CSCI 2110 Data Structures and Algorithms
Fall 2024
Assignment No. 3

Date Given: Tuesday, October 8, 2024
Date Due: Monday, October 21, 2024, 11.59 PM on Brightspace

Submission: One ZIP file uploaded on Brightspace. Please see instructions at the end of this assignment.

Submission Deadline: Monday, October 21, 2024, 11.59 PM

Grace Time: Submissions will be accepted until 4.59 AM on Tuesday, October 22, 2024 without late penalty.

Late Penalty: Submissions received after the grace time will be subject to a 10% per day late penalty, for up to 5 days. For example, if you submit the assignment on Tuesday, October 22 at 12 noon and your score is 8/10, it will be reduced to 7.2/10. Submissions past five days after the grace submission time will not be accepted. The submission portal will close on Sunday, October 27, 4.59 AM.

Dropping of Assignments: One out of six assignments can be dropped during the semester. No SDA submission required.

This assignment on the Unordered List Data Structure. You will write a program to perform data analytics on NHL (National Hockey League) statistics data and derive trends and other descriptive stats. The nhlstats.txt file provided contains stats for a group of players over the course of one season of play.

Download the following files from the Assignment folder on Brightspace.

Node.java (Generic Node Class)
LinkedList.java (Generic Linked List Class)
List.java (Generic Unordered List Class)
nhlstats.txt (Sample text file for input)

Note: Before you begin this exercise, you might want to study the Expense.java, ExpenseList.java and ExpenseListDemo.java programs from the lecture notes.

The nhlstats.txt file has data organized in the following manner:

St.Louis	RW	TB	48	17	43	14	112	2
Stamkos	C	TB	48	29	28	32	157	2
Ovechkin	RW	WSH	48	32	24	36	220	4

...
...

The columns in order represent:

- Name of the player.
- Position (C = Center, LW = Left Wing, RW = Right Wing, LD = Left Defense, RD = Right Defense, G = Goalie)
- Team name
- Games played.
- Goals scored.
- Assists.
- Penalties minutes
- Shots on goal
- Game winning goals.

As an example, row number 3 indicates that Ovechkin played Right Wing for WSH (Washington Capitals) in 48 games this season and scored 32 goals, had 24 assists, 36 penalty minutes, took 220 shots on goal and had 4 game winning goals.

Hockey statisticians can extract interesting information from this data, and that is what you will be required to do programmatically in this exercise. You will build an unordered list of player records, searching the structure and deriving trends and other descriptive stats. Don't worry if you do not know the rules of hockey. You just need to write a program to do analytics.

To complete your solution, follow these steps.

First create a player record class . Call it **PlayerRecord.java**. It will be similar to Expense.java. PlayerRecord Objects will have one field for each column in the nhlstats.txt input file.

Next create an NHL stats class. Call it **NHLStats.java**. It will be similar in many ways to ExpenseList.java. NHLStats Objects should have a field that is an unordered List of type PlayerRecord, and methods to:

- Display the name and team name for the player with the most points (Goals+Assists).

Note: If more than one player had the greatest number of points, display all the players and their teams.

- Display the name, team name, and position for the player who was the most aggressive (had the most penalty minutes).

Note: If more than one player had the greatest number of penalty minutes, display all the players and their teams.

- Display the name and team name for the player who was the MVP (scored the most game winning goals).

Note: If more than one player had the greatest number of game winning goals, display all the players and their teams.

- Display the name and team name for the most promising player (had the most shots on goal). Note:

If more than one player had the greatest number of shots on goal, display all the players and their teams.

- Display the team name for the team that had the most penalty minutes (sum of all penalty minutes for all players on a team).

Note: If more than one team had the greatest number of penalty minutes, display all the teams.

- Display the team name for the team that had the most game winning goals (sum of all game winning goals for all players on a team).

- Note: If more than one team had the greatest number of game winning goals, display all the teams.

You may add other methods as needed.

Finally, create a demo program. Call it **NHLListDemo.java**. It will be similar to ExpenseListDemo.java or StudentListDemo.java in your lab. Your program should **accept input from a user** (specifying the name of an input file) and read a file formatted like the one described (see also: nhlstats.txt). Your program should create an NHLStats Object, and demonstrate all the methods that you have developed. You will print your outputs in the following format to a file called **nhlstatsoutput.txt**:

Enter the filename to read from: nhlstats.txt

NHL Results Summary

Players with highest points and their teams:

YOUR OUTPUT(S)
Most aggressive players, their teams and their positions:

YOUR OUTPUT(S)
Most valuable players and their teams:

YOUR OUTPUT(S)

Most promising players and their teams:

YOUR OUTPUT(S)

Teams that had the most number of penalty minutes:

YOUR OUTPUT(S)

Teams that had the most number of game winning goals:

YOUR OUTPUT(S)

Note: When you read data from the file, each line is read as a String. Use a StringTokenizer to parse the line's components. Also take note that the input file has rows in which the items are delimited by tabs, not spaces. Try the following when setting your delimiter:

```
token = new StringTokenizer(line, "\t");
```

What to submit:
Submit one ZIP file containing all source code (files with .java suffixes) and a text document containing sample outputs.

Your final submission should include the following files: Node.java, LinkedList.java, List.java, PlayerRecord.java, NHLStats.java, NHLStatsDemo.java, nhlstats.txt, nhlstatsoutput.txt.

You MUST SUBMIT .java files that are readable by your TAs. If you submit files that are unreadable such as .class, you will lose points. Please additionally comment out package specifiers.