**Algorithms and Data structures**

**Tic tac toe Game report**

**Callum Buttery**

**Introduction**

Tic tac toe or noughts and crosses is such a simple game that everyone has come across and played in their life, if you met someone who spoke no English and you drew a tic tac toe board on the ground, you would find a language you both speak.

Noughts and crosses is so easy to play and has a simple design, however, to make a programable game version is slightly harder than it sounds as it requires several key features that you don’t even think about if you were to play the game by hand. It has also been asked that we expand on the tic tac toe game with the ability to record moves that each player makes so that we can replay each game from the beginning, this adds further complexity to the game.

The first major features of the game is a user interface. When the user loads the game, they are presented with the options of entering 1 to load a new game, 2 to view previous games and 3 to exit the game. Another major feature is creating and displaying the gameboard. In real life you get out a pencil and draw to line parallel to each other horizontally then vertically and boom, you have a gameboard. In this version a way of storing a gameboard is required, a loop is also required to display the gameboard to the screen, this also must be updated each time a player makes a move. The user must also input their names to the game to differentiate each user. Once a move has been made, the input must be checked to ensure that the user has entered a valid board value, e.g. 0 to 8 in this case (9 squares). If a correct value has been entered, then the move can be recorded, and a save of the updated board state must be made.

After each move has been made a win checker is also called, this will look through all the board squares to see if the user has ended 3 moves across 3 game tiles that are in a line e.g. 3 tiles in one of the 3 rows or columns or diagonally across the board. If the user has won, then the game must be recorded as well as the state of the gameboard at that moment.

The final feature which I was required to implement was to replay through previous games. This meant assigning each game a game number each time a new game was loaded. As previously stated, I then recorded each move each player made and what marker was placed as well as the final board state. I could then ask the user to enter a game number which would load the game with the associated game number, this would display the first move of the game, the user then hits any key to see the next move, this process repeats until the game ends. The user will be returned to the home screen.

**Design**

Noughts and crosses could be designed using several different data structures and algorithms some more efficient than others. However, I felt that using some data structures such as stacks and linked lists made things over complicated when it came to coding. I would rather sacrifice some efficiency to make the actual creation and maintenance of the game easier, I also knew that the game wouldn’t be stressful on any system created past 1960, so this gave me good reasoning to sacrifice efficiency levels, instead of using complex data structures I used several simple arrays. Another reason I used arrays was simply because I didn’t have to worry about the draw back of not knowing the amount of data the user could enter across a game. I knew that a game cannot consist of more than 9 moves, I also knew the board wouldn’t have more than 9 tiles so I could use a fixed sized array without worry.

To record the gameboard I used an array called “gameTrackingArray” which I limited to a size of 9. Each time a game was created a game counter (which had a default value of 1) was incremented by 1, the matching gameTrackingArray space was then assigned to the current game number, this allowed me to track the number of games that had been played so that the user could replay any of their games once again by entering the game number they wanted. I also used 9 arrays [1] which were limited to the size of 18. Let’s say the gameCounter was 1, Then gameArrayOne would record the tile the user entered at array position “[1]” and the players nought or cross would be recorded at array position [2]. This would allow me to record 9 moves and 9 noughts/crosses. The reason for having 9 of these arrays (one for each game), was so that each time the game counter was incremented to a new game number e.g. from 1 to 2, there would be corresponding game array to store the moves made in each game. This made the replay function much easier to do as all I had to do was the take the game number that the user entered to replay, then find the corresponding array to display to the screen. The moves were also entered the array in the order they were made which again made it easier to display in the order the game went.

One other main array I used to develop the game was an array called gameboard which had a size of 9 as the board game would have 9 squares. This was updated each time a move was made e.g. player1 requests to enter their board marker onto tile one, the gameboard value on tile one would be updated with their board marker. This would then be displayed to the screen once again.

The final array I used to develop the game was two char arrays called playerone and playertwo which were both limited to the size of 20. This allowed me to get the player names for each game, they were also limited to size 20 to limit the size of the names the users could enter. The only reasons these arrays exists is due to C not supporting strings.

It is also worth noting that 99% of variables and arrays are declared as global values as the majority were used across several modules. This saved resources and time rather than passing values across several modules. This also made the game easier to develop and maintain.

