CSCI 1300
Summer 2018
Assignment 3

This assignment is all about taking what we've learned about programming and adding functions to the equation. Functions are one of the fundamental abstractions that we use to solve and manage the complexity of the problems that we deal with in computer science and engineering. Once you've solved a particular problem once, why not use it as a building block to solve other problems?

Task 1

Let's warm up by revisiting our prime number problem from last week. **Your task** is to write a function `is_prime` that takes an integer as an input, and returns a boolean value denoting whether or not our input is prime or not.

Then, in your `main` function, you will write code that:

- Declares an integer array of 5 integer values
- Initializes the array with input from the user
- Once the array has been initialized, you will use your `is_prime` function to determine whether or not each array element is prime or not.

Name your file task1.cpp

Advice

Don't fully implement your `main` until you have tested your `is_prime` function to make sure that it works as intended. After you're certain that it works, then go ahead and write the `main` according to our specifications.

CSCI 1300
Summer 2018
Assignment 3

Task 2

Who says homework can't be fun? Task 2 will have us write several functions that will be used to implement a simple tic-tac-toe game with two players.

The Game Board

We will represent the tic-tac-toe game board using a 3-by-3 character array.

- We will call this array `game_state`
- We will denote an empty spot in the board with a space character `' '`
- Players can then mark an empty spot with either an `'X'` or `'O'` during their respective turns.
- **Hint:** Make sure you initialize this properly before the game begins

The Functions

In order to make the game easier to implement, we want you to write the following functions:

- **`print_state`**
  - This function will be used to print the game board
  - It will take the **3-by-3 game_state as an input**
  - It will return **no output**
  - Here is an example output after a couple of turns have been played

```
 X | O | O
-----------
   |   |
-----------
   |   |
```

- **`take_turn`**
  - This function is where a player makes their mark on the game board
  - It will take **2 input arguments**
    - The **first** will be the 3-by-3 `game_state` array
    - The **second** will be a character denoting whose turn it is (i.e. 'X' or 'O')
    - In this function the player will be asked which row/column they choose to mark
    - You must ensure that the row/column entered is a valid location in the game board and that the spot they entered is not already marked
    - Once a valid location has been specified, then the function will mark that location in the game board with the player's marker ('X' or 'O')

    - It will return **no outputs**
        - Wait… Shouldn't I be returning an array to update the game_state?
        - No. Since arrays are passed by reference, then we can simply modify it in our function and the changes will be reflected where the array was declared.

- **`check_winner`**
    - This function, like the name suggests, will be used to check whether a particular player has won the game.
    - It will take **2 input arguments**
        - The **first** will be the 3-by-3 `game_state` array.
        - The **second** will the character denoting which player we are checking.
    - It will return **1 output**
        - A boolean true/false denoting whether the player has won or not.
    - We all know what the win conditions for tic-tac-toe are right?

The `main()`

We now have the building blocks which we can then use to implement our game. Here's a basic outline of how the game logic will work.

```
GAME SETUP:
Initialize our game_state with all empty spots
Print a welcome message as well as the initial game state

GAME LOOP:
      Take a turn (you can pick an ordering)
      Print the game state
      If there is a winner
            Print the winning player
            Terminate program
```

Name your file task2.cpp