Lab 05: Midterm Review with Iterators

This lab can be considered as a midterm review of all the things we've learned up to now. We'll be building a Team, and the Player for a sport and test them out with Iterators.

Player (10)

The players in the team should be represented by a Player class that has a name, and a position (their number on the team). It also needs a default constructor (no arguments) that gives a default name and position of 0. Two instances of Player are equal to each other when the name and position is equal.

Team (10)

A team has a name, a roster of players (an ArrayList), and a maximum number of players. **Make use of the Iterator** from the roster to implement the following:

- **contains:** Check if a Player is in the roster.
- **insert:** Insert a Player into the roster. Cannot add a duplicate player, and you cannot add more than the maximum number of players. Throw an exception in those cases like so:

```
public void makeError() throws Exception {
      throw new Exception("This is an exception message.");
}
```

- **iterator:** A public method to expose the iterator for your roster of players.

Don't use anything other than the **iterator, size, and add method** from your ArrayList variable.

Testing (10)

Let's run some tests to ensure that your code is running correctly!

1. Create Team A, a team with the name "Tigers". Create another team, Team B, using the default constructor. **The maximum number of players should be 17.**
2. Add two players built with the default constructor to Team A and catch the exception that is thrown using a try/catch statement. In the catch section, print the stack trace or another helpful message:

```
try {
	throw new Exception("This is an exception message.");
} catch (Exception e) {
	e.printStackTrace();
}
```

3. Fill up both teams with players.
4. Try inserting more players after filling them and handle the exception.
5. Test to see if contains is consistent in its results.
6. Finally, iterate over the team rosters and legibly output all the players.

Grading Criteria:

Style/submission guidelines: https://gmierzwinski.github.io/bishops/cs321/style_guidelines.html

Comments, Formatting, & Readability	**5 Marks**
Submission Guidelines	**5 Marks**
Program	**20 Marks** **See (X) above**
Testing	**10 Marks**
Total	**40 Marks**