EE2003 Computer Org. & Assembly Lang. Section L, M, N

Assignment 2 v1.1 Due date: 19/Mar/2023

Attention: You should submit the assignment by the due date. Late submission will incur a penalty for each day. No submissions will be accepted 7 days after the due date.

Task

Given a list of numbers, looking at the extreme values (minimum and maximum) sometimes does not give us a full picture. Because those extreme values might be outliers. It might be more helpful to see multiple values at both ends, and calculate their average. In this assignment, you will write an assembly program to get the average of highest three elements in an array, and also the average of lowest three elements in the same array.

You should complete this task in multiple steps.

(1)

Create a **`divide`** subroutine that divides first argument (dividend) by the second (divisor). The subroutine will return the integer quotient of division, discarding the remainder.

To perform the division, you should do repeated subtraction until the dividend gets smaller than the divisor. For example, dividing 23 by 5 will be done as 23 – 5 = 18, then 18 – 5 = 13, then 13 – 5 = 8, then 8 – 5 = 3 which is less than 5, so we stop the subtraction. The quotient would be 4 (number of iterations).

You can NOT use the built in `div` instruction. Assume that we are working with unsigned numbers only. Furthermore, do not worry about invalid arguments (say, dividend smaller than divisor).

(2)

Create a **`list_avg`** subroutine which takes in two parameters – an integer array (starting address) and its size. This subroutine will add all the numbers in the array, then divide the sum by array size to get the average. Average value is to be returned to caller via stack. Division should be performed by calling the divide subroutine.

(3)

Finally, write the main program which will have an array `marks` and its `size` available as global variables.

- To find the lowest and highest numbers in the array, the program will call the `bubblesort` subroutine (already provided to you).

- Then, it will call the `list_avg` subroutine on first three elements of the sorted array. Use the same `marks` array as the first argument and `3` as the second argument of `list_avg`.
- Next, the program will call `list_avg` for last three elements of the sorted array. You will need to get a pointer to 3rd-last element of `marks`, which will become the first argument of call to `list_avg`.

The main program should put the final results in `[topAvg]` and `[bottomAvg]` variables.

Note than the program should be generic, it should work correctly on other arrays of different size and values.

Important: For all subroutines, stack should be used to pass input arguments and to send return values back to caller. Study the document 'Notes on Stack and Subroutines' and follow the suggested pattern.

Deliverable

Download the skeleton file a2.asm from classroom, and write the missing code. bubblesort subroutine is already provided in this file. You need to write two other subroutines and the main program. Submit the completed asm file.

[15 marks]

Write explanatory comments in your program so that the reader can easily understand your code. Do not state the obvious in comments, e.g. for an instruction mov ax, 0, a comment like "move zero to ax" is useless, but a comment like "set sum as zero" is helpful because it tells the intent of the programmer.