**Future Enhancements**

This game is very simple and has loads of potential to be further developed. Several ideas I have on how to develop this further are listed below:

1. Artificial Intelligence – An AI game option could be implemented so that a player can play against an AI bot, this AI bot could also have different difficulty levels to make the game hard/easier.
2. Expanded gameboard – A larger game board could also be implemented to make the games harder/last longer. It would be possible to expand to any size of board
3. Extra players – the game could be expanded to support more than 2 players. This would only be effective if the board was expanded from a 3x3 to a 12x12 or bigger (depending on the number of players added)
4. External saves – The game could save all previous games across to a database or text file. This could then be loaded into the game when it is loaded allowing for a paper trail of how every game ever has went.

**Critical Evaluation**

I feel I have developed a good working game to the specification asked of me however there is always room for further development and expansion of the game’s efficiency and my development skills, all of which I will discuss below.

I feel the way I have been able to store the moves etc of each game of the possible 9 games has been efficient for what I was required to do, it is slightly resource hungry as there is a lot of repetitive code involved however it is very efficient in being able to access each game to replay it to the user.

As stated above, I have used a lot of repetitive code which could be replaced with a smarter coding design to make the program more efficient. I have used over 3000 lines of code to complete this project however I feel I could have done this in less than 2000. The reason I haven’t used smarter coding methods is simply because of my experience using C, before this project I feel that I didn’t have a good enough knowledge of the language to develop this without repetitive code chunks as learning the basics of C pushed me out of my comfort zone enough without using more advanced methods.

I feel my use of 1’s as crosses and 2’s as noughts has also been an improvement over the original game, as player 1 in my game is always a cross and 2 is always a nought, this makes it very easy to understand which tiles player 1 and player 2 have entered into as their board tiles are represented by their tile number.

Although as stated above, I have used more lines of code than required the resources the program is using to run throughout the game is minimal. Upon launching the game with visual studios build in debugger and performance tools the program would only use a maximum of 840KB in memory, this number didn’t increase throughout the game [2]. The CPU is also virtually at idle while running the program and has no problems what so ever [2]. The memory Heap also stayed very consistent throughout the game [3] therefore I feel this game is very efficient and runs smoothly.

**Personal Evaluation**

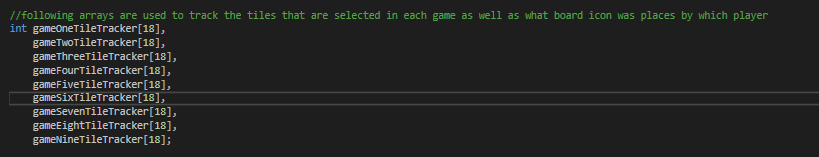
I feel throughout this piece of coursework, I have learned a huge amount. The main learning curve and by far the largest learning curve I came across was the C language itself. Before this coursework I knew so little about it and I found it quite intimidating to use due to how low level it is compared to many other languages I’ve used. To overcome this issue, I went to the modern day “library” YouTube. This gave me a foundation for the language that I could build off. I felt that the error output that C provided could be very cryptic and was often related to a different line of code from the one that was displayed as having the error. To help me fix such errors, I used Google and came across a Microsoft docs website [4] which provided me with easier to understand reasons why the error was occurring. The only other resource I used for error checking was stackoverflow [5].

The only other real challenge I had after working my way round using C was deciding on how to store the gameboard from completed games, I tried using a single array that had 81 array positions meaning I could store the moves of each of the 9 games. However, this became over complicated and hard to remember which array values should be assigned to what in later games just because the large number of array positions. So instead I came up with the idea of using the 9 different arrays to store each game.

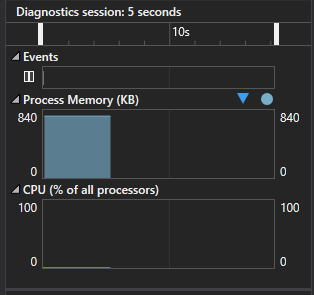
The next issue I had was updating the gameboard when replaying the games to the user, the issue I had was I wanted each move to be displayed to the screen, then the user would then hit any key to continue to see the next move. To do this I would use a printf to ask the user to “press any key to continue\n”, followed by a getchar(). I ran into the issue at this point, the move would be displayed and the message to press any key to continue would display however the game wouldn’t wait for the user to press any key, instead it would display all the moves to the screen without waiting for user input. The issue turned out to bet that the getchar(); line was reading the space generated by the \n at the end of the “press any key to continue\n” line. A simple issue yet it gave me such a sore head to try and work out why it was doing it.

