CS 002 - Assignment 10: Tic Tac Toe

Collaboration Policy

We encourage collaboration on various activities such as lab, codelab, and textbook exercises. However, **no collaboration between students is allowed on the programming assignments**.

Submission Instructions

Submit in Canvas. Make sure to name your file tictactoe.cpp

Assignment Specifications

You are to implement a console version of the game tic-tac-toe. For this assignment, we are providing an initial source code file which contains skeleton code that you must complete. We also provide complete functions for you to utilize. You are **not** allowed to change the provided functions and you are **not** allowed to change the headers of the provided function stubs.

For the functions you must implement, we have provided only a stub. A stub is a function definition that compiles, but does not yet implement the complete specifications for that function. As you develop the program, you should implement each function one at a time and test each as you go.

Implementation Strategies

- We provide some variables and two global **constants** for you to utilize.
- We provide string literals for winning or tie game output in comments with provided file
- We have also provided comments to help you develop the necessary algorithm for 2 users playing the game of tic-tac-toe on a computer. Use these comments along with the function descriptions below to help you develop your program. One or more lines of your code should exist below each comment. **Remove the TODO part when you have completed that step.**
- DO NOT try to implement the entire game at once. Instead, implement one behavior at a time, only developing one particular function at a time. Functions are listed below.
- We highly recommend you unit test the function you are currently developing. You should understand how to walk through your code by hand as well as executing it in unit tests.

- Use one of the following statements when stating who won:

  `Player 1 (x's) wins!` `Player 2 (o's) wins!` `No one wins`

Functions

```
/// @brief Fills vector with characters starting at lower case a.
///
///     If the vector is size 3 then it will have characters a to c.
///     If the vector is size 5 then it will have characters a to e.
///     If the vector is size 26 then it will have characters a to z.
///
/// @param v the vector to initialize
/// @pre-condition the vector size will never be over 26
void initVector(vector <char> &v)


/// @brief Converts a character representing a cell to associated vector index
/// @param the position to be converted to a vector index
/// @return the integer index in the vector, should be 0 to (vector size - 1)
int convertPosition(char position)


/// @brief Predicate function to determine if a spot in board is available.
/// @param board the current tic-tac-toe board
/// @param position is an index into vector to check if available
/// @return true if position's state is available (not marked) AND is in bounds
bool validPlacement(const vector <char> &board, int position)


/// @brief Predicate function to determine if the game has been won
///
///     Winning conditions in tic-tac-toe require three marks from same
///     player in a single row, column or diagonal.
///
/// @param board the current tic-tac-toe board
/// @return true if the game has been won, false otherwise
bool gameWon(const vector <char> &board)


/// @brief Predicate function to determine if the board is full
/// @param board the current tic-tac-toe board
/// @return true iff the board is full (no cell is available)
bool boardFull(const vector <char> &board)


/// @brief Acquires a play from the user as to where to put her mark
///
///     Utilizes convertPosition and validPlacement functions to convert the
///     user input and then determine if the converted input is a valid play.
///
/// @param board the current tic-tac-toe board
/// @return an integer index in board vector of a chosen available board spot
int getPlay(const vector <char> &board)
```

Tie Game Example (user input is **bolded and underlined** for emphasis)

```
c
  a  |  b  |  c
-----|-----|-----
  d  |  e  |  f
-----|-----|-----
  g  |  h  |  i

Please choose a position: a


c
  x  |  b  |  c
-----|-----|-----
  d  |  e  |  f
-----|-----|-----
  g  |  h  |  i

Please choose a position: e


c
  x  |  b  |  c
-----|-----|-----
  d  |  o  |  f
-----|-----|-----
  g  |  h  |  i

Please choose a position: e

Please choose a position: b


c
  x  |  x  |  c
-----|-----|-----
  d  |  o  |  f
-----|-----|-----
  g  |  h  |  i

Please choose a position: c


c
  x  |  x  |  o
-----|-----|-----
  d  |  o  |  f
-----|-----|-----
  g  |  h  |  i

Please choose a position: g


c
  x  |  x  |  o
-----|-----|-----
  d  |  o  |  f
-----|-----|-----
  x  |  h  |  i

Please choose a position: d
```

```
c
  x  |  x  |  o
-----|-----|-----
  o  |  o  |  f
-----|-----|-----
  x  |  h  |  i

Please choose a position: f

c
  x  |  x  |  o
-----|-----|-----
  o  |  o  |  x
-----|-----|-----
  x  |  h  |  i

Please choose a position: h

c
  x  |  x  |  o
-----|-----|-----
  o  |  o  |  x
-----|-----|-----
  x  |  o  |  i

Please choose a position: i

c
  x  |  x  |  o
-----|-----|-----
  o  |  o  |  x
-----|-----|-----
  x  |  o  |  x

No one wins
```