CNC 2012
Programming Languages
Professor Sherri Shulman
Homework 1
Owen Meyer

1.1)
a. Eiffel is an object oriented language designed by Bertrand Meyer. Implements garbage collection, inheritance, static typing (where even the primitive types are class based), as well as other features. Eiffel focuses on readability with simple, extensible, reusable coding solutions. It has introduced features into Java and C# languages.

b.
Perl was invented by Larry Wall in '87 for general purpose Unix scripting. Perl was influenced by Smalltalk, C++, Unix Shell and others. It provides powerful tools for text processing and is fairly popular for internet programming tasks including sys-admin and network programming. It is not strictly OO, imperative, or functional, but supports all these paradigms. It has had influence on Python, Ruby, PowerShell and others.

c.
Python is a popular multi-paradigm language supporting OO, imperative and functional styles. It was developed by Guido Van Rossom in the early '90's. It has taken inspiration from C++, Java, Hakell (yay!) and others. It is the scripting language of the popular free 3-D modeling and animation suite Blender. It focuses on sparse uncluttered style of coding (using indentation instead of brackets for example). Python has had influence on Groovy, Ruby, JavaScript and others. It comes with a large standard library, providing tools suited for a wide range of applications.

1.2)
```
public class B {
        private double b;
…
```

vs.
```
public class BankAccount {
        private double balance;
```

1.3)
a.

b.
It seems like he is saying that no matter how well designed a language is, the craft of computer programming is complex enough an endeavor that there is an endless supply of opportunities to perform given tasks incorrectly. For example by analogy, in construction there are building systems that are seemingly very simple to execute. However, it is just as easy to build incorrectly with these systems as it is to build correctly.

1.10)
Haskell vs. Java (you know who I pick; I'm just biased that way..)

Simplicity/ readability:

Haskell is very elegant and easy to read. Of course to the extreme newbie, it seems confusing, but once the initial curve is overcome, the code practically documents itself due to it's elegant structure.

Java is also easy to read although not nearly as beautiful.

Clarity about binding:

Haskell is crystal clear on this issue, though when monads get into the mix, they introduce bindings that are not easy to understand at first. They become fairly clear with experience.

Java I suppose can be a bit more opaque in some areas though my experience hasn't really delved that deeply into that nature of bindings in Java.

Reliabilty:

Haskell is always there for me.

Java is fickle and can be moody when it doesn't feel real.

Support:

Haskell has lots of great material online for many questions. I have had trouble from time to time finding answers to some questions.

Java is ubiquitous. There are hoards of online heroes waiting around the clock to answer your questions.

Abstraction:

Haskell rules the underworld here. So easy to define abstract data structures and functionality.

Java comes with an enormous library of reusable features. One can get lost in there and never come out.

Orthogonality:

Haskell is built up from very simple components and is highly versatile in it's ability to combine them in an endless array of pearls of wonder.

Java has implemented many features that Haskell naturally displays (such as closures) however, as complexity grows, some of these features are trickier to utilize.

Efficiency:

One must learn how to code for efficiency in Hskell as it is not obvious to the newbie. However, it holds a great power towards efficient computation including (so I have read) naturally implementing concurrency (with monads!).

Java is pretty amazing considering it is running on a virtual machine (though this makes it feel moody sometimes).