I was to give myself a rating out of 10 on how I performed with this project I would give myself a 7.5 out of 10. The reasons for this is I started this project with little to no knowledge of C, so given the time frame and the challenge I had been given I feel I have done decently well to adapt to the situation and learn. However, this has not been a perfect project and it could have been done better with many more added features. The main thing I would change or learn if I had more time with this project would be how to prevent the users from entering characters into ints which in turn will crash or loop the game. During the time of me making this project I did experiment with a few ways of trying to do this however I never found a method that worked like I wanted it to, and the majority were out with my C knowledge so whatever code I was writing I didn’t fully understand so I didn’t feel comfortable putting this in a University report / program.

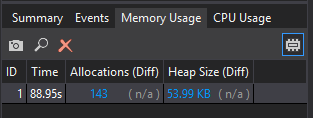
Appendix

[1] 

[2]



[3]



[4] Docs.microsoft.com. (2019). *Compiler Errors C2300 Through C2399*. [online] Available at: <https://docs.microsoft.com/en-us/cpp/error-messages/compiler-errors-1/compiler-errors-c2300-through-c2399?view=vs-2017> [Accessed 25 Mar. 2019].

[5] Stack Overflow. (2019). *Stack Overflow - Where Developers Learn, Share, & Build Careers*. [online] Available at: <https://stackoverflow.com/> [Accessed 25 Mar. 2019].

**Final Code**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

//board values

int crosses = 1;

int noughts = 2;

int emptySpaces = 0;

//used to loop through game and check that no one has won

int checkerLoop = 0;

//used in conjuction with gameTrackingArray

int gameCounter = 0;

//tracks current move

int currentMove = 1;

//following arrays are used to track the tiles that are selected in each game as well as what board icon was places by which player

int gameOneTileTracker[18],

gameTwoTileTracker[18],

gameThreeTileTracker[18],

gameFourTileTracker[18],

gameFiveTileTracker[18],

gameSixTileTracker[18],

gameSevenTileTracker[18],

gameEightTileTracker[18],

gameNineTileTracker[18];

//used to track the number of games that have been played

int gameTrackingArray[10] = { 0,0,0,0,0,0,0,0,0,0 };

//used to hold player namespace

char playerone[20] = "null";

char playertwo[20] = "null";

//gameboard array

int gameBoard[9];

//reset user to reset all game moves each time a new game is created

//used as having issue when each time a new is created, playermoves module starts one the last move of the last game, making games 2/3 moves long

int reset = 0;

//displays gameboard to the screen

void displayGameBoard()

{

//used for looping

int counter = 0;

printf("\nGame Board \n");

printf("\n!!ENTER VALUES FROM 0 TO 8!!\n");

for (counter = 0; counter < 9; counter++)

{

//following loop will create a new line after 3 steps

if (counter != 0 && counter % 3 == 0)

{

//creates new line

printf("\n");

}

printf("%4d", gameBoard[counter]);

}

}

//used to create the game board which the user will

void boardCreation()

{

//used for looping through board tiles

int counter = 0;

//used for creating playable board tiles

int playableCounter = 0;

//looping through to display all tiles

for (counter = 0; counter < 9; counter++)

{

//this will set all game board tiles to empty space

gameBoard[counter] = emptySpaces;

}

//calls displayGameBoard module and passes in the gameBoard

displayGameBoard();

}

//method used to check if the tile the user has entered is free

void inputChecker(int playerSelection, int currentMove)

{

//if the tile enterd is a 0 (emptySpaces) then we enter the if statement

if (gameBoard[playerSelection] == emptySpaces)

{

//if current move is equal to 1 then this means the current player has the board icon crosses

//will only enter the if statement if the player is crosses

if (currentMove == 1)

{

//setting board tile to crosses

gameBoard[playerSelection] = crosses;

//displays updated gameboard with the players updated move

displayGameBoard();

}

//if current move is equal to 2 then this means the current player has the board icon noughts

//will only enter the if statement if the player is not crosses

else

{

//setting board tile to noughts

gameBoard[playerSelection] = noughts;

//displaying updated gameboard with the players updated move

displayGameBoard();

