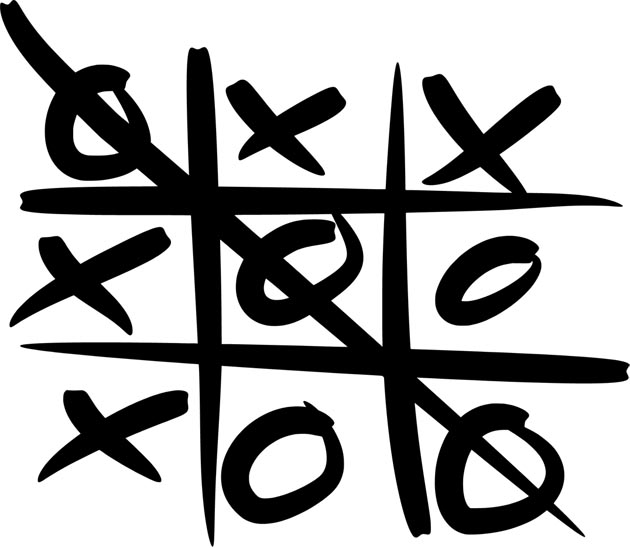
**PROJECT 1**

TIC TAC TOE



Anh Vu

CSC 5 – 46023

July 18th, 2014

**TABLE OF CONTENTS**

**Cover**……………………………………………………………………………………….…….1

**Introduction**……………………………………………………………………………………...3

**Rules**…………………………………………………………………………………...…3

**Design Details**.................................................................................................................................4

**INTRODUCTION**

Tic-Tac-Toe is a game in which two players, represented as either X or O, mark a 3 x 3 grid until one player wins. The objective of the game is to mark 3 spaces that are horizontally, vertically, or diagonally adjacent to one another before the other player does.

**Rules –**

* Each player will take turns marking the grid. No player will mark the grid more than once at a time.
* You must only mark the empty spaces in the grid.
* Many times you will find that neither player wins. This game is considered a draw, or a cat’s game.

**Instructions –**

* The program will randomly decide which player (‘X’ or ‘O’) will play first
* Each space in the 3 x 3 grid will be represented by a number. Please select the numbered space in which you would like to place your mark. Your mark will be placed on the grid after each time you input.
* The game will continue to run until 1) one person wins, or 2) all spaces are occupied and no one wins.
* If you wish to see your total scores, see the text file names "scores"
* Enjoy the game!

**DESIGN DETAILS**

**Approach -**

In order to successfully code Tic-Tac-Toe, I first played a game and broke the game down in small steps in order to fully understand the sub components of the game and to ultimately implement them in a code. Below are the steps I initially drafted:

1. Draw and output a 3x3 board.
2. Designate each player as either 'X' or 'O'
3. Allow users to take turns and to mark the board with their respective marks.
4. Conditions for the game to end
5. Ways which players can win
6. Keeping scores and outputting them so user can see

**Overview -**

At the end of my project, I had developed a game that had encompassed all of the steps that I had originally thought of. (Step 1) My program outputs a board each time a player makes a move. The board I created consists of a set of variables that represents each space. An example of my board is below:

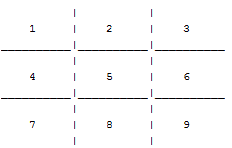


Figure 1. My tic-tac-toe board. In order to make a move, each player is required to input the number where he or she wants to place his or her mark.

If the space is already taken up, the program prompts the user to input another number. (Step 2) My program also designates Player 1 as 'X' and Player 2 as 'O' and randomly chooses which player will be able to play first. Then, the program switches between each player from one move to the next. (Step 3) When the player chooses a number to place his mark, the number in the grid will be substituted with the mark. An example is shown below:

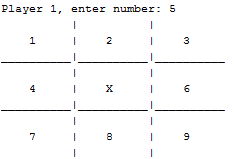


Figure 2. When player 1 (X) chooses number 5, that specific space is then replaced with player 1's mark.

