Lab 01 Specification – A Hand-on Exercise to practice Algorithm Development
50 points

Due by: 03/11/2021 8:00 AM

Lab Goals

- Quick review of GitHub and git commands.

- Refresh your ability to write algorithms.

- Think about how to solve algorithmic challenges.

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:

- Sedgewick chapter 01, section 1.4

Assignment Details

Now that we have discussed some basics of algorithms and analyzed multiple algorithms together in the last few lectures, it is now time to do it practically. In this lab, students will practice developing a variety of algorithms to retain the knowledge from the class discussions so far. This also includes analyzing one or more algorithms. 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.

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 algorithm(s) 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 test, exams, etc · · ·

Part 01 - A Simple Exercise to develop an algorithm formally (15 points)

A diurnal range is scientifically defined as the difference between the maximum and minimum temperature from the set of recordings taken in a day. There are several variations to define this. As per scientific articles referenced below, a large diurnal range can show a positive effect in Wine production. There are many wineries in the Erie area and hence it is great to look at this problem and solve it algorithmically. Develop an Algorithm (formally) to compute the diurnal range by using a set of temperature recordings for any particular day.
https://www.decanter.com/learn/what-is-diurnal-range-ask-decanter-413231/

1. For simplicity, a starter code with a file named `diurnal-algorithm.tex` is provided in the lab repository. The starter code has an example formal algorithm discussed and practiced during class discussions.

2. A latex file may be developed using the Overleaf website link provided below:

   https://www.overleaf.com/login

3. Please note: The Overleaf website may prompt to register and log in before providing the options to compile latex files and generate pdf files. This should be a straight forward process and have been tried out during our practical session 1. If there are any questions, students are encouraged to ask the Technical Leader(s) and the Professor.

4. Make necessary modification(s) to the file named `diurnal-algorithm.tex` to include the steps to compute the diurnal range. The computation should include steps such as finding the minimum, finding the maximum, and finding the difference between the minimum and maximum temperature recording. The difference is then returned as a result of the algorithm.

5. The algorithm should be developed in a formal manner using a similar style as we did in the class examples and practicals. Please refer back to the lecture slides, practical1 submission, and your class notes to look at the examples discussed in class.

6. It may be too tempting to propose a solution that uses two separate for loops (not nested). However, it is expected that the algorithmic solution submitted should contain only one for-loop.

7. **Hint:** One way of doing this is to start from the first value in the set of temperature recordings and compare the first value with the other values within the loop to identify the minimum and the maximum value. In this approach, the assumption is that the algorithm is restricted to only the values in the temperature recordings set. For example, we don't have an external access to the minimum and maximum values such as **Integer.MIN_VALUE** and **Integer.MAX_VALUE** within the algorithm.

8. It is required as part of this lab submission for students to compile the latex files and generate a PDF version of the file. The PDF file should be named as `diurnal-algorithm.pdf` and uploaded to the repository. Both the **tex** and **pdf** files will be used for grading purposes.

Part 02 - An Expense Tracker Algorithm (20 points)

Develop an algorithm to create an Expense Tracker using a series of requirements outlined below.

1. The starter code is provided inside the lab repository in a file named, `exp-tracker1.tex` and `exp-tracker1.pdf`.

2. First, analyze the algorithm provided in the starter code. Identify the asymptotic running time of the algorithm provided.

3. The algorithm takes in a set of expenses made by a person every day in a calendar month. The goal of the algorithm is to find the average for the expenses done for each day in the month by taking into account all the expenses till that day (inclusive). For example, to calculate the average of 5th day, we take the average of all the expenses from day (1 to 5). PS the example provided below to understand the problem better.

4. The algorithm provided in the starter code is not generally classified as the most efficient solution. Think about ways to solve this problem efficiently.

5. The algorithm in `exp-tracker1.tex` is also shown below to easily access it.

   Algorithm: ExpenseTracker(E)

   Input: An n-element array E of expenses each day.

   Output: An n-element array A of values such that A[i] is the average of elements E[0], E[1], ... , E[i]

   1: Initialize $a, i, j = 0$
   2: **for** $(i = 0; i < n; i = i + 1)$ **do**
   3:     $a \leftarrow 0$
   4:     **for** $(j = 0; j < i; j = j + 1)$ **do**
   5:         $a \leftarrow a + E[j]$
   6:     **end for**
   7:     $A[i] \leftarrow a/(i + 1)$
   8: **end for**
   9: **return A**

6. The efficient solution should solve this problem in linear time. The fine level details are purposefully left out in this sheet, to encourage students to think deeply and write their own algorithm.

7. The examples provided below is provided to further ease the understanding of the problem.

   input

30	20	10	40	50	90	180

   output

30	25	20	25	30	40	60

   input

45	15	30	14	16	48	35

   output

45	30	30	26	24	28	29

8. Create a new file named `exp-tracker2.tex` and include the formal algorithm of an efficient solution.

9. It is required as part of this lab submission for students to compile the latex files and generate a PDF version of the file. The PDF file should be named as `exp-tracker2.pdf` and uploaded to the repository. Both the **tex** and **pdf** files will be used for grading purposes.

Part 03 - To Solve (10 points)

An important part of algorithm developed is to analyze the effectiveness of the algorithm (asymptotically). The algorithms to analyze, and apply the asymptotic analysis technique are provided in the `algorithms.pdf` file. Answer the questions listed in the `algorithm-analysis.md` file. The solution to the questions listed should be provided as instructed in the **algorithm-analysis.md** file.

Part 04 - To Think (5 points)

Another important part of algorithm development is to develop thinking skills. By now, we developed one such application in the diurnal range. The key point in diurnal range calculation is computing the difference between the maximum and minimum from a list of values. Think and come up with an idea to propose a different application where finding such a difference is useful. The new idea should enrich and enhance what had been developed already. Include a summary of one or more ideas in a file named **ideas.md**.

Part 05 - 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

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. Commented source code from the "diurnal-algorithm.tex" program.

2. A document containing the PDF version of the algorithm, "diurnal-algorithm.pdf".

3. Commented source code from the "exp-tracker2.tex" program.

4. A document containing the PDF version of the algorithm, "exp-tracker2.pdf" .

5. A document containing the solution to the questions on asymptotic analysis, in a file named
   `algorithm-analysis.md`.

6. A document containing the ideas to enrich the algorithm developed to solve the diurnal range, in a file named
   `ideas.md`.

7. A signed honor code file, named `honor-code.txt`.

8. 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.