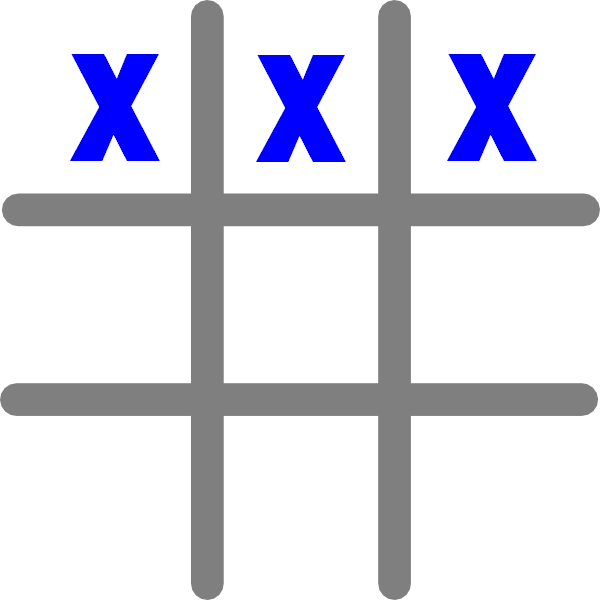
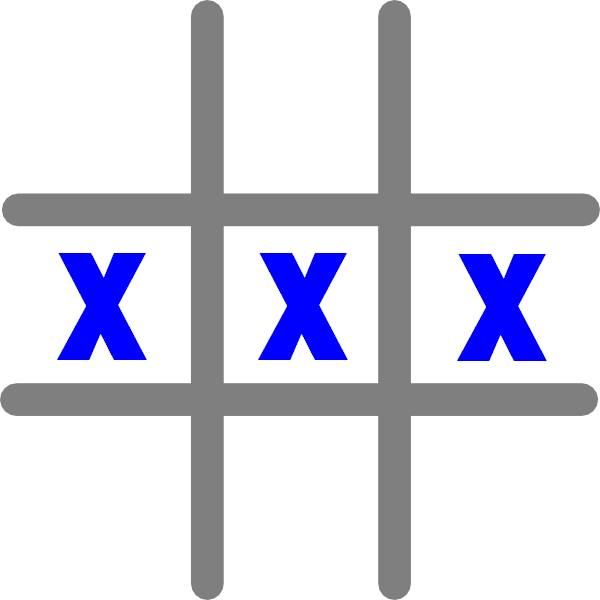
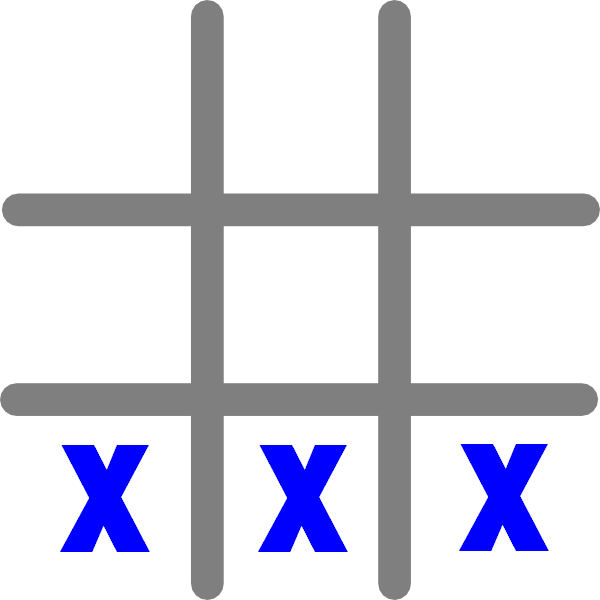
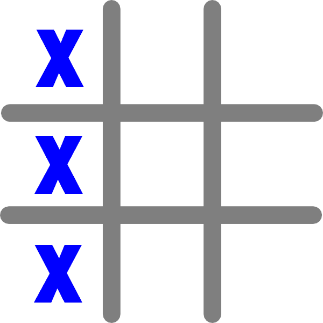