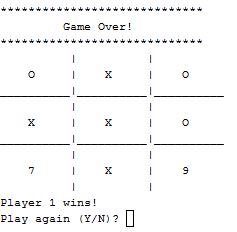
(Step 4) Of course, when playing any game there are conditions to signal the end of the game. The game can end in nine different ways. Three ways the game can is if there are the same marks adjacent to one another in each of the three horizontal rows of the board (e.g. spaces 1, 2, and 3). Another three ways the game can end is the same as the aforementioned condition, but it applies to each of the three vertical columns. Two ways that will end the game is if the same marks are adjacent to one another diagonally. The last way the game can end is if all the spaces are taken up. In this case, neither player wins and the game is considered a draw. Below shows an example of an output to signal the end of the game. Also note that I prompt the user to decide whether he or she wants to play again.

Figure 3. Player 1 wins because he/she placed 3 marks adjacent to one another vertically before the other player could. The program also asks the user if he or she wants to play again.

C:\Users\user\Desktop\Capture.PNG(Step 5) As you can see in *figure 3*, the program is able to determine who wins. The condition that determines who wins is which player is able to end the game first. (Step 6) When each game ends, the scores for each player are subsequently added depending on which player wins. Scores are tallied and output on a text file titled "scores." Below is a sample of a the text file:

Figure 4. Player 1 has one win while Player 2 has none.

The details of my program will be further discussed throughout my report.

**RESEARCH**

**FLOWCHART**

**CODE**

1 /\*

2 \* File: main.cpp

3 \* Author: Anh Vu

4 \* Purpose: Project 1 - Tic Tac Toe!

5 \* Created on July 11, 2014, 11:15 AM

6 \*/

7

8 //System Level Libraries

9 #include <iostream>

10 #include <ctime>

11 #include <cstdlib>

12 #include <fstream>

13 using namespace std;

14

15 //User Libraries

16

17 //Global Constants

18

19 //Function Prototypes

20 void **displayGrid**(char []); //Displays tic tac toe board

21 void **takeTurn**(char[], bool); //Places appropriate mark on board

22 bool **gameOver**(char[]); //Determines whether or not game is over

23 bool **winner**(char[], bool); //Determines winner

24

25

26 //Execution Begins Here:

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

28 //Declare and Initialize Variables

29 const int SIZE=10;

30 char grid[SIZE]={'x','1','2','3','4','5','6','7','8','9'}; //used for tic tac toe board

31 bool p1; //Used to keep track of score

32 unsigned short score1=0; //Calculate score for player 1

33 unsigned short score2=0; //Calculate score for player 2

34 char again; //Whether players want to play again

35 bool over; //Determines if game is over

36

37 //Welcome player and output which player is what mark

38 cout<<**"**Welcome to Tic-Tac-Toe! Who will play first will be randomly chosen...**"**;

39 cout<<**"**May the best man win!**"**<<endl;

40 cout<<**"**Player 1= X**"**<<endl;

41 cout<<**"**Player 2= O**"**<<endl<<endl;

42

43 //Decide which player plays first using random number generator

44 //Set random seed

45 srand(static\_cast<unsigned int>(time(0)));

46

47 //Determine who will go first

48 bool first=rand()%2;

49

50 //Output who will go first

51 if(first){

52 cout<<**"**Congratulations Player 1, you get to play first!**"**;

53 cout<<endl<<endl;

54 }else{

55 cout<<**"**Congratulations Player 2, you get to play first!**"**;

56 cout<<endl<<endl;

57 }

58

59 //Game Play

60 //Reiterates game until players decide to stop

61 do{

62 //Game continues until game is over

63 do{

64 displayGrid(grid); //Display Grid

65 takeTurn(grid,first); //User takes turn

66 //Switches between players 1 & 2

67 if(first){

68 first=false;

69 }

70 else{

71 first=true;

72 }

73 }while(!gameOver(grid));

74

75 //Determines if game is over & outputs game over message

76 gameOver(grid);

