Lab 03 – Part A: Introduction To C – Part 2

Assigned: 2019-02-05 08:00:00

Due: 2019-02-07 23:59:00, concurrent with Part B of this lab.

Instructions:

– Written portions of this assignment are submitted via Canvas. Unless specified otherwise, the written portion of the assignment is to be completed using LaTeX. All derivations, images, graphs, and tables are to be included in this document. Handwritten solutions will receive zero credit.

– Code portions of this assignment are submitted via `code.vt.edu`. Source code must be in the private repository, to which the CMDA 3634 instructors must have access.

Deliverables: For this assignment, you are to submit the following:

1. (Canvas) `<pid>_Lab_03.pdf`: A PDF file, renderd by `pdflatex` (the file generated by Overleaf is sufficient) containing the answers to the questions requiring written answers. Use the template provided in the project repository. Your submission should include both Part A and Part B of this lab.
2. (`code.vt.edu`) The source files required to compile and run your solutions to the lab and the tex and image files for your report, in the appropriate directories.

Collaboration: This assignment is to be completed by yourself, however, you make seek assistance from your classmates. In your submission you must indicate from whom you received assistance.

Honor Code: By submitting this assignment, you acknowledge that you have adhered to the Virginia Tech Honor Code and attest to the following:

> I have neither given nor received unauthorized assistance on this assignment. The work I am presenting is ultimately my own.

Resources

- More on Git:
  - https://www.taniarascia.com/getting-started-with-git/
  - http://rogerdudler.github.io/git-guide/
- More on LaTeX:
  - https://www.overleaf.com/learn
- More on C programming:
  - https://www.geeksforgeeks.org/c-programming-language/
  - Recommended texts
- Passing-by-pointers:
  - https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.cbclx01/pass_by_pointer.htm

Tasks

Warning: I have indicated where you should be running commands in a terminal with the > character. This character is **not** part of the command!

1. **Setup** your coding environment.

   (a) Pull the lab materials from the upstream repository.

   ```
   > git fetch upstream master
   > git merge upstream/master
   ```

   Be careful with the forward slash in the second command.

   (b) Examine the history to see what new files have been added by the instructors. Note that this includes the solutions to Lab 02.

   ```
   > git log --name-status
   ```

   The name status will show, in the log, which files have been added, modified, and removed in each commit.

   (c) Push your local master branch to the origin repository to add the new changes.

   ```
   > git push origin master
   ```

2. **Implement** the following requirements in C. Read the readme file in the `labs/lab03/code/` directory. Be sure to use git to commit your code regularly.

   (a) In `labs/lab03/code/vectors3D/`, there is a copy of the solution (`vectors.c`) to Lab 02, as well as a makefile for building the application.

      i. Run the `make single` rule to be sure that the appliction `vector3d_single` builds.

      ii. Divide `vectors.c` into three files:

      - A header file named `vector3d.h` containing the definition of the `Vector3D` data type and the declarations for the `norm`, `axpy`, and `inner_product` functions.
      - A source file named `vector3d.c` containing the definitions of the `norm`, `axpy`, and `inner_product` functions. Your source file should include the fewest possible `#include` statements.
      - A main program named `vector_driver.c` containing the main function which answers the questions from Lab 02. Your source file should include the fewest possible `#include` statements.

   iii. Verify that your new project is setup correctly by running the `make divided` rule. Compare the outputs of the `vector3d_single` and `vector3d_divided` programs to see that they are the same.

   iv. Commit your changes at this time!

  (b) Convert the program to use pass-by-pointer, so that no copies of instances of the `Vector3D` type are made. Add new functionality.

   i. Convert `norm` to accept a `Vector3D` passed-by-pointer.

   ii. Declare and define a `normalize` function, which uses `norm` to normalize a 3-vector, in place.

   iii. Convert `axpy` to accept its `Vector3D` arguments as passed-by-pointer. Modify the function so that the return vector is an *argument* to the routine, rather than a return value. The return type of `axpy` should then be `void`.

   iv. Convert `inner_product` to accept its `Vector3D` arguments as passed-by-pointer.

   v. Modify `vector_driver.c` so that it works with the new pass-by-pointer interfaces. Run the `vector3d_single` and `vector3d_divided` programs to see that they are the same.

   vi. Verify that your implementation works correctly by running the `make test` rule and running the `vector3d_test` program. Verify that `vector3d_test` indicates that you passed the tests successfully. If not, debug until you have passed the tests. You may consider testing these routines incrementally.

   vii. Redirect the output of `vector3d_test` to a text file named `output.txt`.

```
> ./vector3d_test > output.txt
```

   Be sure to include this output file in your final git submission.

3. **Answer** the questions listed below. The following workflow uses Overleaf, but you are welcome to use a local tex installation if you prefer.

  (a) In Overleaf, create a new empty project.

  (b) Copy the files `lab03_report.tex` from the `labs/lab03/report/` directory to Overleaf.

  (c) Answer the questions, and upload data files where necessary.

  (d) In your VM, copy your report tex source and images into the `labs/lab03/report/` directory. Overleaf allows you to download projects in `.zip` format. Download the zip file, move it to the report directory, and unzip it.

```
> unzip <project_name>.zip .
```

  (e) Add them to git, and commit them.

4. **Submit** your results.

  (a) When you have completed Part B of this lab, upload a PDF of your report to Canvas.

  (b) Push your source code and latex files to `code.vt.edu`. From anywhere in your local projects repository

```
> git push origin master
```

  (c) Examine your assignment repository on `code.vt.edu` to be sure that all of your materials have been correctly submitted.

Questions

Answer the following questions using the latex template provided in the `lab03/report` directory of the assignment repository.

1. Use the `listings` package to include your output (`output_pt_a.txt`) in your pdf. You will need to copy `output_pt_a.txt` to the reports directory.
2. The `axpy` routine no longer uses the return value, as the return variable is an argument to the function. How can we make use of the function return value to get some use out of it?
3. For these routines, give one reason why we might choose to pass only the structures by pointer and not the scalar values?
4. Other than the instructor or TAs, who did you receive assistance from on this assignment?