}

}

//if the player has entered a tile which is already taken then we will enter this else statement

else

{

//if the board tile the user wants to enter into isn't free this loop will enter and ask for fresh input, then recall the inputChecker module with an updated playerSelection value

while (checkerLoop == 0)

{

//if currentMove is equal to 1, this means the player is playerone which is crosses

if (currentMove == 1)

{

//requests input from the user again

printf("Enter a free tile number %s", playerone);

scanf("%d", &playerSelection);

//checks fresh input

if (gameBoard[playerSelection] == emptySpaces)

{

//setting board tile to crosses

gameBoard[playerSelection] = crosses;

//displaying updated gameboard with the players updated move

displayGameBoard();

checkerLoop = 1;

}

}

//if the currentMove is equal to anything other than one we enter the else statement, this means the player is playertwo which is noughts

else

{

//requests input from the user again

printf("Enter a free tile number %s", playertwo);

scanf("%d", &playerSelection);

//checks fresh input

if (gameBoard[playerSelection] == emptySpaces)

{

//setting board tile to noughts

gameBoard[playerSelection] = noughts;

//displaying updated gameboard with the players updated move

displayGameBoard();

checkerLoop = 1;

}

}

}

}

}

//used to create the names of the players that will be playing the game

void playerCreation()

{

//increment game counter as a new game is being made

gameCounter = gameCounter + 1;

//receives player 1 name from user

printf("\nPlayer one please enter your name: \n");

//stores player 1 name to playerone variable

scanf("\n%s", playerone);

//displays playerone name in %s

printf("Player one name: %s\n", playerone);

//recieves player 2 name from user

printf("Player two please enter your name: \n");

//stores player 2 name to playertwo variable

scanf("%s", playertwo);

//displays playertwo name in %s

printf("\nPlayer two name: %s\n", playertwo);

//if the player has entered names for the player variables, we can then move on to playing the game

if (playerone == "null" || playertwo == "null" || playerone == "" || playertwo == "")

{

printf("Unable to receive player names, please reload the game and try again");

return;

}

else

{

//calls game board creation method as the users have entered valid names

boardCreation();

}

}

//used to allow the user to access previous games

viewPreviousGames()

{

int selectedGameToReplay = 0;

int previousGamesCounter = 0;

//if game recorder position 1 is 0 then the user hasn't played any games since launch, so we return them to the home screen again

if(gameTrackingArray[1] == 0)

{

//display error message

printf("\nPlease play a game first so we can record it for you to replay it!\n");

//return home

main();

}

//for loop will loop through the gameTrackingArray which holds the number of games that have been played

for (previousGamesCounter = 0; previousGamesCounter <= 8; previousGamesCounter++)

{

//if the value held in gameTrackingArray isn't 00000000 then we can display it, if it is 00000000 then it is a a blank save so we do not want to display it to the screen

if (gameTrackingArray[previousGamesCounter] != 00000000)

{

//displaying game number to the screen

printf("\n\nGame: %d\n\n", gameTrackingArray[previousGamesCounter]);

}

}

//requests the user enters a game to replay

printf("Please enter a game you want to replay: \n");

scanf("%d", &selectedGameToReplay);

//if game number is 1 then enter if statement

if (selectedGameToReplay == 1)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("\n\n\n\n\n\n\n\n\n");

printf("\n\nStarting from move one\n");

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameOneTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

printf("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameOneTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameOneTileTracker[Array Position] is set to gameOneTileTracker[users nought or cross]

gameBoard[gameOneTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameOneTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 2)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("Starting from move one");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameTwoTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\n\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\n\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameTwoTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameTwoTileTracker[Array Position] is set to gameTwoTileTracker[users nought or cross]

gameBoard[gameTwoTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameTwoTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 3)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("Starting from move one\n");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameThreeTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameThreeTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameThreeTileTracker[Array Position] is set to gameThreeTileTracker[users nought or cross]

gameBoard[gameThreeTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameThreeTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 4)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("Starting from move one\n");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameFourTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFourTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameFourTileTracker[Array Position] is set to gameFourTileTracker[users nought or cross]

gameBoard[gameFourTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameFourTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 5)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("Starting from move one\n");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameFiveTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameFiveTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameFiveTileTracker[Array Position] is set to gameFiveTileTracker[users nought or cross]

gameBoard[gameFiveTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameFiveTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 6)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("Starting from move one\n");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameSixTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSixTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameSixTileTracker[Array Position] is set to gameSixTileTracker[users nought or cross]

gameBoard[gameSixTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameSixTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 7)