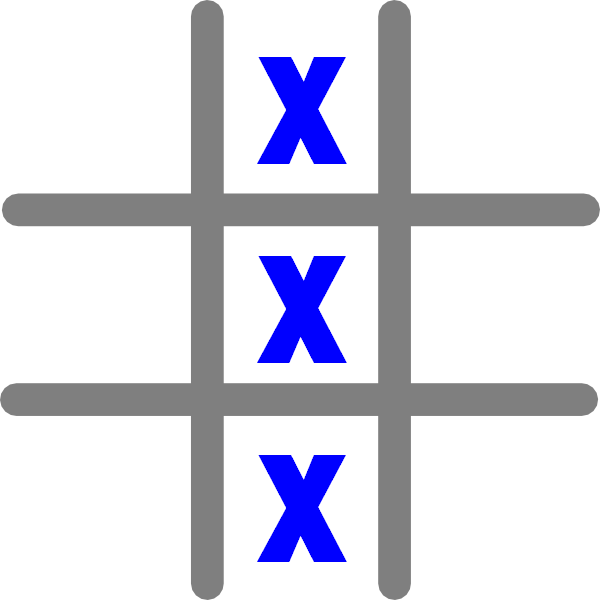
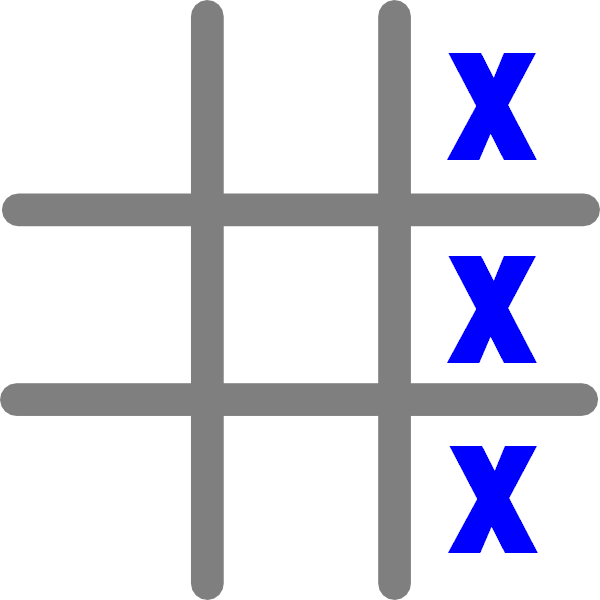
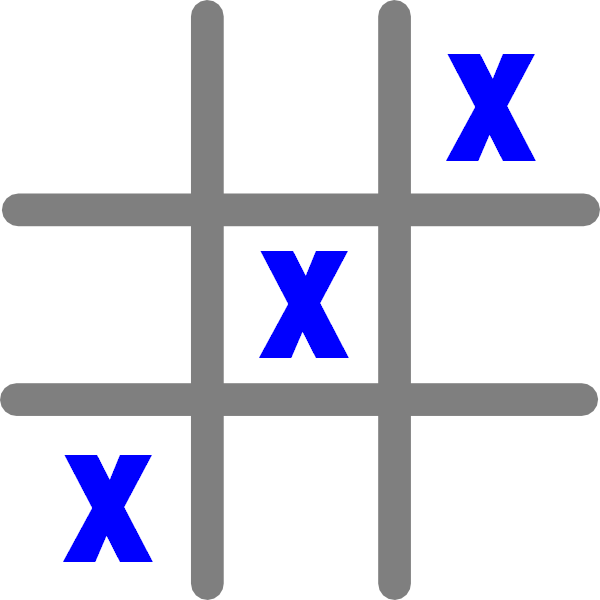
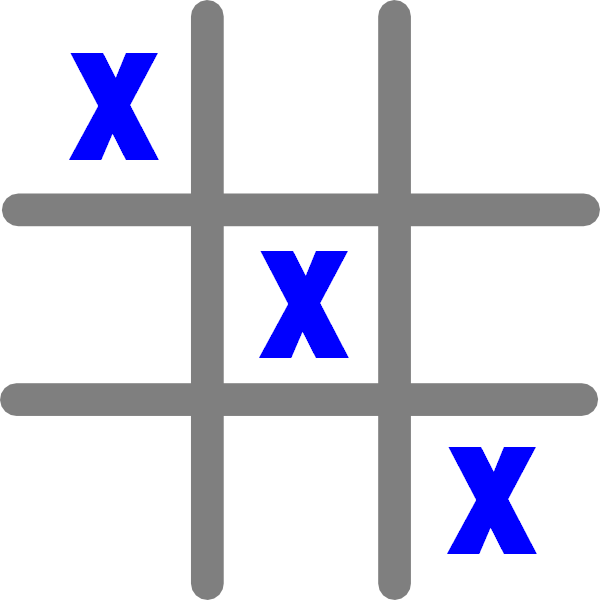
**Problem Statements and Subjects**

