```
while (game.turn < 9)
{
    if (mode == 2)
    {
        (game.turn % 2) ? (cout << "Player 2 ") : (cout << "Player 1 ");
    }
    else
    {
        (game.turn % 2) ? (cout << "Computer's move ") : (cout << "");
    }

    if (mode == 2)
    {
        (game.turn % 2 == 0) ? (p1.make_move()) : (p2.make_move());
    }
    else
    {
        (game.turn % 2 == 0) ? (p1.make_move()) : (p2.make_move(mode));
    }

    eval.update(); // updating evaluation factors

    game.print_matrix();

    bool should_break = eval.evaluate(mode);

    if (should_break)
        break;

    game.turn++;
}

if (game.turn == 9)
{
    cout << "Draw\n\n";
}
```