Introduction to Programming, PIC10A
E. Ryu
Spring 2017

UCLA

Homework 7
Due 5pm, Wednesday, May 24, 2017

Download the starter code `main.cpp`, `ticTacToeBoard.h`, and `ticTacToeBoard.cpp`. Do not modify `ticTacToeBoard.h`. We have provided `main.cpp` to give you an idea of how we intend to use the functions. `ticTacToeBoard.cpp` must not contain a `main` function. You may not use global variables.

We may take off up to 20% of the total marks for poor style; make sure to name your variables reasonably, indent properly, and comment sufficiently. Submit `ticTacToeBoard.cpp`.

Problem 1: (Tic-tac-toe)

Write the function implementations for a program that plays the Tic-tac-toe game on a 4×4 board. The winner is determined by 4 consecutive marks of the same kind. Player `X` goes first, and player `O` goes second.

The `struct`

```
struct ticTacToeBoard {
  int curr_player;
  int points[4*4];
};
```

encodes the state of the game. `curr_player==1` means it's `X`'s turn to play. `curr_player==-1` means it's `O`'s turn to play. `points[4*4]` contains the moves made so far. An empty point has value `0`, a point marked `X` has value `1`, and a point marked `O` has value `-1`.

The procedure

```
void printBoard(ticTacToeBoard board);
```

prints the board. The definition of `printBoard` is provided, and it will clarify the meaning of `ticTacToeBoard`.

The procedure

```
void initBoard(ticTacToeBoard& board);
```

initializes the board. Since `X` goes first, `initBoard` must set `board.curr_player` to `1`. Since the board is empty at the beginning of the game, `initBoard` must set all values of the array `board.points` to `0`.

The predicate

```
bool isEmpty(ticTacToeBoard board, int x, int y);
```

checks if the point `(x,y)` is empty and therefore available to play.

The procedure

```
void mark(ticTacToeBoard& board, int x, int y);
```

marks the point `(x,y)`. For example, if `board.curr_player` is `-1` (so it's `O`'s turn to play) and we wish to mark `(1,1)` then `board.points[0+4*0]` must be assigned to `-1`.

The predicate

```
bool boardFull(ticTacToeBoard board);
```

returns `true` if the board is full and `false` otherwise.

The function

```
int winner(ticTacToeBoard board);
```

returns `0` if there is no winner, `1` if `X` is the winner, and `-1` if `O` is the winner.

The function

```
int main();
```

shows how we intend to use these functions. In particular it shows that the points are refered to with 1-based indexing. So if `X` marked `(2,1)` and `O` marked `(3,4)`, the board will print to

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

You may not use any libraries aside from `iostream`, `string`, and `cassert`.

Hint. In writing `winner` you may find a *helper function* useful. Helper functions are functions intended to aid other functions but not intended to be used by themselves. For example, you could write a function

```
bool winnerHelper(ticTacToeBoard board, int player) {
  ...
}
```

that returns `true` if `player` is a winner and `false` otherwise. Then `winner` will merely call `winnerHelper` twice.