ITI 1121. Introduction to Computing II
Winter 2016

Assignment 4
(Last modified on March 30, 2016)

Deadline: April 12th, 2016, 11:59 pm

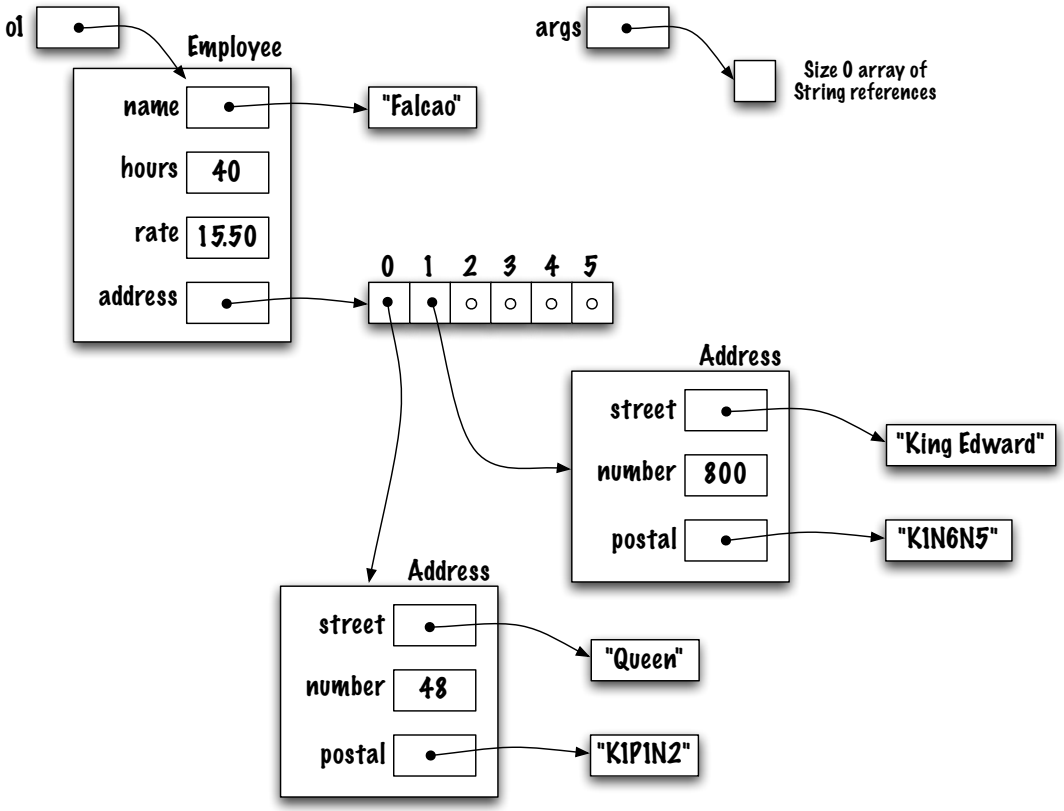
[PDF]

Learning objectives

- **Design** un Java program starting from a memory diagram
- **Implement** an instance method for a doubly-linked list
- **Create** a class method using an iterator
- **Design** a recursive method for a singly-linked list
- **Implement** a recursive method for a binary search tree

1 [20 marks]

Reverse engineer the memory diagrams below. Specifically, give the implementation of all the classes, instance variables, and constructors such that the execution of your **main()** method produces the memory diagram shown below. The name of the classes is given on the top-right corner of each object (that is, **Employee** and **Address**).



Files:

- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q1/Employee.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q1/Address.java
- **Any other classes that you find necessary**
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q1/StudentInfo.java

2 [20 marks]

Complete the implementation of the instance method **void insertAfter(E obj, LinkedList<E> other)**. The method inserts the content of **other** after the leftmost occurrence of **obj** in this list, and the elements are removed from **other**. You must provide a test program which is as exhaustive as possible.

An exception of type **NullPointerException** is thrown if **obj** is **null**. If the parameter **obj** is not found in this list, an exception **IllegalArgumentException** is thrown.