This project’s aim is to make user play one of the simplest games of history, Tic-Tac-Toe; against the computer. Tic-Tac-Toe is a turn-based game and it’s mechanics are simple (which are listed below).

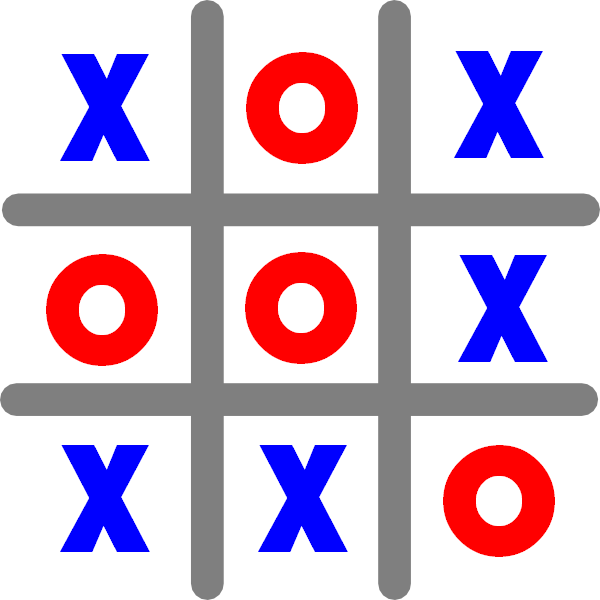
* Tic-Tac-Toe takes two players to play.
* Each player has their own character which are ‘X’ and ‘O’.
* Game takes place in a 3x3 board where each user can put their character in them. (#)
* It is a turn-based game where Player 1 puts his/her character in one of the slots, then it’s Player 2’s turn to play his/her character in one of the slots, and so on.
* Players cannot play on a slot which has been played by Player 1 or 2 already.
* The first player to make a flat slope, a vertical line or a horizontal line with his/her characters wins.
* Game has 8 different winning conditions.



* Game can also end in draw if both players fulfill all 9 slots without finishing a single flat line.



**Algorithm**

**Main() Function**

1. Ask ID from the user.
2. Call MainMenu() function.
3. Open the Scores.txt file (create one if it doesn’t exist).
4. Write all the information kept to the file.
5. Close the file.
6. Show end screen.
7. Terminate the program.

**MainMenu() Function**

1. Show the menu to the user and ask for a selection.
   1. Play 🡪 Call Play() function.
   2. Scoreboard 🡪 Call Scoreboard() function.
   3. Exit 🡪 End the MainMenu() function.

