SOFE 3950U / CSCI 3020U: Operating Systems

Lab #2: MyShell Project

Objectives

- Learn to work in groups to develop software using git
- Gain experience developing multi-source file projects in C
- Experience using Makefiles and other software build tools

Important Notes

- Work in groups of **three** students
- All reports must be submitted as a PDF on blackboard, please include the url of the repository in the report or in the submission.
- Save the submission as <lab_number>_<first student's id> (e.g. lab2_100123456.pdf)

If you cannot submit the document on blackboard then please contact the TA with your submission at **neil.seward@uoit.ca**

Lab Details

Notice

It is recommended for this lab activity and others that you save/bookmark the following resources as they are very useful for C programming.

- http://en.cppreference.com/w/c
- http://www.cplusplus.com/reference/clibrary/
- http://users.ece.utexas.edu/~adnan/c-refcard.pdf
- http://gribblelab.org/CBootcamp

The following resources are helpful as you will need to perform string tokenization and use POSIX functions.

- http://www.tutorialspoint.com/c_standard_library/c_function_strtok.htm
- http://www.thegeekstuff.com/2012/06/c-directory/

Lab In-Class Activity

1. For the purpose of this lab, either create a new repository on GitHub for this lab, or create a folder in your existing GitHub repository that you created in the previous lab.

2. Download the example source code and Makefile and use git to add the contents and push it to your GitHub repository.

3. Ensure that you are able to use the Makefile and C source code examples to build the code using **make**. If you are having issues you may need to modify the Makefile and set the **CC** variable to **gcc** instead of **clang**.

4. Before writing any of the source code for your project, review the entire shell project description and ensure that you understand what is required.

5. Next, work in groups to write the documentation for the **README** file for the shell project, which is displayed when the **help** command is used.

6. Finally, work on implementing the string tokenization for the shell to process each of the commands using the resources listed above.

Lab In-Class Deliverables

Notice

Please complete the following deliverables during the lab session and demonstrate your work to the TA before the end of the lab to receive a group evaluation.

1. Demonstrate that you have created a GitHub repository for the lab activity and that you have committed and pushed all of the source code provided in the templates to the repository.

2. Demonstrate that you are able to build the source code using the **make** command.

3. Demonstrate your current **README** that you have written for the shell project.

4. Demonstrate that you have a string tokenizer that can be used to tokenize the shell commands working. It is imperative that you have a **basic string tokenizer working** by the end of the lab as this is one of the most challenging aspects of the shell project, the TA is present in the lab to assist you in completing the string tokenizer.

Lab Project

Notice

The lab project consists of completing the entire MyShell project with some clarifications and items removed, you will be required to submit your completed MyShell project several weeks later following the lab session, it is **NOT DUE** during the lab session.

Project Clarifications

1. The lab project consists of completing and submitting the **entire** MyShell project, and is an extension of the lab 2 in-class session where you were given guidance on the project and only required to submit a subset of the entire project.

2. For the purpose of the lab project, make sure to use the existing GitHub repository created during the lab 2 in-class activity. You will be working in the **same groups** for the project as in the labs, as this is merely a project for your lab groups.

3. For the project you must ensure that all of the shell features (items **i … ix**) are implemented as specified, most of these features are merely commands that your shell will provide to the user.

4. Shell command clarifications:
   - **i**, each time the directory is changed you must set the character array **PWD** to the new current directory (this is a shell environment). You must use POSIX functions to change directories on the local filesystem.
   - **iii**, for dir you must use POSIX functions, see the resources above.
   - **iv,** for environ list all of your environment strings and their values in the terminal (e.g. PWD, SHELL, etc.)
   - **ix**, you must use a POSIX or other similar method in C to determine the directory the shell was executed from, the character array **SHELL** must then be set to this path.

5. **item 2** and **item 5** of the MyShell description are **NOT REQUIRED** for the submission, as they require the use of forking, which has not been covered yet.
6. **Item 4** is much easier than it appears, any program executed in the terminal (which is a shell itself, likely **BASH**) supports STDIN and STDOUT file streams, your shell should already satisfy the criteria for **item 4** without any modification. You can use STDIN and STDOUT to have a text file containing all the commands to test as the STDIN **inputfile** and record the output in STDOUT to the **outputfile** verify that all your shell commands works as expected.

Lab Project Deliverables

Notice

Please complete the deliverables as specified and include your README document, all source files, and the Makefile. Your blackboard submission should be a single zip file containing all of your source code for the project and your README document.

1. All sources files, all of the **Project Requirements** described in the shell project document must be met, however recall that **items 2 and 5**, which require forking are **NOT REQUIRED**. The primary evaluation will be that all of the shell features (items **i … ix**) are implemented as specified.

2. A Makefile is included so that the source code can be compiled, if your Makefile does not work then marks will be deducted, make sure that your code can be compiled using a Makefile.

3. A detailed README document as outlined in the shell project document, which explains all of the functionality and usage of your shell.