77 if (over=true){

78 cout<<**"\n**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***\n"**;

79 cout<<**"** Game Over! **\n"**;

80 cout<<**"**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***\n"**;

81 }

82

83 //Display final board that ends game

84 displayGrid(grid);

85

86 //Displays winner

87 winner(grid,p1);

88

89 //Keep track of scores

90 if (p1=true)

91 score1+=1;

92 else

93 score2+=1;

94

95 //Output scores to file

96 ofstream output;

97 output.open (**"**scores.txt**"**);

98 output<<**"**Player 1: **"**<<score1<<**"** wins & **"**

99 <<**"**Player 2: **"**<<score2<<**"** wins...........**"**;

100

101 if(score1<score2)

102 output<<**"**Player 1, you need to step up your game!**\n"**;

103 else if (score2<score1)

104 output<<**"**Player 2, you need to step up your game!**\n"**;

105 else

106 output<<**"**Player 1 and Player 2, you are tied.**\n"**;

107

108 //Close File

109 output.close ();

110

111 //Ask if player wants to play again

112 cout<<**"**Play again (Y/N)? **"**;

113 cin>>again;

114

115 //Reset board so user can play again

116 grid[1]='1';

117 grid[2]='2';

118 grid[3]='3';

119 grid[4]='4';

120 grid[5]='5';

121 grid[6]='6';

122 grid[7]='7';

123 grid[8]='8';

124 grid[9]='9';

125 }while((again=='Y')||(again=='y'));

126

127 //Exit Stage Right!

128 return 0;

129 }

130

131 //Function displays grid

132 void **displayGrid**(char grid[]){

133 //Board with variables at designated positions

134 cout<<**"** | | **"**<<endl;

135 cout<<**"** **"**<<grid[1]<<**"** | **"**<<grid[2]<<**"** | **"**<<grid[3]<<endl;

136 cout<<**"**\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_**"**<<endl;

137 cout<<**"** | | **"**<<endl;

138 cout<<**"** **"**<<grid[4]<<**"** | **"**<<grid[5]<<**"** | **"**<<grid[6]<<endl;

139 cout<<**"**\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_**"**<<endl;

140 cout<<**"** | | **"**<<endl;

141 cout<<**"** **"**<<grid[7]<<**"** | **"**<<grid[8]<<**"** | **"**<<grid[9]<<endl;

142 cout<<**"** | | **"**<<endl;

143 }

144

145 //Function places mark on board

146 void **takeTurn**(char grid[], bool p1){

147 unsigned short choice; //Where user wants to place mark (1,2,3...9)

148

149 //Gather Data Input

150 //Takes user's choices as long as spaces are available

151 do{

152 if(p1){

153 cout<<**"**Player 1, **"**;

154 }else{

155 cout<<**"**Player 2, **"**;

156 }

157 cout<<**"**enter number: **"**;

158 cin>>choice;

159 }while(choice>9||choice<1||grid[choice]!=(choice+'0'));

160

161 //If player one makes mark, place an X. If player 2, place O.

162 if(p1){

163 grid[choice] = 'X';

164 }

165 else{

166 grid[choice] = 'O';

167 }

168

169 }

170

171 //Determines whether game is over

172 bool **gameOver**(char grid[]){

173 //Declare Variables

174 bool over=false;

175

176 //Determine if game is over

177 //Game is over if one of the players wins, or if all the spaces are taken up

178 if((grid[1]==grid[2])&&(grid[1]==grid[3]))

179 over=true;

180 else if((grid[4]==grid[5])&&(grid[4]==grid[6]))

181 over=true;

182 else if((grid[7]==grid[8])&&(grid[7]==grid[9]))

183 over=true;

184 else if((grid[1]==grid[4])&&(grid[1]==grid[7]))

185 over=true;

186 else if((grid[2]==grid[5])&&(grid[2]==grid[8]))

187 over=true;

188 else if((grid[3]==grid[6])&&(grid[3]==grid[9]))