{

//recreates and clears game board

boardCreation();

printf("Starting from move one\n");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameSevenTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameSevenTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameSevenTileTracker[Array Position] is set to gameSevenTileTracker[users nought or cross]

gameBoard[gameSevenTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameSevenTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 8)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("Starting from move one\n");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameEightTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameEightTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameEightTileTracker[Array Position] is set to gameEightTileTracker[users nought or cross]

gameBoard[gameEightTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameEightTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

//if game number is 1 then enter if statement

if (selectedGameToReplay == 9)

{

//recreates and clears game board

boardCreation(gameBoard);

printf("Starting from move one\n");

//used when the user is prompted below to enter any key to continue, variable has no other purpose other than to store this input

if (gameNineTileTracker[1] != 0)

{

printf("Player One Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[0]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[3] != 0)

{

printf("Player Two Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[2]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[5] != 0)

{

printf("Player One Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[4]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[7] != 0)

{

printf("Player Two Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[6]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[9] != 0)

{

printf("Player One Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[8]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[11] != 0)

{

printf("Player Two Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[10]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[13] != 0)

{

printf("Player One Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[12]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

printf("Please enter any key to see next move\n");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[15] != 0)

{

printf("Player Two Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[14]] = noughts;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

if (gameNineTileTracker[17] != 0)

{

printf("Player One Move");

//the position of game board held in gameNineTileTracker[Array Position] is set to gameNineTileTracker[users nought or cross]

gameBoard[gameNineTileTracker[16]] = crosses;

//displays updated gameboard to the screen

displayGameBoard();

//requests the user enters a key to continue playing through the game

puts("\nPlease enter any key to see next move");

//holds the key input the user enters, this variable is only here to stop the code from executing until a key is entered, the variable has no other purpose

getchar();

}

else

{

printf("\nGame has ended, Please press any key to return to the home screen");

getchar();

main();

}

//loops through every array position to check if they have all been filled with a value other than 0

//if all values don't equal 0 that means thats every tile on the board has been filled up

for (int counter = 0; counter < 18; counter++)

{

//

int AllPositionsPlayed = 1;

if (gameNineTileTracker[counter] != 0)

{

AllPositionsPlayed = AllPositionsPlayed + 1;

if (AllPositionsPlayed == 18)

{

printf("\nGame has ended in a draw\n");

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

else

{

printf("Please press any ket to return to the home screen");

getchar();

main();

}

}

}

}

//method used to stop code execture and exit the game

//declared as an int as exit fuction within the method has to have an int value passed into it

int stopping()

{

//prints that game is exiting

printf("\nExiting game\n");

//code to exit

//exit function requires a in value which is why it has a 0

exit(0);

}

// s user to game

//Gets initial user decision input

void welcomeScreen()

{

//load choice will be set to a value the user enters

int loadchoice = 0;

//possible choices the user can select on launch

int possiblechoice1 = 1;

int possiblechoice2 = 2;

int possiblechoice3 = 3;

//welcome to game message

printf("\n Welcome to noughts and crosses! \n");

/\*the following will run until the user enters a valid option

1) load new gamelll

2) view pervious games

3) exit game

\*/

printf("\nPlease enter \n1)To load a new game\n2)To view previous games \n3)To exit\n");

scanf("\n%d", &loadchoice);

//enter if the user hasn't entered a valid choice

if (loadchoice != 1 && loadchoice != 2 && loadchoice != 3)

{

//loop to insure the user has entered a valid choice

while (loadchoice != 1 || loadchoice != 2 || loadchoice != 3)

{

//requests the user enters a valid choice

printf("\nPlease enter a valid choice option\n");

scanf("%d", &loadchoice);

}

}

//if the user has entered 1 then a new game will be loaded

if (loadchoice == possiblechoice1)

{

//states which choice the user has picked

printf("\nchoice 1 loaded \n");

//module call

playerCreation();

}

//if the user has entered value 2 then the viewPreviousGames module will be called

if (loadchoice == possiblechoice2)

{

//if gameTrackingArray[1] is 0 then this means no games have been played since the user has loaded the software

//meaning me have nothing to present to the user so we will return them to the home screen

if (gameTrackingArray[1] != 0)

