

Programming Assignment 1

Due Tuesday, January 31st

This assignment asks you to write a short Cool program. The purpose is to acquaint you with the Cool language and to give you experience with some of the tools used in the course.

1 Hash Tables in Cool

The Cool language provides very few library classes. Your task will be to write an `Hashtable` class loosely based on Java's `Hashtable` and use it for a simple computation. This class should have (at least) the following methods:

```
put(key,element)  Set the binding for key; return previous binding (or null)
get(key)          Get the binding for the key, or return null if none.
size()           Return the number of entries in the table.
elements()       Return an enumeration of (key,element) pairs in table.
```

For simplicity, the key type is `Symbol`, but the element type is the most general type, `Any`. The key is not allowed to be null. (If your code causes a dispatch abort error, that's OK.) A symbol has a hashcode. Currently these are small integers, but your code should work even if two symbols have the same hash code or if one is negative or very large.

Elements may be of any Cool class, including all basic classes and any user-defined classes. Start the table with ten buckets and increase the size whenever the number of entries exceeds the number of buckets.

2 Histogram Program

Next write a simple histogram program that reads a series of lines on its input until eof-of-file (you will need to figure out how to detect this condition), and then outputs the number of times each line appeared prefixing the line. For example, if the input is

```
hello
hello?
I said
hello?
who's there?
hello?
goodbye!
```

the output could be

```
1: I said
3: hello?
1: hello
1: goodbye!
1: who's there?
```

To get started, try the following:

```
% cd /afs/cs.uwm.edu/users/classes/cs654/401/PantherID/PA1
% make -f /afs/cs.uwm.edu/users/classes/cs654/src/PA1/Makefile
```

This will provide you with some starting files. (You can `make` the starting files into any directory, not just the PA1 directory of your afs volume; however, using your course afs volume is very convenient.)

We wrote a solution in approximately 170 lines of Cool source code including blank lines and comments. This information is provided to you as a rough measure of the amount of work involved in the assignment—your solution may be either substantially shorter or longer. If using emacs, you can use the “scala-mode” to make it easier to write syntactically correct Cool programs.

3 Extra Credit

There is a chance that you will discover a bug in our Cool compiler. We will award extra credit for legitimate bug reports; to get credit, send a bug report to `boyland@cs.uwm.edu`. Your report must include all of the needed Cool source and a transcript of a terminal session showing how to reproduce the bug (use the `script` command). There are a number of ways the compiler can potentially fail: the compiler may crash, the generated code may be incorrect, the compiler may refuse to accept a legal program, it may accept an illegal program, etc. *Please be sure you have found a bug before submitting a report!* The course staff are the final arbiters of what is a “bug” and what is a “feature”. Credit usually will be awarded only to the first person to report a bug.

4 Type casts

A frequent question students have when doing this assignment is how to do the equivalent of a C++ dynamic cast or a Java checked cast. The answer is that you will need to use Cool’s “match” construct.

5 Files

Turn in your program by placing all relevant files into the PA1 directory in your afs volume. That is, place `Hashtable.cool`. `Histogram.cool` in the directory

```
/afs/cs.uwm.edu/users/classes/cs654/001/PantherID/PA1
```

This is particularly convenient if you do your work in this directory (as mentioned above).