CSCI 2110 Data Structures and Algorithms - Fall 2014
Assignment No. 2
Date Given: Friday, October 3, 2014
Date Due: Friday, October 17, 2014 (11.59 p.m.)

This assignment is on the application of Unordered Lists. You will need the source codes for Node.java, LinkedList.java and List.java. Download them (they are given next to the assignment link).

Before you begin this assignment, it will be useful to study the Expense.java, ExpenseList.java and ExpenseListDemo.java programs that were discussed in the lectures.

You will also need knowledge of File Processing and String Tokenizer (from CS 1101).

You are given the raw scores of various NHL players regular hockey games in the file nhlstats.txt. Download this file as well. The file has data organized in the following manner:

```
Name         Pos    Team   GP     G      A      PIM    SOG    GWG
Marchand     LW     BOS    45     18     18     27     91     5
Eberle       C      EDM    48     16     21     16     133    3
…
etc.
```

Note: The first row is not there in the actual nhlstats.txt file. It just tells you what each column stands for.
The meanings of abbreviations are as follows:
Pos stands for Position. In this column, C means Center, LW means Left Wing, RW means Right Wing, LD means Left Defense, RD means Right Defense, and G means Goalie.
GP stands for Games Played.
G stands for Goals scored.
A stands for Assists.
PIM stands for Penalties In Minutes
SOG stands for Shots on Goal
GWG stands for Game Winning Goals.

So, for example, in the above file, Marchand plays Left Wing, belongs to team BOS (Boston Bruins), played 45 games during the season, scored 18 goals, had 18 assists, spent 27 penalty minutes, had 91 shots on goal and 5 game winning goals.

Now, from these numbers, hockey statisticians calculate additional interesting information, such as the following:

- P (Points: this equals Goals plus Assists). For example, Marchand has 36 points and Eberle has 37 points.
- PG (Points per Game: this equals P divided by GP). For example, Marchand's PG is 36/45 = 0.80 and Eberle's PG is 37/48 =0.77. (Caution: Integer division!)
- PCT (Shooting percentage: this equals goals divided by shots on goal multiplied by 100). For example, Marchand's PCT is (18/91)*100 = 19.8 and Eberle's PCT is (16/133)*100 = 12.0. (Caution: Integer division!).

Here is what your program should do. Follow these steps.

1. First create a class called PlayerRecord.java that has all the instance variables for one player (Name, Position, Team, GP, G, A, PIM, SOG, GWG). Also include instance variables P, P/G, and PCT for that player. Add the constructor and associated get, set and toString methods. Also add the following methods:
    a. Method that calculates the player's P and sets that value

b. Method that calculates the player's PG and sets that value
c. Method that calculates the player's PCT and sets that value

Note: Use the DecimalFormat class to round off P/G and PCT to two decimal places.

```
public class PlayerRecord
{
….
}
```

2. Create a class called NHLStats.java that has as attribute an unordered list of PlayerRecord objects and a constructor to create such an empty unordered list. This class can use methods from the List class, but it cannot use methods from the LinkedList class or Node class.

```
public class NHLStats
{
	private List<PlayerRecord> playerlist;

	……
}
```

In this class, include the following methods:
Constructor to create an empty list (of PlayerRecords)
add a PlayerRecord to the list
check if the list is empty
get the first item
get the next item
enumerate
Also add the following methods:

1. Who is/are the players with the highest points? : Method that prints the player/players' name(s) with the maximum number of points and their team names.
2. Who is/are the most aggressive players? : Method that prints the player/players' name(s) who had the maximum number of penalty minutes, their team names and their positions.
3. Who are the MVPs among all teams? : Method that prints the player/players' name(s) who scored the most number of game winning goals and their team names.
4. Who are the most promising players? : Method that prints the player/players' name(s) who took the most number of shots on goal and their team names.
5. Method that prints the team/teams that had the most penalty minutes (sum of all penalty minutes of all players in that team).
6. Method that prints the team/teams that had the most number of game winning goals (sum of all GWGs for that team).
7. Method that prints the team/teams that had the least number of game winning goals(sum of all GWGs for that team).

You may add other methods as needed.

Note: If you find it useful, you can create one more class called the Team class and store the attributes of each team. This will help you search and get answers specific to a team.

3. Write a client program NHLListDemo.java with the main method that reads the file nhlstats.txt and prints the following into another file nhlstatsoutput.txt.

Note: When you read data from the file, each line is read as a String. Use StringTokenizer to break it down into individual components.
Note that the input file has rows in which the items are delimited by tabs. So you need to use the StringTokenizer in a manner similar to the following:
token = new StringTokenizer(line, "\t");

Also to convert a String to an integer value, use Integer.parseInt(…).

Your output should be similar to the format given below:

```
NHL Results Summary
Name          Position Team  GP    G      A      PIM    SOG GWG  P  P/G  PCT
Marchand      LW        BOS  45    18     18     27     91  5    36 0.80 19.8
Eberle        C         EDM  48    16     21     16     133 3    37 0.77 12.0
……
etc.

Players with highest points and their teams:
…

Most aggressive players, their teams and their positions:
…

Most valuable players and their teams:
…

Most promising players and their teams:
…

Teams that had the most number of penalty minutes:
….

Teams that had the most number of game winning goals:
….

Teams that had the least number of game winning goals:
….
```

4. In addition to retrieving the above results, your program must keep track of the number of the number of times the unordered list was scanned in order to retrieve them. Multiply this number by the number of PlayerRecords (that is, the number of lines in nhlstats.txt) and report it in your output file nhlstatsoutput.txt. This is the number of comparisons you had to make in order to conduct the above analysis.

```
Total number of comparisons done by the program:
….
```

Submit a zip file containing the following source codes:
PlayerRecord.java
NHLList.java
NHLListDemo.java

and the output text file:
nhlstatsoutput.txt