Lab 4 Specification – Exploring Internal Representation of Data
Due (via your git repo) no later than 2 p.m., Wednesday, 7th October 2020.
50 points

Lab Goals

- Practice a few data convertion techniques.
- Reflection on Pointers.
- More practice with C programming.

Summary

You will do a few exercises just to refresh your understanding and practice low-end data conversion techniques Assessment techniques, and extend the learning from in-class activities. We will watch a video clip, where the notion of pointers is discussed. We will take up an additional programming exercise to further expand our knowledge in C programming (in an un-explored low-end territory).

Learning Assignment

If you have not done so already, please read all of the relevant ”GitHub Guides”, available at https://guides.github.com/, which explains how to use many of the features that GitHub provides. In particular, please make sure that you have read guides such as ”Mastering Markdown” and ”Documenting Your Projects on GitHub”; each of them will help you to understand how to use both GitHub and GitHub Classroom. To do well on this assignment, you should also read

- Section 1.6 in **KR**
- Section 1.10 in **PH**
- Class discussion notes, slides, and in-class coding files.

Assignment Details

Now that we have discussed some fundamental principles behind low-end conversion techniques at the hardware-level, and got familiarized with the C programming language, it is now time to implement some challenging requirements of data conversion procedures. In this process, we will also implement a tool using a C program to convert a number from one format to another. Additionally, we will watch and reflect on a video to learn some intricacies of the C programming language, to retain the knowledge from the class discussions so far.

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) is strictly not allowed. Please note that all work done during laboratory 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 lab assignments, skill tests, projects, etc.

At any duration during and/or after the lab session, students are recommended to team up with the Professor and/or the Technical Leader(s) to clarify if there is any confusion related to the items in the lab sheet and/or class materials.

Section 1: Data Conversion Exercise

This section is worth 20 points. The points breakdown is provided below:

- Task 1 = 14 points (2 points each)
- Task 2 = 6 points (3 points each)

Data conversion is the conversion of data by computers at different levels of granularity. For example: data is displayed and represented in different ways, namely:

1. decimal for human users to read, and write (in code).
2. binary for machines to read and write.
3. hexa-decimal for managing memory, using a wider range of memory addresses.

In this section, we will practice the data conversion schemes using the algorithms discussed in lesson-4 (part-1) slides. To practice and retain knowledge from class discussions, it is required to complete the following tasks, outlined below:

- **Task 1:** Solve the following translation problems:
    1. Convert $(11101000)_2$ to Decimal
    2. Convert $(298)_{10}$ to Binary
    3. Convert $(BEAF)_{16}$ to Decimal
    4. Convert $(FACADE)_{16}$ to Decimal
    5. Convert $(987654321)_{10}$ to HexaDecimal

6. Convert $(DECADE)_{16}$ to Binary
7. Convert $(10001111011110101101 1101)_2$ to HexaDecimal

- **Task 2:** Answer the following questions to showcase your understanding of data representation at the internal level?

  Let us suppose that you have the following line in a C Program.

  unsigned int alpha = 100

  1. What is the maximum number that can be assigned to this unsigned integer variable.
  2. What is the minimum number that can be assigned to this unsigned integer variable.

- To complete the submission for this section, include your solution for both tasks, in a markdown file named `translate.md` and submit in your git repository.

Section 2: Reflection on Pointers

This section is worth 10 points. The points breakdown is provided below:

- Task 3 = 10 points, a maximum of 2 points awarded for each question.

