CS 150 Lab 13
Parallel Arrays

The purpose of today's lab is to for you to work with functions, parallel arrays and reading from & writing to files.

- Be sure your output works as described below.
- You will be saving one project called PUNetIDParallel (replacing the XXXXXXXX with your PUNetID) in CS150-02 Lab. There is no need to create a separate folder.
- Show the instructor or TA your solution after implementing **EACH** function and writing the results to a file.

Lab Description

Consider the following datafile "**scores.txt**" of soccer scores:

```
Concordia 3 2
Schreiner 3 0
Wartburg 3 2
Iowa 3 3
LaVerne 3 2
UCSC 2 3
CalLutheran 2 2
Pomona 2 3
```

Each line of the datafile is an opponent (string), Pacific's score (unsigned int), and the opponent's score (unsigned int). You are to write each of the following functions one at a time and show the instructor or TA your solution after writing each function. The data file will have **at most 25** lines of data.

```
unsigned int soccerReadData (ifstream &inFile,
                             string opponentName[],
                             unsigned int pacificScore[],
                             unsigned int opponentScore[]);
```

The function soccerReadData fills three parallel arrays with the specified information and returns the number of matches played. To test this function, print from the main function after calling soccerReadData, the number of teams to a file "**soccerresults.txt**". Open up the text file and show the instructor or TA your results.

```
void soccerComputeRecord (const unsigned int pacificScore[],
                          const unsigned int opponentScore[],
                          unsigned int pacificRecord [],
                          unsigned int numMatches);
```

The function soccerComputeRecord computes Pacific's win-loss-tie record and stores the number of wins in pacificRecord[0], the number of losses in pacificRecord[1], and the number of ties in pacificRecord[2]. When you return from soccerComputeRecord, print to the results file the number of wins, losses, and ties properly labeled each on its own line.

```
string soccerFirstPacificWin (/* */);
```

The function soccerFirstPacificWin returns the name of the first team that Pacific beat. You figure out what needs to be passed and how each argument needs to be passed. In the main function, print to the results file the first Pacific win properly labeled.

Optional:

Write the function prototype and a function call from the main function that will print the last team that Pacific beat. The function is to be named **soccerLastPacificWin**. If they did not lose a match, print "Won all matches".