Lab 01 Specification – Explore PL's
50 points

Due by: 09/06/2021 2:00 PM

Lab Goals

- Exploring Syntax and Semantics in different Programming Languages.
- Practice implementing logic in Imperative PL's.

Learning Assignment

If not done previously, it is strongly recommended to read all of the relevant ”GitHub Guides”, available at the following website:
`https://guides.github.com/`
that explains how to use many of the features that GitHub provides. This reading assignment is useful to understand how to use both GitHub and GitHub Classroom. To do well on this assignment, it is also recommended to do the reading assignment from the section of the course textbook outlined below:

- **PLP chapter 01, section 1.1, 1.2**

Assignment Details

Now that we have discussed some basics of Programming Languages, we are ready to explore and program in different PL's. In this lab, we are going to program in 5 popular languages, namely, **C, C++, CSharp, Java, and Python**.

In this lab, students will practice developing a programmatic solution to problems such as generating an **Inverted Triangle** and **Pyramid** in a variety of PL's to retain the knowledge from the class discussions so far. This also includes learning new syntax and new semantics in different languages and the realization of programming the same logic regardless of the language implemented.

Students are not expected to achieve mastery in any single language in this course. Instead, we are trying to learn different languages in the context of studing the principles of programming languages. In other words, our goal is to put the principles of programming languages at the forefront and not achieving langugage mastery. In this process, this lab expects you to think, and come up with the logic to solve CS problems and practice integrating your solution (logic) in different PL's.

At any duration during and/or after the lab, students are recommended to team up with the Professor and the TL(s) to clarify if there is any confusion related to the lab and/or class materials. The Professor proofread the document more than once, if there is an error in the document, it will be much appreciated if you can communicate that to the Professor. The class will be then informed as soon as possible regarding the error in the document. Additionally, it is highly recommended that students will reach out to the Professor in advance of the lab submission with any questions. Waiting till the last minute will minimize the student's chances to get proper assistance from the Professor and the Technical Leader(s).

Students are recommended to get started with this part in the laboratory session, by discussing ideas and clarifying with the Professor and the Technical Leader(s). It is acceptable to discuss high-level ideas with your peers, while all the work should be done individually. Late submission is accepted for the part(s) in this section, based on the late policy outlined in the course syllabus.

It is required for all students to follow the honor code. Some important points from the class honor code are outlined below for your reference:

1. Students are not allowed to share code files and/or other implementation details. It is acceptable to have a healthy discussion with your peers. However, this discussion should be limited to sharing ideas only.
2. Submitting a copy of the other's program(s) and technical reports is strictly not allowed. Please note that all work done during lab sessions will be an opportunity for students to learn, practice, and master the materials taught in this course. By doing the work individually, students maximize the learning and increase the chances to do well in other assessments such as skill tests, exams, etc · · ·

Preliminary Steps

It is important that you can set up Docker and GitHub to complete the rest of the lab. Please follow the guidelines below to complete the preliminary steps.

1. **[Docker Setup.]** At this point, I expect the MAC, Linux, and Windows Pro users, to have this step completed based on our previous class discussions. For those who had not completed this step, the documentation below should provide more details regarding the download and installation setup.
   - Get Docker setup completed on your laptops:
   - Docker Mac Setup:

     `https:/docs.docker.com/docker-for-mac/install/`
   - Docker Ubuntu Setup

     `https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-18-04`
   - Docker Windows Setup:

     `https:/docs.docker.com/docker-for-windows/install/`
   - If the setup goes correctly as desired, you should be able to get started and validate the Docker version and run the hello world docker container using the following commands:

     docker –version

     docker run hello-world
   - There are some more documentation for Docker get started to test your installation in the link provided below:

     `https:/docs.docker.com/docker-for-mac/`

     `https:/docs.docker.com/docker-for-windows/`

2. **[Loading Docker Container.]** There are two steps in loading the container, namely:

   - Build the container
   - Connect and Run the container

   Build the container: So to build the container. the following steps should be performed.

   (a) First, accept the lab URL provided in Slack. After downloading the lab folder from the GitHub classroom, navigate to the cmpsc201-fall-21-lab01 directory using terminal (Mac/Ubuntu) or Command Prompt/Docker quick start terminal (windows).

