EE306 Introduction to Computing

Lab 4 (due on 4/9, 9pm, on GitHub)
Course Instructor: Dr. Nina Telang

All Lab assignments must be completed individually. You are not permitted to seek help or clarification from anyone other than the instructor or the TAs.

Your file should be named exactly after your EID, for example, xy1234.asm. Your program will not be graded if you fail to follow the file naming convention.

Problem Statement:

Consider the same test cases as Lab 3, where you were given a list of students with unique IDs and their number of credit hours, starting at location x4004. In Lab4 you will write a program in **LC-3 assembly language** to create to:

1. create a histogram, and
2. calculate the range and mean of the credit hours.

The following highlighted portion has been taken from Lab3 to emphasize that the test cases for Lab4 will be formatted the same way as Lab 3.

The list of credit hours completed by students, **integers** between 0 and 150 **(for the sake of this lab)**, along with the corresponding **unique non-zero** ID of the students will be stored in memory starting at address x4004. The ID and Credit hours of each student will be specified as follows:

- **Bits [15:8]** – Unique ID
- **Bits [7:0]** – Credit Hours Completed

The end of this list is specified using a Unique ID of **zero**, for which the Credit Hours field could be **anything**. (This entry merely acts as the sentinel or terminator.)

A histogram is a table representing the number of items that fall between a given set of intervals. We create the histogram based on the following intervals of credit hours (shown below is decimal) completed:

29 and Below - **'F' (Freshman)**

30 - 60 - **'S' (Sophomore)**

61 – 90 - **'J' (Junior)**

Above 90 - **'G' (Senior)**

The number of individuals falling in each of the 4 intervals is stored in consecutive memory locations starting at address **x4000**. Bits [7:0] contain the number of individuals in each interval, and bits [15:8] contain the identifier of each interval in ACSII, which are '**F**', '**S**', '**J**', and '**G**'.

Refer to the ASCII code chart to find the ASCII codes for 'F', 'S', 'J', 'G'.

Example: Tables 1 and 2 show content of memory locations corresponding to list and outputs before and after execution of program for a given list. Note that in this lab you are not expected to sort the list. You can, however, sort it (or use the program that you wrote in Lab3).

Table 1. State of memory before program is executed

Address	**Bits[15:8]** (in decimal)	**Bits[7:0]** (in decimal)
x4000	-	-
x4001	-	-
x4002	-	-
X4003	-	-
x4004	23	28
x4005	10	21
x4006	56	15
x4007	2	19
x4008	13	41
x4009	84	95
x400A	91	89
x400B	45	90
x400C	67	73
x400D	19	35
x400E	114	105
x400F	0	-

Contains data that will be overwritten by your program.

Table 2. State of memory after program is executed

Address	**Bits[15:8]** (in decimal)	**Bits[7:0]** (in decimal)
x4000	70	4
x4001	83	2
x4002	74	3
X4003	71	2
x4004	23	28
x4005	10	21
x4006	56	15
x4007	2	19
x4008	13	41
x4009	84	95
x400A	91	89
x400B	45	90
x400C	67	73
x400D	19	35
x400E	114	105
x400F	0	-

Note that these are the ASCII codes (in decimal) for ‘F’, ‘S’, ‘J’, ‘G’.

Report the statistics of the student list.

1. Range as a concatenation of highest number of credit hours in higher order byte (bits [15:8]) and lowest number of credit hours in lower order byte (bits [7:0]) at location x6000.
2. Mean of the credit hours as a 16-bit integer at location x6001.

If **mean has a fractional part, round it down to the lower integer**. For example, 55.545 can rounded to 55.

Table 3. State of memory after program is executed

Address	**Bits[15:8]** (in decimal)	**Bits[7:0]** (in decimal)
X6000	105	15
X6001	00	55

Notes

1. Unique IDs are all unsigned and non-zero.
2. If there are zero individuals in the list, the range and mean should be set as -1 (xFFFF).
3. If there is one individual in the list, both the highest and lowest number of credit hours will be same.
4. Start your program at location x3000. So, the first line of your assembly program must be .ORIG x3000.
5. You can test your program by manually loading data in locations x4004 onwards (with the last item in the list being the null ID number).
6. Note that the length of the list is not specified, though the maximum number of individuals is determined by the maximum possible unique IDs that can be specified in bits [15:8]. When you test your program for a variety of test cases make sure that you include a case with zero students in the list, and a case with the maximum number.

SUBMISSION INSTRUCTIONS:

The file that you will upload to online repository for this assignment must be named **youreid.asm.**

For example, if your eid is **ma123** then the file name should be ma123.asm)