**Play() Function**

1. Empty all slots (user, AI, gameboard, flag).
2. Remind the player that he/she is ‘X’.
3. Demonstrate the gameboard with current slots.
4. Check if the game is over. Is the game over?
   1. Yes 🡪 Go to Step 13.
   2. No 🡪 Go to Step 5.
5. Ask for a slot to play from the user. Is the selected slot valid and empty?
   1. Yes 🡪 Go to Step 6.
   2. No 🡪 Go to Step 5.
6. Assign ‘X’ to the played slot.
7. Call Conditions() function with flags as parameters.
8. Check if the user has won. Did the user win?
   1. Yes 🡪 Go to Step 13.
   2. No 🡪 Go to Step 9.
9. Call Generate() function with flags as parameters.
10. Assign ‘O’ to the generated number slot.
11. Check if the AI has won. Did the AI win?
    1. Yes 🡪 Go to Step 13.
    2. No 🡪 Go to Step 12.
12. Go to Step 3.
13. Check if the user has won. Did the user win?
    1. Yes 🡪
       1. Demonstrate winning screen.
       2. Increment the counter of wins.
       3. Go to Step 17.
    2. No 🡪 Go to Step 14.
14. Check if the AI has won. Did the AI win?
    1. Yes 🡪
       1. Demonstrate losing screen.
       2. Go to Step 17.
    2. No 🡪 Go to Step 15.
15. Demonstrate draw screen.
16. Increment the counter of draws.
17. Increment the counter of plays.
18. Call the MainMenu() function.

**Generate( int ) Function**

1. Generate a random number from 1 to 9.
2. Check if the generated number slot is empty. Is it empty?
   1. Yes 🡪
      1. Assign the generated number to the AI selection.
      2. Go to Step 3.
   2. No 🡪 Go to Step 1.
3. Return the AI selection. 🡪||

