Problems relating to Unit 7 (Lists, and integrating the collections)

1. The sample program NameList. Copy, examine and run NameList.java.
   1. What List implementation is being used to hold the names?
   2. Are there two Elizabeths in the list?
   3. Are the names displayed in the order in which you added them?
   4. Add code to display the first person (index 0) and the last person (use size() to work out the correct index).
2. The sample program NameQueue. Copy, examine and run NameQueue.java
   1. What List implementation is being used to hold the names?
   2. Examine the API documentation for LinkedList, and identify two methods from it which do not belong to List.
   3. Are there methods which will get you the first person and the last person without using an index? If so, use them to display the first person and the last person in the queue.
   4. What method is being used to process the first person and remove this person from the queue?
   5. Add two more persons to the queue, process the person currently at the top, then display the remaining queue
3. The sample program SubjectList. Copy, examine and run SubjectList.java
   1. Note that this program declares and uses an array of Strings, a Collection, an ArrayList, a HashSet and a TreeSet. Locate the statements which construct each of them, and explain why each one is used.
   2. This program contains one user-written method as well as main(): what is it?
   3. What type of parameter is passed to this method?
   4. Would it work if the parameter was declared to be a Set?
   5. Would it work if the parameter was declared to be a List?
   6. Would it work if the parameter was declared to be an implementation like a HashSet or an ArrayList?
   7. What is the term used to describe a method which can accept lots of different types of object and decides which one it has at run time?
   8. What is the expression used to describe the process of deciding which version of a method to call at run time?
4. The sample program ReversedFriendsList. Copy, examine and run ReversedFriendsList
   1. What type of object is being used to display the list forwards?
   2. How many methods are in the interface Iterator?
   3. What type of object is being used to display the list backwards?
   4. How many methods are in the interface ListIterator?
   5. Could you use a ListIterator object like it2 to display the list forwards? Try it and see.
5. Reverse Polish Notation

Consider how you write the expression (3+4)\*5. You can eliminate the parentheses if you write the operators after the numbers like this: 3 4 + 5 \*. This is known as Reverse Polish Notation and allows computation of the expression by reading the expression from left to right. An algorithm for implementing this approach using a stack is as follows:

The algorithm for reverse polish notation is:

If you read an operator,

Pop two values off the stack,

Combine the two values off the stack,

Push the result back onto the stack,

Else if there is no more input,

Pop and display the result.

Else If you read a number,

Push it on to the stack,

Write java code to implement this algorithm.

Enter expression in Reverse Polish Notation/Q to quit

57

5

+

2

/

Quit

Total is: 32

1. Programming exercise, integrating various Collections: Write a program with a GUI interface containing a JTextArea and a JButton. When you type some text into the JTextArea and press enter, the program should make use of Collections to do the following:
   1. List all the words used in the text area, one per line, in arbitrary order, without duplicates
   2. List all the words used in the text area, ordered alphabetically
   3. Display the contents of the text area in reverse
   4. Display the first word and the last word used in the text area

Hint: You will need to use a StringTokenizer to extract individual words from the text. It works in a similar manner to an Iterator: consult the API docs to see what exact methods are available. Display the output from each step in a separate message dialog. Advanced modification: display the output, step by step, in a second JTextArea, with a ‘Next Step’ button to clear the second text area after each step: this will also require an int ‘step number’ attribute to make sure the program waits after each step.