{

//calls viewPreviousGames method

viewPreviousGames();

}

else

{

//tells user no games have been created and will return them to the home screen

printf("\nPlease play a game first so we can record it for you to play!");

main();

}

}

//if the user has entered value 3 then the exit game module is called

if (loadchoice == possiblechoice3)

{

//module call

//will stop program execution;

stopping();

}

}

//used to record the number of games that have been played

//will be called once a user has won a game

gameRecorder()

{

//used to hold the current game counter positon

int position = gameCounter;

//sets the current game position in the array to the current game number

gameTrackingArray[position] = position;

//calls on main

main();

}

//Module used to check if any player has won the game

void winChecker()

{

//CHECKS FOR FIRST ROW

//checks first row for crosses

if (gameBoard[0] == 1 && gameBoard[1] == 1 && gameBoard[2] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks first row for noughts

if (gameBoard[0] == 2 && gameBoard[1] == 2 && gameBoard[2] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

//CHECKS FOR SECOND ROW

//checks second row for crosses

if (gameBoard[3] == 1 && gameBoard[4] == 1 && gameBoard[5] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks second row for noughts

if (gameBoard[3] == 2 && gameBoard[4] == 2 && gameBoard[5] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

//CHECKS FOR THIRD ROW

//checks third row for crosses

if (gameBoard[6] == 1 && gameBoard[7] == 1 && gameBoard[8] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks third row for noughts

if (gameBoard[6] == 2 && gameBoard[7] == 2 && gameBoard[8] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

//CHECKS FOR FIRST COLUMN

//checks first column for crosses

if (gameBoard[0] == 1 && gameBoard[3] == 1 && gameBoard[6] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks first column for noughts

if (gameBoard[0] == 2 && gameBoard[3] == 2 && gameBoard[6] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

//CHECKS FOR SECOND COLUMN

//checks second column for crosses

if (gameBoard[1] == 1 && gameBoard[4] == 1 && gameBoard[7] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks second column for noughts

if (gameBoard[1] == 2 && gameBoard[4] == 2 && gameBoard[7] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

//CHECKS FOR THIRD COLUMN

//checks third column for crosses

if (gameBoard[2] == 1 && gameBoard[5] == 1 && gameBoard[8] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks third column for noughts

if (gameBoard[2] == 2 && gameBoard[5] == 2 && gameBoard[8] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

//CHECKS FOR LEFT DIAGONAL WIN

//checks for left diagonal win for crosses

if (gameBoard[0] == 1 && gameBoard[4] == 1 && gameBoard[8] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks for left diagonal win for noughts

if (gameBoard[0] == 2 && gameBoard[4] == 2 && gameBoard[8] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

//CHECKS FOR RIGHT DIAGONAL WIN

//checks right diagonal win for crosses

if (gameBoard[2] == 1 && gameBoard[4] == 1 && gameBoard[6] == 1)

{

printf("\n\nCongratulation %s you have won!", playerone);

gameRecorder();

}

//checks right diagonal win for noughts

if (gameBoard[2] == 2 && gameBoard[4] == 2 && gameBoard[6] == 2)

{

printf("\n\nCongratulation %s you have won!", playertwo);

gameRecorder();

}

}

//used to announce that the game has ended in a draw

void drawModule()

{

printf("\n\n!!!!game has ended in a draw!!!!\n\n");

gameRecorder();

}

//used to allow players to make moves throughout the game

//will call on inputer checker module

void Playermoves()

{

//called to check is game has ended

int GameOver = 0;

//assigned to work out which player goes next

int nextmove = 0;

//what tile the player picks

int playerSelection = 0;

int counter = 0;

int alreadyAssigned = 1;

//used to work out which player is currently making a move

int currentMove = 1;

//used to count the number moves that have been

int moveCounter = 0;

//displays which user is noughts and which is crosses

//player one is always assinged to crosses

printf("\n\nYou are crosses %s \n", playerone);

int player1values = crosses;

//player two is always assigned to noughts

printf("\nYou are noughts %s \n", playertwo);

int player2values = noughts;

//displays that player one is first

printf("\nYou are first %s\n", playerone);

nextmove = crosses;

//requests the user enters a tile

printf("\nEnter tile number %s", playerone);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 1;

moveCounter = moveCounter + 1;