189 over=true;

190 else if((grid[1]==grid[5])&&(grid[1]==grid[9]))

191 over=true;

192 else if((grid[3]==grid[5])&&(grid[3]==grid[7]))

193 over=true;

194 else if ((grid[1]!='1')&&(grid[2]!='2')&&(grid[3]!='3')&&(grid[4]!='4')&&(grid[4]!='4')&&(grid[5]!='5')&&(grid[6]!='6')&&(grid[7]!='7')&&(grid[8]!='8')&&(grid[8]!='8')&&(grid[9]!='9'))

195 over=true;

196 else

197 over=false;

198

199 //Over=true, Game over

200 //Over=false, game not over

201 return over;

202 }

203

204 //Determines winner

205 bool **winner**(char grid[], bool p1){

206 if((grid[1]=='X')&&(grid[2]=='X')&&(grid[3]=='X')){

207 cout<<**"**Player 1 wins!**"**<<endl;

208 p1=true;

209 }else if((grid[1]=='O')&&(grid[2]=='O')&&(grid[3]=='O')){

210 cout<<**"**Player 2 wins!**"**<<endl;

211 p1=false;

212 }else if((grid[4]=='X')&&(grid[5]=='X')&&(grid[6]=='X')){

213 cout<<**"**Player 1 wins!**"**<<endl;

214 p1=true;

215 }else if((grid[4]=='O')&&(grid[5]=='O')&&(grid[6]=='O')){

216 cout<<**"**Player 2 wins!**"**<<endl;

217 p1=false;

218 }else if((grid[7]=='X')&&(grid[8]=='X')&&(grid[9]=='X')){

219 cout<<**"**Player 1 wins!**"**<<endl;

220 p1=true;

221 }else if((grid[7]=='O')&&(grid[8]=='O')&&(grid[9]=='O')){

222 cout<<**"**Player 2 wins!**"**<<endl;

223 p1=false;

224 }else if((grid[1]=='X')&&(grid[4]=='X')&&(grid[7]=='X')){

225 cout<<**"**Player 1 wins!**"**<<endl;

226 p1=true;

227 }else if((grid[1]=='O')&&(grid[4]=='O')&&(grid[7]=='O')){

228 cout<<**"**Player 2 wins!**"**<<endl;

229 p1=false;

230 }else if((grid[2]=='X')&&(grid[5]=='X')&&(grid[8]=='X')){

231 cout<<**"**Player 1 wins!**"**<<endl;

232 p1=true;

233 }else if((grid[2]=='O')&&(grid[5]=='O')&&(grid[8]=='O')){

234 cout<<**"**Player 2 wins!**"**<<endl;

235 p1=false;

236 }else if((grid[3]=='X')&&(grid[6]=='X')&&(grid[9]=='X')){

237 cout<<**"**Player 1 wins!**"**<<endl;

238 p1=true;

239 }else if((grid[3]=='O')&&(grid[6]=='O')&&(grid[9]=='O')){

240 cout<<**"**Player 2 wins!**"**<<endl;

241 p1=false;

242 }else if((grid[1]=='X')&&(grid[5]=='X')&&(grid[9]=='X')){

243 cout<<**"**Player 1 wins!**"**<<endl;

244 p1=true;

245 }else if((grid[1]=='O')&&(grid[5]=='O')&&(grid[9]=='O')){

246 cout<<**"**Player 2 wins!**"**<<endl;

247 p1=false;

248 }else if((grid[3]=='X')&&(grid[5]=='X')&&(grid[7]=='X')){

249 cout<<**"**Player 1 wins!**"**<<endl;

250 p1=true;

251 }else if((grid[3]=='O')&&(grid[5]=='O')&&(grid[7]=='O')){

252 cout<<**"**Player 2 wins!**"**<<endl;

253 p1=false;

254 }else

255 cout<<**"**No one wins.**"**<<endl;

256

257 //If p1=false, p1 loses and p2 wins

258 return p1;

259 }