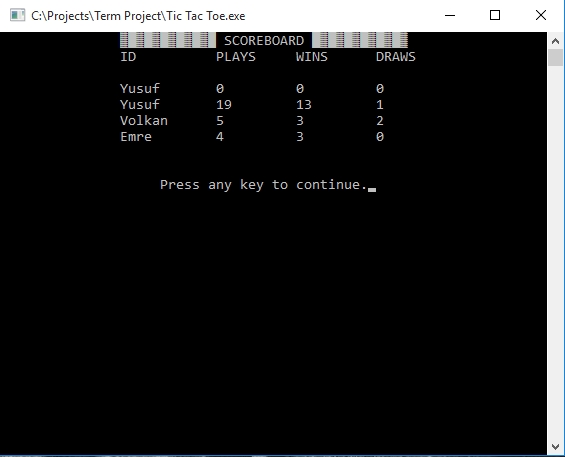
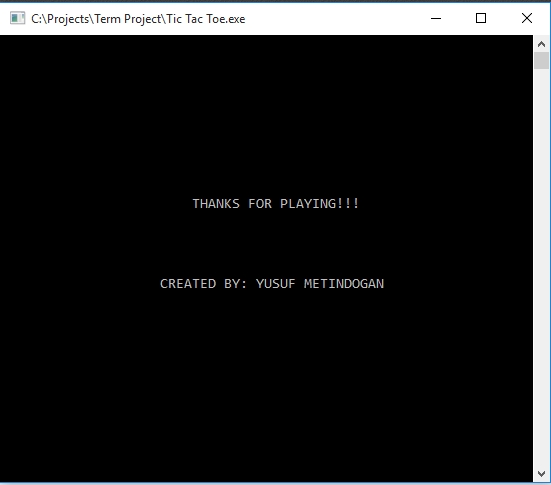
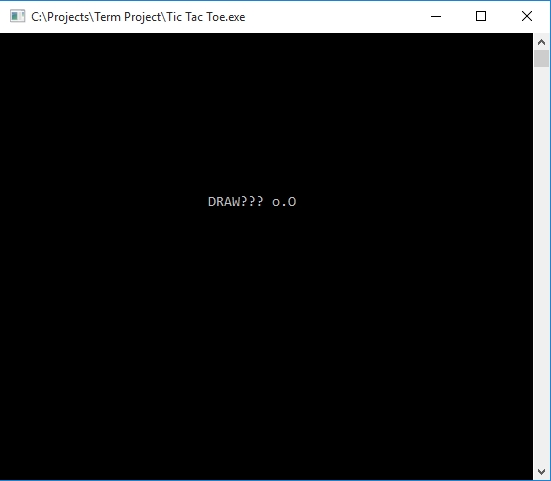
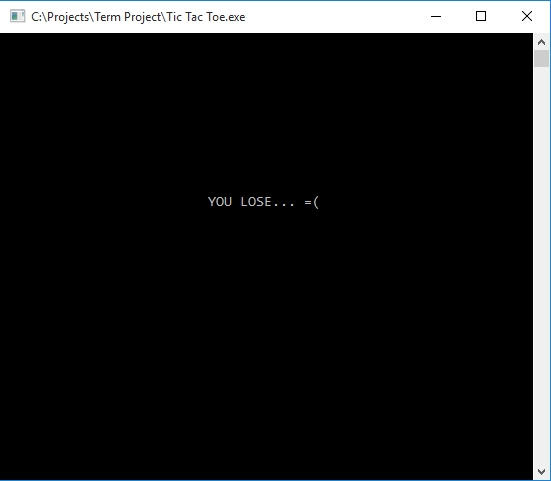
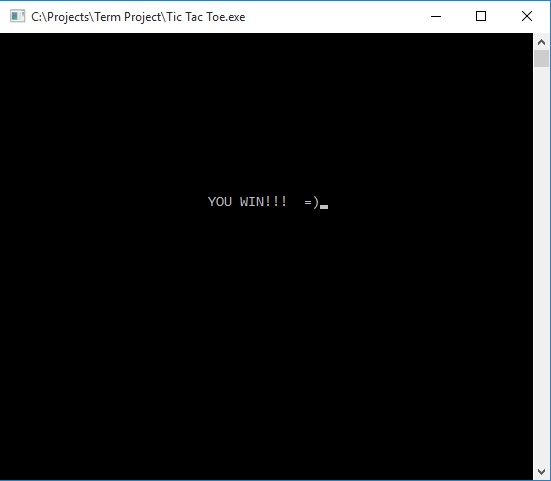
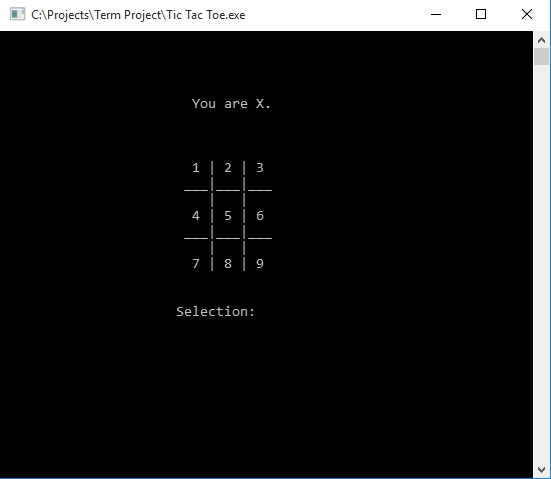
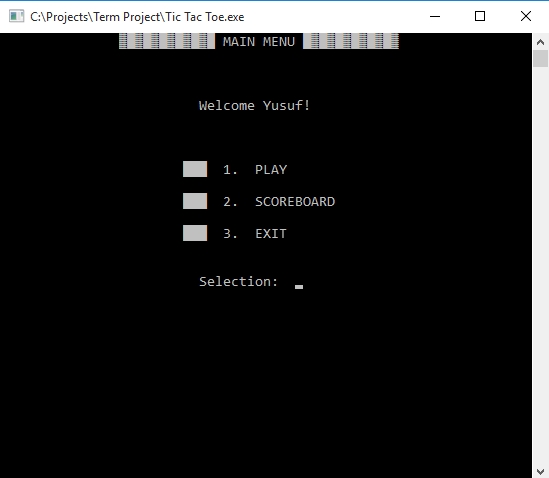
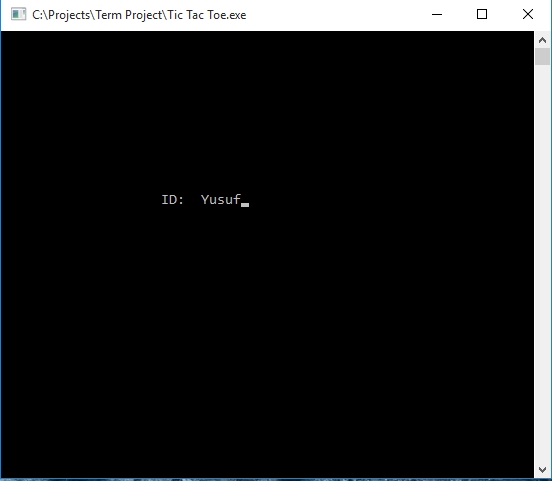
**Conditions( int ) Function**