   (b) Build the docker image using the following command:

   docker build -t cs201lab01 .

   Please note, you are required to have the period in the command above.

   (c) Note: In the command above, cs201lab01 is the user-provided image name. This could be random. But it is recommended to use the same name to easily follow the rest of this document. Additionally, it is required to be inside the `cmpsc201-fall-21-lab01` directory to run the build command. If you are not inside the `cmpsc201-fall-21-lab01` directory, you may receive an error message.

   (d) Upon successful build, it is recommended to verify the correctness of image creation by using the following command:

   docker image ls

   (e) The image named ”cs201lab01” should be listed as one of the outputs from the command above.

   Connect and Run the container: So to create and run the container. the following steps should be performed.

   (a) Run the docker container based on the image created in the previous steps using the following command:

   Mac/Ubuntu:

   ```
   docker run --rm -v $(pwd)/src:/root -it cs201lab01
   ```

   Windows:

   ```
   docker run --rm -v "%cd%/src":/root -it cs201lab01
   ```

   (b) To run the above command, it is required to be inside the `cmpsc201-fall-21-lab01` directory. And, please note, you will log in to the container after entering the above command.

   (c) After creating the container, the run command above creates a mount between the host machine and the container with a shared folder space. So, any files placed inside the host mount directory can be easily accessible inside the container mount directory and vice versa.

   (d) Compile and Execute the programs in the `triangle` directory using the commands provided in the `run-commands` file in `triangle` directory.

   (e) You are required to compile the program first before executing it by following the commands in `run-commands` file. At this point, you will see a Triangle Shape printed on your screen, based on the user-provided number of rows. That is, similar to the line below:

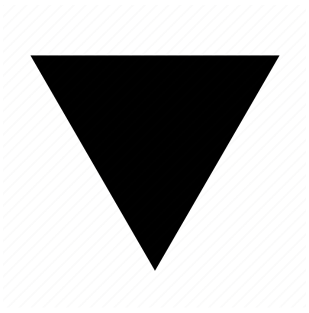
   Enter the number of rows: 5

   ```
   *
   * *
   * * *
   * * * *
   * * * * *
   ```

3. **[GitHub Setup.]** Take a look at the detailed documentation for getting started with GitHub, which is available at: `https://docs.github.com/en/get-started`

   You are required to know the procedure to git clone, git pull, git add, git commit, and git push to access the lab specification folder and to submit your lab for grading purposes. If there is an issue with your GitHub setup please discuss it with your Technical Leader(s) and/or the Professor.

Part 01 - Inverted Triangle (20 points)



Modify the code files in the `triangle directory` to implement an inverted triangle. The programming solution in each language is worth 4 points each. The starter code displays the triangle using the logic outlined in the pseudocode below:

```
1: for $(i = 0; i < rows; i = i + 1)$ do
2:     for $(j = 0; j < i; j = j + 1)$ do
3:         Print ("*") and Tab
4:     end for
5:     Print New Line
6: end for
```

Explore how the logic is implemented and pay attention to the syntax used in Compiled languages such as **C, C++, CSharp, and Java** and Interpreted language such as **Python**. Note: It is the same logic with similar semantics but the syntax is different in different PL's.

First, you are required to think and design the logic to display an inverted triangle. Next, you are required to implement the logic in all the languages by modifying the code files in the `triangle` directory. Although it is not required to attain technical mastery in the PL's, you are expected to know the basic syntax as provided in the starter code.

You are required to compile the program first before executing it by following the commands in `run-commands` file in the `triangle` directory. At this point, you are expected to display an Inverted Triangle Shape printed on your screen, based on the user-provided number of rows. That is, similar to the line below:

Enter the number of rows: 5

```
* * * * *

* * * *

* * *

* *

*
```

Part 02 - Pyramid (20 points)

