**Lab and Program Reports**

All labs programs and project should result in a report describing the actions and results of the lab, and a set of appropriate deliverables. While it may be tempting to simply write a program and believe the assignment is complete, the code is never enough. Any assignment that is not appropriately documented is not a completed assignment. In professional programming, the coding is what we do for fun, the documentation is how we earn our paycheck.

**The report format:**

1. **Introduction/Purpose/Intent:** A short statement about the intent of the assignment. This is also where interpretations of the assignment are explicitly stated (Note: stating an interpretation does not mean that the interpretation is correct, or that full credit will be given, but it does provide a basis for partial credit in the assignment).

***Example:*** *“For this assignment we will write a compiler for the C language using Python. As the only C book I have on my shelf is Whitesmith C, rather than K&R, I will write it to that specification.”*

1. **Process:** Describes the process used to accomplish the assignment. More detail is included here if this is an assignment where the task is to follow a process. Even if there is a final output (program, etc.) the process needs some description. Key algorithm implementations should be shown and explained in this description.

***This should make plentiful use of screenshots and other intermediate artifacts when possible. Snippets of code should be included and described.***

Remember to explain what compilers and compiler options are used.

***Example:***

“To implement this function I used the algorithm of a for loop and post increment to ensure that my number is always bigger than yours. The algorithm is show below:

for (i=0; i<100; i++) {mynumber[i] += yournumber[i]++}

The next section prepends an ‘est’ to any word from the word list to make sure that my word is superior to yours.”

1. **Testing:** Any program or other artifact needs to have some testing performed to ensure that the results are correct. The test plan and status are described here.

**Example:**

“Testing for this program will include random values of yournumber, as well as the edge cases of MIN\_VALUE, -1, 0, 1, MAX\_VALUE. After the insertion of these values, the output will be inspected to ensure that mynumber is bigger than yournumber for all values of yournumber.

Testing will also include various words from the word list, such as big, heavy, etc. as well as a sampling of words not included in the word list. For those words on the word list, the output should provide a correct expression of the betterness of my word. The results will be evaluated by inspection. “

1. **Results:** Describes the results of the assignment. This should fully demonstrate that the intent of the assignment has been accomplished. This may include screenshots, output, and other artifacts that demonstrate the results. Complete code and lengthy materials are included as attachments and referenced here. Short snippets of code should be included in the text.

***Example:***

The following screenshot shows the results of testing using a variety of values for yournumber. Observe that the program failed to accomplish its goal as it did not end up with a bigger number when yournumber had the value of MAX\_VALUE.

The second screenshot shows the results of inputting the words tall, big, and tiny. The correct results appear for tallest, and tiny (not on the list), however big resulted in ‘bigest’, not the correct ‘biggest’.

1. **Conclusions:**  Based on the results and the intent of the assignment, what conclusions can you draw about the assignment. This may also include lessons learned (i.e. “*Whitesmith C is a crappy implementation of C as is any version where white space and machine word boundaries have significance”*), or other valuable observations.

***Example:***

Although my attempts to ensure my superiority were not always successful, in most cases they were. Those areas that were not can be fixed with special conditions in the code to handle these kinds of special cases.

*[Instructor note: This student would receive a lower grade for not having handled special cases in the assignment.]*

1. **References and Acknowledgements**: This section is for the documentation of sources referenced and of assistance obtained in completing the exercise. Care should be taken to ensure that any work submitted is either the work of the individual submitting the work, or is acknowledged to be the work of others. The extent of documentation needed is within the guidelines of the University and the Department regarding academic integrity.

**General Report Instructions**

All reports need to be written in correct English using full sentences and proper grammar.

All reports need to be submitted in a .docx format with the report as a single document. Lengthy appendices may be in individual files.

**Deliverables:**

* Deliverables for an assignment should consist of no more than two separate files.

1. The first of these is the report in a .docx format.
2. The second may be either a single file in an appropriate format for the assignment or as a .zip file (use only the .zip format, no other file compression format is acceptable) which contains all needed files for the submission.
   1. The files within the .zip file need to be organized in a clear and understandable manner through the use of appropriate folders/directories, labeling, and the use of README files.

* Executables submitted need to be in “ready to run” format. That may include either a Windows/Linux executable (as designated in the assignment instructions), or an installer which ensures that all needed libraries, files, and other elements are included in their proper location.
  + The student should not make any assumptions about what is included on the computer used to evaluate the assignment.
  + Any external libraries or files not included in the installer must be noted by the student.
* When submitting Executables and installers the student needs to include explicit instructions the proper running of the program.