1. Check if one of the winning conditions match with user slots. Did the user win?
   1. Yes 🡪 Return 1. 🡪||
   2. No 🡪 Go to Step 2.
2. Check if one of the losing conditions match with AI slots. Did the AI win?
   1. Yes 🡪 Return 2. 🡪||
   2. No 🡪 Go to Step 3.
3. Return 0. 🡪||

**Scoreboard() Function**

1. Demonstrate the scoreboard and listed categories.
2. Open the Scores.txt file.
3. Take all the information in it line by line.
4. Print them in order.
5. Get an input from the user.
6. Call the MainMenu() function.

**Sample Outputs**



**Source Code**

**#include <stdio.h> // For basic pre-defined functions like printf, scanf etc.**

**#include <conio.h> // For the keyboard function getch**

**#include <time.h> // For the user-defined function Wait() and pre-defined function srand()**

**#include <stdlib.h> // For the rand() function**

**#include <windows.h> // For the user-defined function Set() and pre-defined function COORD**

**FILE \*file; // Defined our file global so the Scoreboard() and main() functions can access it**

**COORD place = {0,0}; // Coordinating our set point to left-top end for Set() function**

**COORD cursor\_size;**

**typedef struct Records // A typedef structure to keep records of the plays and to make it's own alias to build it easier**

**{**

**int winCount = 0;**

**int drawCount = 0;**

**int playCount = 0;**

**char ID[10];**

**} Records;**

**Records scores;**

**Records scoreb;**

**void Wait(int mini) // This function is built to make our program delayable**

**{**

**clock\_t goal = mini + clock();**

**while (goal > clock());**

**}**

**void Set(int x, int y) // This function is built to make our program more esthetic and useful**

**{**

**place.X = x;**

**place.Y = y;**

**SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), place);**

**}**

**void MainMenu(); // Prototypes of some of our functions**

**void Play();**

**void Scoreboard();**

**int Generate(int flag[9]);**

**int Conditions(int flag[9]);**

**int main()**

**{**

**int i;**

**char bye[22] = {"THANKS FOR PLAYING!!!"}, creator[29] = {"CREATED BY: YUSUF METINDOGAN"};**

**Set(20,10);**

**printf("ID: "); scanf("%s", &scores.ID);**

**MainMenu(); // As soon as program gets an ID from user, calls the menu function**

**file = fopen("Scores.txt", "ab+"); // File operations after the user quit the game to record his scores and ID**

**fseek(file, 0, SEEK\_END); // Finds the end of text to add on it**

**fwrite(&scores, sizeof(scores), 1, file); // Adds the scores made by user**

**fclose(file); // Closing file to save it**

**system("cls");**

**Set(24,10); // End of the program**

**for (i = 0; i<=21; i++)**

**{**

**printf("%c", bye[i]);**

**Wait(50);**

**}**

**Set(20,15);**

**for (i = 0; i<=28; i++)**

**{**

**printf("%c", creator[i]);**

**Wait(50);**

**}**

**Wait(2000);**

**return 0;**

**}**

**void MainMenu() // This is the menu function**

**{**

**int select=0;**

**static int enter=0; // A static variable to print "Welcome user!" only the first time MainMenu() function called**

**while (select!=1 && select!=2 && select!=3)**

**{**

**system("cls");**

**Set(15,0);**

**printf("\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB MAIN MENU \xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2");**