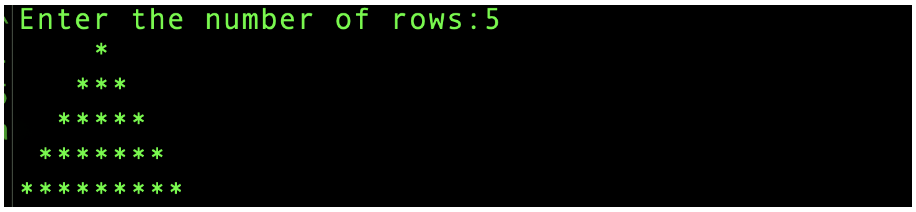
Modify the code files in the `pyramid directory` to implement a Pyramid shape. The programming solution in each language is worth 5 points each. To ease the process in coming up with the logical solution to the problem on your own, the starter code, in C Programming language is provided. Now, I do expect that you are able to address the challenges with the Syntax in each of these languages. Only Basic Syntax is required to solve this problem. The starter code has some examples on learning these basic syntax. If you have any questions related to Syntax, feel free to consult with the Technical Leader(s) and the Professor. You are welcome to explore your own logic to solve the problem and then revise the Logic portion of the C program accordingly. The program displays the Pyramid shape using the logic outlined in the pseudocode below:

```
1:  for ($i = 1; i <= rows; i = i + 1$) do
2:      for ($j = 0; j <= 2 * rows - 1; j = j + 1$) do
3:          if j >= rows - (i-1) and j <= rows + (i-1) then
4:              Print ("*")
5:          else
6:              Print space
7:          end if
8:      end for
9:      Print New Line
10: end for
```

Explore how the logic is implemented for triangle and pay attention to the syntax used in languages such as **C, C++, CSharp, Java, and Python**. Note: It is the same logic with similar semantics but the syntax is different in different PL's.

First, you are required to understand the logic to display the Pyramid using the C code. Next, you are required to implement the logic in all the other languages by modifying the code files in the `pyramid` directory. Although it is not required to attain technical mastery in the PL's, you are expected to know the basic syntax to solve the problem.

You are required to compile the program first before executing it by following the commands in `run-commands` file in the `pyramid` directory. At this point, you are expected to display the Pyramid Shape printed on your screen, based on the user-provided number of rows. That is, similar to the line below:



```
Enter the number of rows:5
    *
   ***
  *****
 *******
*********
```

Part 03 - Technical Report (10 points)

Modify the **technical-report** file in the lab directory to add a technical report to summarize the pros and cons of the programming languages, including, **C, C++, CSharp, Java, and Python**. To complete this step, conduct an online study and read through one or more articles published in websites and add at least two pros and cons for each programming language. Add the links to the online articles in the References section.

Part 04 - Honor Code

Make sure to **Sign** the following statement in the `honor-code.txt` file in your repository. To sign your name, simply replace Student Name with your name. The lab work will not be graded unless the honor code file is signed by you.

This work is mine unless otherwise cited - Student Name

Part 05 - Reflection

Add a Reflection to the repository by modifying the `reflection` file in the lab repository. List out the biggest learning points and any challenges that you have encountered during this lab.

PS next page ...

Submission Details

For this assignment, please submit the following to your GitHub repository by using the link shared to you by the Professor:

1. Modified source code in `triangle` directory.
2. Modified source code in `pyramid` directory.
3. A document containing the technical report of pros and cons of the different PL's in the file named `Technical-report.`.
4. A document containing the reflection of the lab in the file named `reflection`.
5. A signed honor code file, named `Honorcode`.
6. To reiterate, it is highly important, for you to meet the honor code standards provided by the college. The honor code policy can be accessed through the course syllabus.

Grading Rubric

1. There will be full points awarded for the lab if all the requirements in the lab specification are correctly implemented. Partial credits may be awarded if deemed appropriate.
2. Failure to upload the lab assignment code to your GitHub repository will lead to receiving no points given for the lab submission. In this case, there is no solid base to grade the work.
3. There will be no partial credit awarded if your code doesn't compile correctly. It is highly recommended to validate if the correct version of the code is being submitted before the due date and make sure to follow the honor code policy described in the syllabus. If it is a late submission, then it is the student's responsibility to let the professor know about it after the final submission in GitHub. In this way, an updated version of the student's submission will be used for grading. If the student did not communicate about the late submission, then automatically, the most updated version before the submission deadline will be used for grading purposes. If the student had not submitted any code, then, in this case, there are no points awarded to the student.
4. If a student needs any clarification on their lab grade, it is strongly recommended to talk to the Professor. The lab grade may be changed if deemed appropriate.