CptS 580 Advanced Programming Languages
Final Exam
April 27, 2012

Instructions
Time limit: none, but a reasonable amount of time to spend is 5 hrs. If it is getting to be a lot more than that, please contact me for clarification of how to proceed.
Due date: midnight, Friday, May 4. Turn it in by emailing it to me – hauser@eecs.wsu.edu. Your submission should consist of 3 attachments to your email:

1. a PDF containing the answers to problems 1 and 2 and the narrative portion of answers to problems 3 and 4;
2. a source code file for problem 3
3. a zip file containing your source code for problem 4 – you should supply code for both the Range class and the IteratorHelper class.

Resouces: open book, open notes. You may also use web resources, etc. Your work must be your own – without collaboration with others.
Questions: email me; questions asked after noon Friday, May 4, may not be answered.

Turn in format: Put your name in every file that you submit. Please also write at the top of the first page of the PDF file, clearly labeled, the amount of time you spent researching and writing the exam. If you used references other than the textbook you must cite them as part of the answer to each question. In the PDF, please start the answer to each *numbered* question on a new page – i.e. your submission should have at least 4 pages.

1. Grammars (1st exam second chance)
In ordinary algebraic notation, multiplication may be denoted by juxtaposition of constants and variables, while exponentiation may be denoted with ^. Assuming that non-terminal <t> will generate (through some unspecified grammar whose details don't matter) any variable or number, give a grammar for algebraic expressions involving multiplication (denoted by juxtaposition) and exponentiation (denoted by ^). Exponentiation has higher precedence than multiplication and associates to the *right.* Multiplication associates to the *left.*

2. Function closures and storage management
Based on what you know about manual and automatic storage management (garbage collection), call stack design, and what it means to be a first-class value, argue convincingly that first-class function values are much more likely to be found in languages with garbage collection (e.g. C#, Haskell) than in languages without garbage collection. Be sure to consider the implications of first-class function values' lifetimes for management of the stack, both manually and when garbage collection is used.

3. Argument evaluation order
In the imperative language of your choice, implement a program that demonstrates the order in which arguments are evaluated for a function of 3 parameters. 1) Tell me the name of the language, 2) identify the particular implementation that you tested, 3) provide the source code that you wrote, and 4) the output of the program along with notes describing why the output justifies your conclusion about argument evaluation order. In addition, research the language's documentation to see if you can find a specification of what the order *should* be. Cite any sources, of course.

Example: suppose there is a call in your chosen language that looks like

```
f(exp1, exp2, exp3)
```

In what order are exp1, exp2, and exp3 evaluated? Note: you *must* write a program to demonstrate the order.

4. True Iterators using Threads

Refer to the discussion of iterators in section 6.5.3 of the textbook and the Java Iterator interface specification found at http://docs.oracle.com/javase/1.4.2/docs/api/java/util/Iterator.html. From the textbook, you first need to make sure that you understand the difference between a true iterator and what the book describes as an iterator object, because I don't want an iterator object as the answer to this problem!

The goal of this problem is to write a class that can be used as helper class (or base class if that proves appropriate) to allow programmers to create true iterators. So a user class might look like this (note that there are much better ways to do this particular example in Java but this is intended to illustrate the use of IteratorHelper with pretty much the simplest possible example):

```
class Range implements Runnable {
   int low, high;
   IteratorHelper<int> ih = new IteratorHelper<int>(this);
   public Range (int low, high) {
      this.low = low;
      this.high = high;
   }
   public boolean hasNext() {
     return ih.done();
   }
   public int next() {
     return ih.get();
   }
   run () {
      for (int i = low; i<high; i++) {ih.yield(i);}
      // … something else to handle being done? …
   }
}
```

The idea of IteratorHelper that I want to see you work out is that it runs the run() method of the object passed to its constructor in a separate thread when the ih start() method is called. That run() method repeatedly calls the ih yield method to deliver up the sequence of values from the iterator. The following is a sketch of the general idea of IteratorHelper.

```
class IteratorHelper<T> {
   … you fill this in …
   public IteratorHelper (Runnable r) { … you fill this in … }
   public start() {… you fill this in …}
   public yield(<T> v) { … you fill this in … }
   public <T> get () { … you fill this in … }
}
```

I don't guarantee that the sketch is workable in any or all respects – it is just suggestive of what is needed. I'm not looking for subtleties about public/private, etc. You will need to make some of the methods be `synchronized` and add `wait()` and `notify()` operations as needed. And it may well be that you can't get away with `<int>` as the type parameter in any but the most recent version of

Java: I don't care about that either; switch to `Int` if you need to. Similarly, feel free to modify Range to work with your implementation of IteratorHelper in whatever ways are needed.

In short: the problem is about creating a helper class for doing true iterators using threads. You may change anything I've supplied above if it proves useful in doing that.

Provide a narrative explanation of how you intend for your code to work and make sure to comment any significant tricky bits in the source.