```cpp
#include "ticTacToe.h"

/************************************************************
*   THIS METHOD IS A CONSTRUCTOR THAT SETS VARIABLES TO      *
*                    THEIR DEFAULT VALUE                     *
************************************************************/

ticTacToe::ticTacToe()
{
    board[0] = "1";
    board[1] = "2";
    board[2] = "3";
    board[3] = "4";
    board[4] = "5";
    board[5] = "6";
    board[6] = "7";
    board[7] = "8";
    board[8] = "9";

    for (int i = 0; i < 3; i++)
    {
        p1Stats[i] = 0;
        p2Stats[i] = 0;
    }

    turn = true;

    p1 = "x";
    p2 = "o";
}

/************************************************************
*   THESE FUNCTIONS SET PRIVATE METHODS                      *
*                                                            *
************************************************************/

void ticTacToe::setp1Name(string name)
{
    p1Name = name;
}

void ticTacToe::setp2Name(string name)
{
    p2Name = name;
}

void ticTacToe::setfName1(string name)
{
    fName1 = name;
}

void ticTacToe::setfName2(string name)
```

```
53  {
54      fName2 = name;
55  }
56
57  void ticTacToe::setlName1(string name)
58  {
59      lName1 = name;
60  }
61
62  void ticTacToe::setlName2(string name)
63  {
64      lName2 = name;
65  }
66
67  void ticTacToe::setp1(string marker)
68  {
69      p1 = marker;
70  }
71
72  void ticTacToe::setp2(string marker)
73  {
74      p2 = marker;
75  }
76
77  void ticTacToe::setboard(string* gameBoard)
78  {
79      for (int i = 0; i < 9; i++)
80      {
81          board[i] = *(gameBoard + i);
82      }
83  }
84
85  void ticTacToe::setp1Stats(int* stats)
86  {
87      for (int i = 0; i < 3; i++)
88      {
89          p1Stats[i] = *(stats + i);
90      }
91  }
92
93  void ticTacToe::setp2Stats(int* stats)
94  {
95      for (int i = 0; i < 3; i++)
96      {
97          p2Stats[i] = *(stats + i);
98      }
99  }
100
101 void ticTacToe::setturn(bool Turn)
102 {
103     turn = Turn;
104 }
```

```
105
106 /************************************************************
107 *   THESE FUNCTIONS GET PRIVATE METHODS                     *
108 *                                                           *
109 ************************************************************/
110
111 string ticTacToe::getp1Name()
112 {
113     return p1Name;
114 }
115
116 string ticTacToe::getp2Name()
117 {
118     return p2Name;
119 }
120
121 string ticTacToe::getfName1()
122 {
123     return fName1;
124 }
125
126 string ticTacToe::getfName2()
127 {
128     return fName2;
129 }
130
131 string ticTacToe::getlName1()
132 {
133     return lName1;
134 }
135
136 string ticTacToe::getlName2()
137 {
138     return lName2;
139 }
140
141 string ticTacToe::getp1()
142 {
143     return p1;
144 }
145
146 string ticTacToe::getp2()
147 {
148     return p2;
149 }
150
151 string* ticTacToe::getboard()
152 {
153     return board;
154 }
155
156 int* ticTacToe::getp1Stats()
```

```
157 {
158     return p1Stats;
159 }
160
161 int* ticTacToe::getp2Stats()
162 {
163     return p2Stats;
164 }
165
166 bool ticTacToe::getturn()
167 {
168     return turn;
169 }
170
171 /************************************************************
172 *   THIS FUNCTION OUTPUTS A HEADER FOR THE GAME             *
173 *                                                           *
174 ************************************************************/
175
176 void ticTacToe::outputHeader()
177 {
178     cout << " /*************************************************************\ "
          << endl;
179     cout << " *                                                           * "
          << endl;
180     cout << " *                      TIC TAC TOE                          * "
          << endl;
181     cout << " *                                                           * "
          << endl;
182     cout << " \*************************************************************/ "
          << endl << endl;
183
184     cout << " WELCOME USERS!" << endl << endl;
185     cout << " # TO GET STARTED ENTER YOUR FIRST AND LAST NAMES." << endl;
186     cout << " # ONCE YOU FIGURE OUT WHO GOES FIRST THE GAME WILL START." << endl;
187     cout << " # ENTER A NUMBER TO MAKE YOUR MOVE. " << endl << endl;
188     cout << " ENJOY THE GAME!!!" << endl << endl;
189 }
190
191 /************************************************************
192 *   THIS FUNCTION GETS THE PLAYERS' NAMES                   *
193 *                                                           *
194 ************************************************************/
195
196 void ticTacToe::getNames()
197 {
198     cout << " Player 1, Enter your first and last name >> ";
199     cin >> fName1 >> lName1;
200
201
202     cout << " Player 2, Enter your first and last name  >> ";
203     cin >> fName2 >> lName2;
```

```
204     cout << endl;
205
206     p1Name = fName1 + " " + lName1;
207     p2Name = fName2 + " " + lName2;
208 }
209
210 /************************************************************
211 *       THIS FUNCTION GETS THE PLAYER THAT GOES FIRST        *
212 *                 (ONLY FOR THE FIRST GAME)                 *
213 ************************************************************/
