FIT2102 Tutorial Worksheet - WEEK 1

Each week we will complete a set of exercises that complement the material covered in lectures. Complete the activities below either in class, or if you run out of time then at home.

Complete the exercises below, compress the VS project directory into a single zip file, and submit it on Moodle.

You also will need to show your code and be able to adequately explain it to your tutor at the start of the following week's tute in order to receive a mark for the tutorial participation.

MASM Assembler and C

Learning Outcomes:

- Explain the need for abstraction from machine instructions to high-level languages
- Explain how assembly instructions work with registers and memory to perform computation
- Explain how assembly programs are structured into subroutines through jumps
- Create a basic x86 assembly program to perform a computation

Tasks:

1. If you have not already done so, then please install Visual Studio 2015 following these instructions.

2. Download the MASM starter project. Unzip it onto your local drive (do not use a shared drive) and open in VS.

3. Make sure it compiles and runs. If the program runs successfully the final message in the Debug output will be: "...exited with code 5 (0x5)".

FYI: Working through the problems at Project Euler is a fantastic way to learn a new language. The problems start off easy and gradually get more difficult. If you create an account on the site, when you enter the correct solution to a problem it will give you access to forum where people discuss the solutions to that problem in lots of different programming languages.

4. Project Euler Problem 1 reads:

   If we list all the natural numbers below 10 that are multiples of 3 or 5, we get

3, 5, 6 and 9. The sum of these multiples is 23.

Find the sum of all the multiples of 3 or 5 below 1000.

Go ahead and create a program based on the starter project, that solves this problem in MASM. To test whether a number is divisible by 3 or 5, create a procedure called IsDivisibleBy that checks if eax is evenly divisible by ebx using the div operator.

In developing your code, make sure to try out the Visual Studio debugger to step through the assembler line by line. See the values of registers change and set up watches for the variables you define in the .data block.

5. Right-click on your .asm file in the VS Solution Explorer and select “Exclude from Project”. Right-click on the project file and choose “Add->New Item” and create a new C++ file. Recode your solution to Project Euler Problem 1 in a simple vanilla C function, and create a main function that invokes it and returns the result.

6. View and use the debugger to step through the disassembly for your C program. Observe how the stack frame is set up, compare your disassembly to that listed in the course notes. Copy paste the disassembly into a text document and document line-by-line as in the course notes. Be ready to explain this code to your tutor at the start of next week’s tute.

7. Compile the above code in release mode and look at the disassembly carefully in the debugger. What are the differences?

8. Write a simple recursive C function to compute the sum of positive integers up to a given *n*.

9. What’s the biggest *n* you can specify before the stack overflow crash occurs (make sure you are running in debug mode)?

10. If the function is not already tail-recursive then rearrange it to make it so (the recursive call must be the very last operation in the function). The tail-recursive version will still crash in debug mode for large n, but try it again in Release mode. What happens? Why?