**if (enter==0)**

**{**

**Set(25,4);**

**printf("Welcome %s!", scores.ID);**

**}**

**Set(23,8);**

**printf("\xDB\xDB\xDB 1. PLAY");**

**Set(23,10);**

**printf("\xDB\xDB\xDB 2. SCOREBOARD");**

**Set(23,12);**

**printf("\xDB\xDB\xDB 3. EXIT");**

**Set(25,15);**

**printf("Selection: "); scanf("%d", &select);**

**enter++;**

**if (select!=1 && select!=2 && select!=3) // To prevent the user enter a wrong number**

**{**

**Set(27,17);**

**printf("Wrong Input!");**

**Wait(1100);**

**continue;**

**}**

**else if (select==1)**

**{**

**Play();**

**}**

**else if (select==2)**

**{**

**Scoreboard();**

**}**

**else**

**{**

**break;**

**}**

**}**

**}**

**int Generate(int flag[9]) // This function is built to make AI generate a number and play that slot**

**{**

**int ai, k = 0;**

**while (k==0)**

**{**

**srand(time(NULL));**

**ai = 1 + (rand() % 8);**

**if (flag[ai-1]==1 || flag[ai-1]==2) continue; // If the slot of generated number has been played by user or AI, it regenerates a number**

**else break;**

**}**

**return ai;**

**}**

**int Conditions(int flag[9]) // This function is built to check winning conditions everytime user or AI plays a slot**

**{**

**if (flag[0]==1 && flag[1]==1 && flag[2]==1) return 1;**

**else if (flag[3]==1 && flag[4]==1 && flag[5]==1) return 1;**

**else if (flag[6]==1 && flag[7]==1 && flag[8]==1) return 1;**

**else if (flag[0]==1 && flag[3]==1 && flag[6]==1) return 1;**

**else if (flag[1]==1 && flag[4]==1 && flag[7]==1) return 1;**

**else if (flag[2]==1 && flag[5]==1 && flag[8]==1) return 1;**

**else if (flag[0]==1 && flag[4]==1 && flag[8]==1) return 1;**

**else if (flag[2]==1 && flag[4]==1 && flag[6]==1) return 1;**

**else if (flag[0]==2 && flag[1]==2 && flag[2]==2) return 2;**

**else if (flag[3]==2 && flag[4]==2 && flag[5]==2) return 2;**

**else if (flag[6]==2 && flag[7]==2 && flag[8]==2) return 2;**

**else if (flag[0]==2 && flag[3]==2 && flag[6]==2) return 2;**

**else if (flag[1]==2 && flag[4]==2 && flag[7]==2) return 2;**

**else if (flag[2]==2 && flag[5]==2 && flag[8]==2) return 2;**

**else if (flag[0]==2 && flag[4]==2 && flag[8]==2) return 2;**

**else if (flag[2]==2 && flag[4]==2 && flag[6]==2) return 2;**

**else return 0;**

**}**

**void Play() // This is the function where playing happens**

**{**

**int ui[5], ai[5], flag[9]={0,0,0,0,0,0,0,0,0}; // Empty user slots, AI slots and slots of the gameboard (0 for empty, 1 for user, 2 for AI)**

**char slot[9]={'1','2','3','4','5','6','7','8','9'}; // Demonstration slots, changing to X or O whether AI or user played**

