Course Notes for CS310 – Intro to Programming Assignments

Nurit Haspel (notes adapted from Prof. Betty O’Neil)

Intro to PA1 (and PA’s in General)

This is basically a chance to practice hash tables (and lookup tables in general), as well as testing performance. General guidelines for this and other PA’s:

1. **READ THE INSTRUCTIONS** carefully. Done? Good. Now read them again. It’s not optional. Make sure you understand what is required of you before you start programming. It will save you (and me) a lot of time and e-mails. More often than not, you will realize that it’s much simpler than you think. If you ask me questions that can be answered by looking at the PA, I’ll send you to RTFM (google it).

2. While you can develop your code on any IDE you want, you will have to submit your code to your account and it will be tested there. Eclipse and IntelliJ are two nice IDE’s I’m familiar with. They are powerful but they do a lot of the work for you, which may cause troubles when porting the code. Therefore...

3. Make sure you know how to compile and run executables on linux, and in particular our linux system. There is a unix/linux user’s guide linked from the bottom of the course webpage. Use it, it is very practical and helpful. In particular, make sure you are familiar with the directory structure in Unix, the difference between an absolute and a relative path, and how to check and set permissions. All your files should be readable to everyone (permission 644), and all your directories and subdirectories should be readable and executable to everyone (permission 755). Also - you should know how to transfer files into your Unix account.

4. Just because it happens to students too often in this course – remember that Unix is completely case-sensitive, while Windows isn’t. Therefore, assume your files are case sensitive. If your code compiles on Windows but not on Linux, this may be the problem. Case sensitive means that small and capital letters are different letters! So if I ask you to open a directory called PA1, it should not be called pa1 (with small letters), or the compiler won’t recognize it.

5. Please follow the directory and file naming instructions as-is. The grader uses automated scripts which expect a certain input/output. If the code does not run as instructed, the scripts will fail.

6. Follow the compilation instructions as-is, but make sure you understand them. Don’t just parrot it. I will explain as much as I can in this document and in class.

7. I will not debug your code. There is no exception to that. Otherwise, I will try to help as much as I can but please give me as much information about your problem as you can. “My code doesn’t work” is certainly not specific enough (as said – I will not debug it for you).
Introduction to Packages

A package in java is a piece of code that performs a certain functionality, something like a module or a namespace. Packages are a good way to organize the code and are very useful in large projects. **Examples:** java.lang, java.io etc. (the package names are lang and io, respectively). Remember: source files of the same package should be in a separate directory with the same name. It won’t compile otherwise! All source files must start with a package statement. For example:

```
package cs310;
```

For outside classes, the package name becomes part of the class name. For example – if package cs310 contains a class TestPerf, then any class outside the cs310 package should import cs310 and refer to TestPerf as cs310.TestPerf. Packages can be defined inside packages. The directory structures and names are hierarchical.

As you can see in the compilation instructions, to compile the code you have to be outside the package subdirectory. When you run the code, you have to refer to it by its package name too, or it won’t compile/run. This is another common problem if you are used to working with IDE’s that do it all for you. Your code may compile from the IDE since it already sets all the correct paths, but when you compile from a command line you have to include the package name explicitly.

Sedgewick and Wayne’s Source Code

The code can be downloaded at [http://algs4.cs.princeton.edu/code/](http://algs4.cs.princeton.edu/code/). Each of their classes can be viewed and downloaded separately and the entire code can be downloaded in one jar file – algs4.jar. I recommend the latter for downloading, and the former for understanding their code. The code is very clear and very well documented. Specifically, the programming assignment focuses on the following four implementations of lookup tables:

- `SeparateChainingHashST.java`, implementing a hash table using separate chaining.
- `LinearProbingHashST.java`, implementing a hash table using linear probing.
- `ST.java`, implementing a sorted lookup table (which has a tree map).
- `SequentialSearchST.java`, implementing a sequential lookup table (essentially a list).

**Important!** Read carefully and make sure you understand the code for each one of these implementations. Also make sure you are familiar with the expected (average) runtime for search, insert and delete for each one of the implementations (it’s in the class notes and was mentioned in class). Do it before you start writing anything.