COMP20012 Tutorial 1: Collections

Happy New Year. For some of you, this tutorial takes place before many lectures have taken place for COMP20012, so it does not depend on any material from the course itself. We have based it on material already covered in previous programming courses, for which you have notes. There is also useful background material in the course textbook, Weiss, and copious material on the web e.g. `http://java.sun.com/docs/books/tutorial/collections/`.

1. `java.util` provides the Collections Framework, a set of interfaces for working with groups of objects. Read your previous notes on this topic and explain to your tutor the main features of the Collections Framework.

2. A *stack* is a collection of items that has the property that *the last item in is the first item out* (LIFO), so the operation *add* adds an item onto the stack and *remove* takes off the item added most recently. The operations *add* and *remove* for stacks are usually called *push* and *pop*.

   A common text processing problem is that of bracket-matching. For example the string `"[(z + b) + (c * d)]"` has matching brackets, but `"[(z + b] + (c * d))"` does not, even though there are an appropriate number of opening and closing brackets of each type, because bracket pairs are not properly nested.

   Describe an algorithm which uses a stack to determine whether a given string has matching brackets. You do not need to produce `Java` code, just a description of the algorithm.

3. Explain to your tutor how `Iterator`s are used. Why is a separate `Iterator` object needed, rather than using a class or instance variable of the `Collection` of interest?

   Write static methods to print out the items in any `Collection` in each of the following cases:-

   a) in the default order given by an `Iterator`

   b) in reverse order. (Note: you can not use a `ListIterator`, as this is not available for all `Collection` objects.)

   c) in reverse order, but without using another intermediate `Collection` (Hint: recursion)