214
215 bool ticTacToe::whoseFirst()
216 {
217     cout << " " << fName1 << ", you will be " << p1 << " in the game!" << endl;
218     cout << " This means " << fName2 << " will be " << p2 << "." << endl << endl;
219
220     string first;
221     cout << " Choose who will go first (1 or 2) >> ";
222     cin >> first;
223     cout << endl;
224
225     if (first == "1")
226     {
227         turn = true;
228
229         cout << " " << fName1 << " goes first!" << endl << endl;
230     }
231
232     else
233     {
234         turn = false;
235
236         cout << " " << fName2 << " goes first!" << endl << endl;
237     }
238     return turn;
239 }
240
241 /************************************************************
242 *   THIS FUNCTION OUTPUTS THE GAME BOARD                    *
243 *                                                           *
244 ************************************************************/
245
246 void ticTacToe::outputBoard()
247 {
248     cout << " +---+---+---+" << endl;
249     cout << " | " << board[0] << " | " << board[1] << " | " << board[2] << " |"
          << endl;
250     cout << " +---+---+---+" << endl;
251     cout << " | " << board[3] << " | " << board[4] << " | " << board[5] << " |"
          << endl;
252     cout << " +---+---+---+" << endl;
253     cout << " | " << board[6] << " | " << board[7] << " | " << board[8] << " |"
```

```
          << endl;
254     cout << " +---+---+---+" << endl;
255     cout << endl;
256 }
257
258 /*************************************************************
259 *   THIS FUNCTION CHECKS FOR A WIN AND UPDATES STATS         *
260 *                                                            *
261 *************************************************************/
262
263 string ticTacToe::isWin()
264 {
265     if (board[0] == board[1] && board[1] == board[2])
266     {
267         if (board[0] == p1)
268         {
269             board[0] = board[1] = board[2] = "X";
270             p1Stats[0] += 1;
271             p2Stats[1] += 1;
272             return p1;
273         }
274         else if (board[0] == p2)
275         {
276             board[0] = board[1] = board[2] = "O";
277             p1Stats[1] += 1;
278             p2Stats[0] += 1;
279             return p2;
280         }
281     }
282
283     else if (board[3] == board[4] && board[4] == board[5])
284     {
285         if (board[3] == p1)
286         {
287             board[3] = board[4] = board[5] = "X";
288             p1Stats[0] += 1;
289             p2Stats[1] += 1;
290             return p1;
291         }
292         else if (board[3] == p2)
293         {
294             board[3] = board[4] = board[5] = "O";
295             p1Stats[1] += 1;
296             p2Stats[0] += 1;
297             return p2;
298         }
299     }
300
301     else if (board[6] == board[7] && board[7] == board[8])
302     {
303         if (board[6] == p1)
304         {
```

```
305             board[6] = board[7] = board[8] = "X";
306             p1Stats[0] += 1;
307             p2Stats[1] += 1;
308             return p1;
309         }
310         else if (board[6] == p2)
311         {
312             board[6] = board[7] = board[8] = "O";
313             p1Stats[1] += 1;
314             p2Stats[0] += 1;
315             return p1;
316         }
317     }
318
319     else if (board[0] == board[3] && board[3] == board[6])
320     {
321         if (board[0] == p1)
322         {
323             board[0] = board[3] = board[6] = "X";
324             p1Stats[0] += 1;
325             p2Stats[1] += 1;
326             return p1;
327         }
328         else if (board[0] == p2)
329         {
330             board[0] = board[3] = board[6] = "O";
331             p1Stats[1] += 1;
332             p2Stats[0] += 1;
333             return p2;
334         }
335     }
336
337     else if (board[1] == board[4] && board[4] == board[7])
338     {
339         if (board[1] == p1)
340         {
341             board[1] = board[4] = board[7] = "X";
342             p1Stats[0] += 1;
343             p2Stats[1] += 1;
344             return p1;
345         }
346         else if (board[1] == p2)
347         {
348             board[1] = board[4] = board[7] = "O";
349             p1Stats[1] += 1;
350             p2Stats[0] += 1;
351             return p2;
352         }
353     }
354
355     else if (board[2] == board[5] && board[5] == board[8])
356     {
```

```
357         if (board[2] == p1)
358         {
359             board[2] = board[5] = board[8] = "X";
360             p1Stats[0] += 1;
361             p2Stats[1] += 1;
362             return p1;
363         }
364         else if (board[2] == p2)
365         {
366             board[2] = board[5] = board[8] = "O";
367             p1Stats[1] += 1;
368             p2Stats[0] += 1;
369             return p2;
370         }
371     }
372
373     else if (board[0] == board[4] && board[4] == board[8])
374     {
375         if (board[0] == p1)
376         {
377             board[0] = board[4] = board[8] = "X";
378             p1Stats[0] += 1;
379             p2Stats[1] += 1;
380             return p1;
381         }
382         else if (board[0] == p2)
383         {
384             board[0] = board[4] = board[8] = "O";
