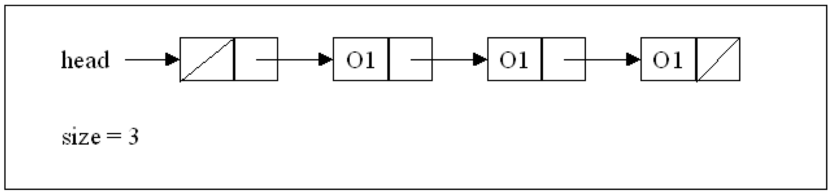
CALIFORNIA STATE POLYTECHNIC UNIVERSITY
Computer Science Department
CS 240 02 (4) ***T. Nguyen/W15***

PROJECT: **2**
DUE DATE: **Tuesday February 17, 2015**

<u>Description</u>:

You are to implement a Set ADT using a **singly linked list** where your data structure in your class will look something like this:



Note that this is a singly linked list **with a dummy head** node. Each node has two fields: the object and the link. You shouldn't assume that data elements are integers. Instead, they should be objects.

An element either belongs to a set or it does not. No duplicates are allowed. Here are the some standard operations you must build into you set class. Note there are no exceptions raised.

contain: returns boolean value. Returns true if the given object is contained in the set and false otherwise.

remove: returns boolean value. Returns true if the node containing the object is removed from the set and false otherwise.

addElement: returns boolean value. Returns false if element not added because it is already in the set and true if the element is added.

size: returns an integer equal to the number of distinct objects are in the Set.

subsetOf: returns a boolean value true if every element in set A is in set B and false otherwise.

isEqual: returns true if both sets contain the same elements where order in either set does not count.

union: if A, B, C are sets, has the form C = A.union(B). Union returns a Set that contains all the elements in set A and B, but only list duplicates once.

intersection: if A, B, C are sets, has form C = A.intersection(B) and returns a set containing only elements that are common to both A and B, but no duplicates .

complement: if A, B, C are sets, has form C = A.complement(B) and returns a set containing only elements that are in A but not in B.

toString: displays a message that indicates an object's state. An object's state is the data that is stored in the object's fields at any giving moment. Format: {1,2,3} {} {1}

Remember from your CS130 that the empty set is a set and the empty set is a subset of every set. If A={1, 3, 5, 7}, B={3, 4, 5}, D = {}, then

if C=A.union(B): then C={1, 3, 4, 5, 7};

if C=A.intersection(B): then C={3, 5};

if C=A.complement(B): then C={1, 7};

D.subsetOf(A) returns true; D.subsetOf(B) returns true; D.subsetOf(C) returns true and; D.subsetOf(D) also returns true.

A.subsetOf(B) returns false, but (A.intersection(B)).subsetOf(A) returns true

You may implement the Set using either a Node outside the Set class with all its getters and setters or where the node is a private class of set.

You must include a main to test your Set ADT. Remember that when you test subset, union, intersection and complement, test each method with at least the following five cases:

Case 1: A and B are equal but distinct sets, for example, A = {1, 2, 3}, B = {2, 1, 3}

Case 2: A and B are such that they have different sizes but one is a subset of the other, for example, A = {1}, B = {1, 2}

Case 3: A and B are non-empty and different in size but have common elements, for example, A = {1, 2, 3}, B = {2, 3, 4, 5}

Case 4: they are non-empty with nothing in common, for example, A = {1}, B = {2, 3}, and

Case 5: one is non-empty and the other empty.

You need to make sure that you have thoroughly tested every method of your set class. For the test program, you can hard-code all the special cases.

Project report:

- Page 1: Cover page with your name, class, project, and due date
- Page 2 to 3:
    - Section 1 (**Project specification**): Your ADT description. Description of data structures used and a description of how you implement the ADT
    - Section 2 (**Testing methodology**): Description of how you test your ADT, refer to your testing output. Explain why your test cases are rigorous and complete. Demonstrate that you test each method.
    - Section 3 (**Lessons learned**): Any other information you wish to include.

<u>Turn in</u>:

1. Print out of project report.
2. Email the <u>source code</u> with the file name *flast*-**Set.java**
   (ex. tnguyen-Set.java) to tvnguyen7@cpp.edu with the subject: **cs240w15 p2**. Rename the file before submitting.

Grading Guide:

- 10%: Project report and project output.
- 80%: Program correctness
- 10%: Coding – efficiency, style, comments, formats

<u>Notes</u>:

1. The following information is required in the beginning of every source file.

```
//
//    Name:       Last, First
//    Project:    #
//    Due:        date
//    Course:     cs-240-02-w15
//
//    Description:
//                A brief description of the project.
//
```

2. The submission **<u>must</u>** be legibly printed.