Calvin Nelms
iEx6
Due: 11/15/11
Software Engineering I

1. - A programming environment should provide an easy and efficient way to debug your written code. It is important for a team to have an straightforward way to step through code to discover errors that aren't necessarily syntax errors.
   - A programming environment should provide a way to test your written code. It is crucial to test your program with as many different variables as possible. The more you can test your code before a production release, the better your code will be.
   - A programming environment should provide the coder with in depth reports on when and where errors occur. An example of this would be when a program does not compile. The environment should present the user with a report of information such as what line number the error occurred at, what type of error it was, and maybe provide suggestions to fix the issue.

2. - One advantage of the Dracula programming environment is its ability to perform predicate-based testing. This is the first time in my college career that such a topic has been brought up. I really see the need and importance of testing your program thoroughly to check for errors.
   - A second advantage is its simple GUI design. There is not much to the interface. This creates an easy to learn environment to focus on understanding the programming language. Other GUI's such as Netbeans or Eclipse require one to have an in-depth understanding on how the interface works.
   - A last like of the Dracula programming environment is the fact that you can limit the amount of memory a Dracula program can utilize. This prevents the program from running excessively. There were a number of times where I coded an infinite loop, but didn't realize one existed. Limiting was a good thing as it prevented my system from running out of memory.

3. - One dislike I have on Dracula is the speed issues. Dracula is slow to start up, slow to run, and even slow to do common tasks such as scrolling up and down the page. I have even had issues where Dracula never loads at all. I will then have to force quite the application, and reopen.
   - A second thing I dislike is the pain of running PSP reports. I constantly had issues with the formatting of the text-based program. It is too finicky to type up long error reports and expect the format of them to be perfect. The debugging of the PSP reports never worked as well.
   - The last dislike I have of the Dracula programming environment is how outdated it is. In today's world, graphics are a necessity. It is extremely inefficient to write a Lisp program and have an advanced GUI interface.

Team Lovelace

To: Dr. Rex Page
From: Calvin Nelms
Date: November 15, 2011
Subject: Programming Environment Selection

Problem

This memo is specifying the programming environment Team Lovelace will utilize to design and implement the control software for the new nuclear reactor.

Recommendation

Because of the importance of this project, it is crucial our team utilizes the language based on three important qualities. These qualities include the productivity of the language/IDE, the power of the language, and the familiarity of our staff of that chosen language. After much research and consideration, Team Lovelace has chosen language of C++, to create the control software that will run the nuclear reactor.

Reasoning

The productivity of the language and appropriate IDEs is crucial when selected a language for such an important project. The C++ language will allow our team to choose from several including Microsoft Visual Studio, xCode, or even Netbeans. Not only is there an abundance of IDE's to choose from, each choice warrants many community support forums and training sections that will allow our team to excel.

A second reason in choosing C++ is the fact that it is a very powerful language. Due to the nature of the nuclear facility project, it will require a low level language that is safe and secure. Although there is a danger to low-level programming, C++ allows you to complete it in a safe way. The language is also efficient, which allows for quick calculations of critical data.

Lastly, Team Lovelace chose C++ due to the fact that we are very familiar with the language. This comfort level will be important as we are dealing with critical calculations. Another advantage is the availability of support forums and communities for the C++ language. There are many paid and free sites that offer in-depth learning materials for this language.