**int i = 0, winner = 0, k = 0;**

**char win[15]={"YOU WIN!!! =)"}, lose[15]={"YOU LOSE... =("}, draw[12]={"DRAW??? o.O"};**

**while (i <= 4) // A loop to make a turn-based game**

**{**

**system("cls");**

**if (i==0) // Reminds the player that he is X and shows it only the first time**

**{**

**Set(24,4);**

**printf("You are X.");**

**}**

**Set(24,8); // Demonstration of the gameboard**

**printf("%c | %c | %c", slot[0], slot[1], slot[2]);**

**Set(23,9);**

**printf("\_\_\_|\_\_\_|\_\_\_");**

**Set(24,10);**

**printf(" | | ");**

**Set(24,11);**

**printf("%c | %c | %c", slot[3], slot[4], slot[5]);**

**Set(23,12);**

**printf("\_\_\_|\_\_\_|\_\_\_");**

**Set(24,13);**

**printf(" | | ");**

**Set(24,14);**

**printf("%c | %c | %c", slot[6], slot[7], slot[8]);**

**if (winner==1) // These conditions checks if the game is over or not before AI or user plays another slot**

**{**

**Wait(1000);**

**break;**

**}**

**else if (winner==2)**

**{**

**Wait(1000);**

**break;**

**}**

**Set(22,17);**

**printf("Selection: ");**

**while (k==0) // A loop to prevent the user play an invalid slot or a slot already played**

**{**

**Set(34,17);**

**scanf("%d", &ui[i]);**

**if (flag[ui[i]-1]==1 || flag[ui[i]-1]==2)**

**{**

**Set(16,19);**

**printf("The slot is already been played.");**

**Wait(500);**

**Set(16,19);**

**printf(" ");**

**Set(34,17);**

**printf(" ");**

**continue;**

**}**

**else if (ui[i] < 1 || ui[i] > 9)**

**{**

**Set(16,19);**

**printf("Invalid slot number.");**

**Wait(500);**

**Set(16,19);**

**printf(" ");**

**Set(34,17);**

**printf(" ");**

**continue;**

**}**

**else break;**

**}**

**slot[ui[i]-1] = 'X'; // Assigns X to the slot user played on the gameboard and 1 to game slots**

**flag[ui[i]-1] = 1;**

**if (i >= 2) // Checks if user has won the game or not (only checks after third play because winning before third play is not possible)**

**{**

**winner = Conditions(flag);**

**if (winner==1)**

**{**

**continue;**

**}**

**}**

**if (i != 4) // AI's turn to play (AI can't play on the last turn because the slot number is odd, so user starts and user ends the game)**

**{**

**ai[i] = Generate(flag);**

**slot[ai[i]-1] = 'O';**

**flag[ai[i]-1] = 2;**

**}**

**if (i >= 2) // This time checks if AI has won the game or not**

**{**

**winner = Conditions(flag);**

**if (winner==2)**

**{**

**continue;**

**}**

**}**

**i++;**

**}**

**system("cls");**

**if (winner==1) // Checks after the play loop if the game has been won, lost or ended in draw**

**{**

**Set(26,10);**

**for (i=0; i<=13; i++)**

**{**

**printf("%c", win[i]);**

**Wait(50);**

**}**

**scores.winCount++; // Increments winCount if the user has won**

**}**

**else if (winner==2)**

**{**

**Set(26,10);**

**for (i=0; i<=13; i++)**

**{**

**printf("%c", lose[i]);**

**Wait(50);**

**}**

**}**

**else**

**{**

**Set(26,10);**

**for (i=0; i<=10; i++)**

**{**

**printf("%c", draw[i]);**

**Wait(50);**

**}**

**scores.drawCount++; // Increments drawCount if the game has ended draw**

**}**

**scores.playCount++; // Increments playCount if the game has ended in any condition**

**Wait(2000);**

**MainMenu(); // Calls the menu function at the end of the game if user wants to exit, play again or check scoreboard**

**}**

**void Scoreboard() // This function takes all the information kept in Scores.txt file and prints it in an order**

**{**

**int y = 3;**

**system("cls");**

**Set(15,0);**

**printf("\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB SCOREBOARD \xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2\xDB\xB2");**

**Set(15,1);**

**printf("ID PLAYS WINS DRAWS ");**

**file = fopen("Scores.txt", "rb");**

**while (fread(&scoreb, sizeof(scoreb), 1, file)==1)**

**{**

**Set(15,y);**

**printf("%s", scoreb.ID);**

**Set(27,y);**

**printf("%d", scoreb.playCount);**

**Set(37,y);**

**printf("%d", scoreb.winCount);**

**Set(47,y);**

**printf("%d", scoreb.drawCount);**

**y++;**

**}**

**fclose(file);**

**Set(20,y+2);**

**printf("Press any key to continue.");**

**getch();**

**MainMenu();**

**}**

**Created and Written by Yusuf Metindoğan.**