The implementation of **LinkedList** has the following characteristics.

- An instance always starts off with a dummy node, which serves as a marker for the start of the list. The dummy node is never used to store data. The empty list consists of the dummy node only;
- The nodes of the list are doubly linked;
- The list is circular, i.e. the reference **next** of the last node of the list is pointing at the dummy node, the reference **previous** of the dummy node is pointing at the last element of the list. In the empty list, the dummy node is the first and last node of the list, its references **previous** and **next** are pointing at the node itself;
- Since the last node is easily accessed, because it is always the previous node of the dummy node, the header of the list does not have a tail pointer.

Example: The list referenced by **xs** contains **[a,b,c,f]**, and the list referenced by **ys** contains **[d,e]**. After the call: **xs.insertAfter("c", ys)**, **xs** contains **[a,b,c,d,e,f]**, and **ys** is empty.

- **You cannot use the methods of the class LinkedList. In particular, you cannot use the methods add(), addLast or remove().**

Files:

- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q2/LinkedList.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q2/StudentInfo.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q2/Test.java

3 [15 points]

Create a class called **ListUtil** that contains a class method **<E> indexOfAll(LinkedList<Integer> list, E obj)**. You must use an iterator to find all the positions of the parameter **obj** in the linked listed **list**. These positions are returned in a list of integers.

Let **l** designate a list containing the elements {A, B, A, A, C, D, A}, then a call to **ListUtil.indexOfAll(l,A)** returns the following list: {0, 2, 3, 6}.

You have to provide a test program that is as exaustif as possible.

Files:

- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q3/Iterator.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q3/LinkedList.java

- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q3/List.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q3/StudentInfo.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q3/ListUtil.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q3/Test.java

4 [15 marks]

In the class **SinglyLinkedList**, write a **recursive** (instance) method that returns a **new** linked list consisting of the first **n** elements of this list. This instance must remain unchanged. The method **public LinkedList<E> take(int n)** must be implemented following the technique presented in class for implementing recursive methods inside the class, i.e. where a recursive method is made of a public part and a private recursive part. The public method initiates the first call to the recursive method.

You have to provide a test program that is as exaustif as possible.

Files:

- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q4/SinglyLinkedList.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q4/StudentInfo.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q4/Test.java

5 [15 Marks]

Implement the method **int count(E low, E high)** for the binary search tree presented in class. The method returns the number of elements in the tree that are greater than or equal to **low** and smaller than or equal to **high**.

- The elements stored in a binary search tree implement the interface **Comparable<E>**. Recall that the method **int compareTo(E other)** returns a negative integer, zero, or a positive integer as the instance is less than, equal to, or greater than the specified object.
- Your mark will be reduced if your method visits too many nodes.
- Given a binary search tree, **t**, containing the values **1, 2, 3, 4, 5, 6, 7, 8**, the call **t.count(3,6)** returns the value **4**.

You have to provide a test program that is as exaustif as possible.

Files:

- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q5/BinarySearchTree.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q5/StudentInfo.java
- https://www.site.uottawa.ca/~gvj/Courses/ITI1121/assignments/04/src/Q5/Test.java

Rules and regulation (15 marks)

Follow all the directives available on the assignment directives web page, and submit your assignment through the on-line submission system Blackboard Learn.

Your programs should not only work, they should be easily readable and follow object-oriented principles.

You must preferably do the assignment in teams of two, but you can also do the assignment individually. Pay attention to the directives and answer all the following questions.

You must use the provided template classes.

Files

This assignment has 5 questions. You have to put the source code for each questions in a sub-directory named **QN**, where **N** is the question number. Thus, there will be five sub-directories: **Q1**, **Q2**, **Q3**, **Q4**, and **Q5**.

You must hand in a zip file containing the following files.

- A text file README.txt which contains the names of the two partners for the assignments, their student ids, section, and a short description of the assignment (one or two lines).
- The source code of **all** your classes

Last Modified: March 30, 2016