In computer science, a pointer is an object in many programming languages that stores a memory address. This can be that of another value located in computer memory, or in some cases, that of memory-mapped computer hardware. As an analogy, a page number in a book's index could be considered a pointer to the corresponding page; a pointer is then used by flipping to the page with the given page number and reading the text found on that page. The actual format and content of a pointer variable is dependent on the underlying computer architecture. (text adapted from wiki). An Intro to Pointers is discussed in the video. To complete this part, it is required to do the following:

- **Task 3:** Watch this 10-minute video, by using the link below:

  `https://www.youtube.com/watch?v=h-HBipu_1P0`

  After watching the video, create a markdown file and name it as `video-reflection`. In this file, provide a detailed description of the questions presented below:

  1. What is a pointer?
  2. What is dereferencing? How is it related to indirection discussed in class?
  3. How is a pointer variable different from a regular variable?
  4. The video discussed the byte allocation of int, char, and float data types. How much is the memory allocated for (int, char, and float) in terms of bits resp?

5. How do computers differentiate memory address from the actual value stored in a variable? For example, explain what happens when we execute the following line

   int alpha = 100; in a C program?

Section 3: Data Converter

This section is worth 20 points. The points breakdown is provided below:

- Task 4 = 10 points
- Task 5 = 10 points

It is worth making a note, that the data conversion from decimal to binary (and vice-versa) is done by our computers, every nanosecond, tirelessly. In this part, we will implement two distinct methods in the Data Converter tool. These methods are expected to perform decimal to binary and binary to decimal conversions. To complete this part, it is required to have a solid understanding of the basic concepts of C Programming from class discussions and the previous two labs. It is important to review the slides, reason through and understand the logical outflow of the algorithms discussed in class, and also complete the reading assignments as required. There is a C program provided in the starter-code provided repository, to help stay focused on the implementation details, and to get started with the development of this part. To complete this part, it is required to do the following:

1. Review the starter-code in a file named `converter.c`. Execute the code a few times to understand the program flow in the code file. The overall goal of this program is to implement a data conversion tool that converts a decimal to binary (and vice-versa). Read the program completely, to make sure you understand how the program works. prompts to get the user input. This understanding is important to complete the tasks outlined below:
2. **Task 4:** Complete the **binaryToDecimal** method by implementing the following:
   - Introduce a new variable called temp (of integer type) and assign the value of binary to temp.
   - Iterate, with the use of a while loop, to compute the number of bits in the given input (that is in temp). This may involve dividing temp over 10 repeatedly.
   - Iterate, with the use of another new while loop, to implement the binary to a decimal algorithm. It is worth noting the highlight of this algorithm is that we need to get each bit separately, and multiply it by 2, and add to the result repeatedly. This process will end once we reach the last bit and we terminate the while loop. To implement this part, you may need to develop an if-else condition to implement the corresponding logic inside the while loop and insert the right values in the variable named `res`. By doing the above steps, this method should return the final output in a decimal format using the variable `res`.

3. **Task 5:** Complete the **binaryToDecimal** method by implementing the following:
   -
   - Iterate, with the use of a while loop, to compute the binary equivalent for the given decimal. You may need to repeatedly divide `decimal` over 2, till we cannot divide anymore.
   - To implement this part, you may need to develop an if-else condition to implement the corresponding logic inside the while loop and insert the right values in the variable named `res`. Hint: We need to use the modulo and pow operators to solve this part. The if-else condition may be used as an indicative factor to break out of the loop.

Submission Details

For this assignment, please submit the following to your GitHub lab repository.

1. **video-reflection** markdown file
2. upload of **translation.md** file
3. updated version of **converter.c** file
4. It is highly important, for you to meet the honor code standards provided by the college and to ensure that the submission is made before the deadline. The honor code policy can be accessed through the course syllabus. Make sure to add the statement ”This work is mine unless otherwise cited.” in all your deliverables such as source code and PDF files.
5. It is recommended to upload a summary file, with the details that you would like the Professor to know while grading the work. For example, it may be a reflection of your experience in the lab by highlighting some of the challenges that you had faced and a brief mention of how you addressed those challenges while implementing this lab. The summary file may also include a brief mention of any details that one should know about executing your program and what to expect during the execution.

Grading Rubric

1. Details including the points breakdown are provided in the individual sections above.
2. If a student needs any clarification on their lab credits, it is strongly recommended to talk to the Professor. The lab credits may be changed if deemed appropriate.