385             p1Stats[1] += 1;
386             p2Stats[0] += 1;
387             return p2;
388         }
389     }
390
391     else if (board[2] == board[4] && board[4] == board[6])
392     {
393         if (board[2] == p1)
394         {
395             board[2] = board[4] = board[6] = "X";
396             p1Stats[0] += 1;
397             p2Stats[1] += 1;
398             return p1;
399         }
400         else if (board[2] == p2)
401         {
402             board[2] = board[4] = board[6] = "O";
403             p1Stats[1] += 1;
404             p2Stats[0] += 1;
405             return p2;
406         }
407     }
408
```

```
409     return " ";
410 }
411
412 /************************************************************
413 *   THIS FUNCTION PROMPTS THE USER TO TAKE THIER TURN        *
414 *                                                           *
415 ************************************************************/
416
417 void ticTacToe::makeMove(bool turn)
418 {
419     int choice;
420     if (turn)
421     {
422         cout << " " << fName1 << "(x), Make Your Move >> ";
423         cin >> choice;
424
425         board[choice - 1] = p1;
426     }
427
428     else
429     {
430         cout << " " << fName2 << "(o), Make Your Move >> ";
431         cin >> choice;
432
433         board[choice - 1] = p2;
434     }
435     cout << endl;
436 }
437
438 /************************************************************
439 *   THIS FUNCTION SWITCHES THE TURN OF THE PLAYER            *
440 *                                                           *
441 ************************************************************/
442
443 void ticTacToe::switchTurn()
444 {
445     if (turn == true) { turn = false; }
446     else { turn = true; }
447 }
448
449 /************************************************************
450 *   THIS FUNCTION RESETS THE BOARD FOR A NEW GAME TO START   *
451 *                                                           *
452 ************************************************************/
453
454 void ticTacToe::clearBoard()
455 {
456     board[0] = "1";
457     board[1] = "2";
458     board[2] = "3";
459     board[3] = "4";
460     board[4] = "5";
```

```
461     board[5] = "6";
462     board[6] = "7";
463     board[7] = "8";
464     board[8] = "9";
465 }
466 
467 /*************************************************************
468 *   THIS FUNCTION OUTPUTS THE PLAYER STATS                   *
469 *                                                            *
470 *************************************************************/
471 
472 void ticTacToe::outputStats()
473 {
474     cout << " ******Player Stats******" << endl << endl;
475     cout << " " << p1Name << endl;
476     cout << " | Wins: " << p1Stats[0] << " | Losses: " << p1Stats[1] << " |       ↵
        Draws: " << p1Stats[2] << " |" << endl << endl;
477     cout << " " << p2Name << endl;
478     cout << " | Wins: " << p2Stats[0] << " | Losses: " << p2Stats[1] << " |       ↵
        Draws: " << p2Stats[2] << " |" << endl << endl;
479 }
480 
481 /*************************************************************
482 *   THIS FUNCTION PLAYS THE TICTACTOE GAME                   *
483 *                                                            *
484 *************************************************************/
485 
486 void ticTacToe::startGame()
487 {
488     char playAgain = 'y';
489     string winner;
490     string loser;
491     bool isTie = false;
492     bool whoStarts;
493 
494     outputHeader();
495     getNames();
496     whoStarts = !whoseFirst();
497 
498     while (playAgain == 'y')
499     {
500         loser = " ";
501         cout << " GAME START!" << endl;
502         outputBoard();
503         for(int i = 0; i < 9; i++)
504         {
505             makeMove(turn);
506             outputBoard();
507             winner = isWin();
508 
509             if (winner != " ")
510             {
```

```
511                 if (winner == p1)
512                 {
513                     loser = p2;
514                     cout << " " << fName1 << " Wins!" << endl;
515                 }
516 
517                 else
518                 {
519                     loser = p1;
520                     cout << " " << fName2 << " Wins!" << endl;
521                 }
522 
523                 outputBoard();
524                 break;
525             }
526 
527             if (i == 8)
528             {
529                 p1Stats[2] += 1;
530                 p2Stats[2] += 1;
531                 cout << " IT'S A TIE!" << endl << endl;
532                 outputBoard();
533                 isTie = true;
534             }
535 
536             switchTurn();
537         }
538 
539         clearBoard();
540 
541         if (isTie)
542         {
543             turn = !whoStarts;
544             whoStarts = turn;
545             isTie = false;
546         }
547 
548         if (loser == p1) { turn = true; }
549         else if (loser == p2) { turn = false; }
550 
551         outputStats();
552         cout << " CONTINUE? (y/n) >> ";
553         cin >> playAgain;
554         cout << endl;
555     }
556 
557     cout << " THANK YOU FOR PLAYING!" << endl;
558 }
```