//calls on inputChecker which checks the user hasn't input into a tile which already contains input

/\* passes in:

1) the gameboard in its current statement

2) the currentMove (which players turn it is)

3) the tile the user has tried to enter into \*/

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[1] = crosses;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[1] = crosses;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[1] = crosses;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[1] = crosses;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[1] = crosses;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[1] = crosses;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[1] = crosses;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[1] = crosses;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[0] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[1] = crosses;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playertwo);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 2;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[3] = noughts;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[3] = noughts;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[3] = noughts;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[3] = noughts;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[3] = noughts;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[3] = noughts;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[3] = noughts;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[3] = noughts;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[2] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[3] = noughts;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playerone);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 1;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[5] = crosses;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[5] = crosses;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[5] = crosses;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[5] = crosses;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[5] = crosses;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[5] = crosses;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[5] = crosses;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[5] = crosses;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[4] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[5] = crosses;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playertwo);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 2;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[7] = noughts;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[7] = noughts;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[7] = noughts;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[7] = noughts;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[7] = noughts;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[7] = noughts;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[7] = noughts;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[7] = noughts;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[6] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[7] = noughts;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playerone);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 1;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[9] = crosses;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[9] = crosses;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[9] = crosses;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[9] = crosses;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[9] = crosses;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[9] = crosses;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[9] = crosses;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[9] = crosses;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[8] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[9] = crosses;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playertwo);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 2;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[11] = noughts;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[11] = noughts;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[11] = noughts;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[11] = noughts;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[11] = noughts;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[11] = noughts;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[11] = noughts;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[11] = noughts;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[10] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[11] = noughts;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playerone);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 1;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[13] = crosses;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[13] = crosses;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[13] = crosses;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[13] = crosses;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[13] = crosses;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[13] = crosses;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[13] = crosses;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[13] = crosses;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[12] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[13] = crosses;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playertwo);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 2;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[15] = noughts;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[15] = noughts;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[15] = noughts;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[15] = noughts;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[15] = noughts;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[15] = noughts;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[15] = noughts;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[15] = noughts;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[14] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[15] = noughts;

}

winChecker();

//requests the user enters a tile

printf("\nEnter tile number %s", playerone);

scanf("%d", &playerSelection);

//current move 1 is always assigned to playerone/crosses

currentMove = 1;

moveCounter = moveCounter + 1;

inputChecker(playerSelection, currentMove);

//second value is the value of the game the software is on e.g. game 1 of 9 in this case

if (gameCounter == 1)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameOneTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameOneTileTracker[17] = crosses;

}

if (gameCounter == 2)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameTwoTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameTwoTileTracker[17] = crosses;

}

if (gameCounter == 3)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameThreeTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameThreeTileTracker[17] = crosses;

}

if (gameCounter == 4)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFourTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFourTileTracker[17] = crosses;

}

if (gameCounter == 5)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameFiveTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameFiveTileTracker[17] = crosses;

}

if (gameCounter == 6)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSixTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSixTileTracker[17] = crosses;

}

if (gameCounter == 7)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameSevenTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameSevenTileTracker[17] = crosses;

}

if (gameCounter == 8)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameEightTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameEightTileTracker[17] = crosses;

}

if (gameCounter == 9)

{

//sets the first position of gameOneTileTracker to the tile the user wants to enter their board value into

gameNineTileTracker[16] = playerSelection;

//sets the position of gameOneTileTracker to the board value the user wants to place e.g. noughts or crosses

gameNineTileTracker[17] = crosses;

}

winChecker();

//if the code has got to this point and the game still hasn't been won this must mean the game has ended in a draw

drawModule();

}

int main()

{

int resetOption = 0;

//if game counter is 9 then this means all game arrays have been filled and the game will need to restart to clear them

if (gameCounter < 9)

{

//calls modules to start game

welcomeScreen();

Playermoves();

}

else

{

//if the user hasn't entered a 1 or a 2 then we will reask for input

while (resetOption != 1 || resetOption != 2)

{

printf("Game memory full, please enter 1 to do nothing or 2 to reset games");

scanf("%d", &resetOption);

}

//if 1 entered then the user wants to do nothing so we will return to home screen

if (resetOption == 1)

{

main();

}

//if the user has entered 2 then the program will stop by calling the stopping function

else

{

stopping();

}

return 0;

}

}