

Coursework 1 – Assembly Language Programming

Deadline: Thursday 18th March at 17:00

Weighting: 15%

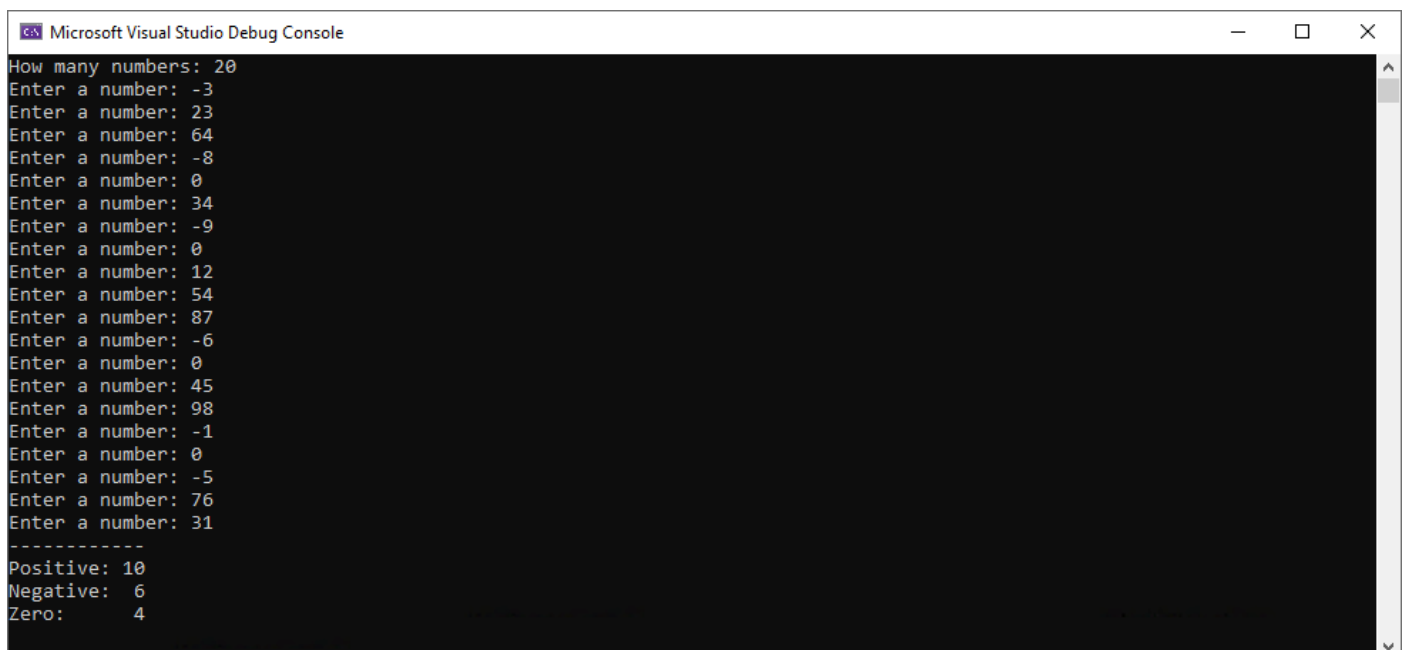
Follow the instructions at the end of this document to submit your work. Penalties for late work will be applied in accordance with the Code of Practice on Assessment. The standard department marking descriptors apply to this assessment.

Overview

The purpose of this assessment is to test your ability to write efficient assembly language code and describe clearly how it works. Write a small assembly language program that...

- Asks the user how many numbers they would like to enter
- Loops for the given number of times
- Prompts the user to input a number each time round the loop
- Displays a summary of how many positive, negative and zero numbers were entered

Your assembly code will be inside an `_asm` block within a C program, with variables declared using C syntax, as shown during the lab tasks. Create a new Visual Studio project from your existing template. The screenshot below shows the expected output of the program.



```
Microsoft Visual Studio Debug Console
How many numbers: 20
Enter a number: -3
Enter a number: 23
Enter a number: 64
Enter a number: -8
Enter a number: 0
Enter a number: 34
Enter a number: -9
Enter a number: 0
Enter a number: 12
Enter a number: 54
Enter a number: 87
Enter a number: -6
Enter a number: 0
Enter a number: 45
Enter a number: 98
Enter a number: -1
Enter a number: 0
Enter a number: -5
Enter a number: 76
Enter a number: 31
-----
Positive: 10
Negative: 6
Zero: 4
```

All the aspects of the program have been covered in the lab tasks, so your program will be based on code that you've already written. In particular, you will use code to output strings, input integers, test or compare register values, jump according to status flags, and loop according to a counter register. You should try to make your code as optimal and efficient as possible.

Useful Hint

This program will require a slightly more complex set of tests and jumps than the lab examples, because you are testing for three possible values (positive, negative, and zero). Write an initial version that detects only positive and negative numbers, because this will be easier. Submit this version if you can't work out how to detect zero values. You will still be able to obtain a passing mark for this assessment. A higher mark will be awarded if your code can detect all three input possibilities.

Code Comments & Structure

Use the C style comment notation (comments start with `//`) or the assembly language comment notation (comments start with `;`) to place useful explanations within the code. However, do not write lengthy comments that get in the way of readability. **Include your student ID as a comment at the top of your code.**

Assembly language doesn't make much use of indentation, but you should still make sure your code is easy to understand, including correct placement of any code labels. The C code containing your `_asm` block should be properly indented.

Short Report

You should also write a **short** report that describes how you implemented the loop, comparisons and jumps, and any optimisations you made. This should be just a couple of paragraphs at most. Do not provide excessive detail. If you encountered any problems (for example if your program doesn't work), write about that and show what you were trying to do. **Include your student ID at the top of your report.** Show that you tested your program with a variety of input, so **include a screenshot of your program running within the console window.** Paste this into your report so it's all in one document.

Marking Breakdown

Your work will be marked according to the following criteria.

Report:	15%	– Clarity, succinctness and understandability
Testing:	10%	– Evidence of testing with a range of inputs
Style:	10%	– Overall layout, presentation and structure of the code
Programming:	65%	– Correctness and efficiency of the code

How to Submit

Locate the Visual Studio project folder in your file system and navigate to the source file. It will have a `.cpp` extension. Refer back to the very first lab sheet for guidance on this. Copy this to another location so you don't corrupt your project, then change its name so it contains your username and student ID. Save your report as a PDF document with the same name. For example, if your username is **sgnabog** and your student ID is **201355426**, your files should be named:

- `sgnabog_201355426.cpp`
- `sgnabog_201355426.pdf`

Submit just these two files via the assessment page on Canvas. Please don't submit any other documents. We can only mark your work if we can easily locate and open the files. Please do not submit the entire Visual Studio project.