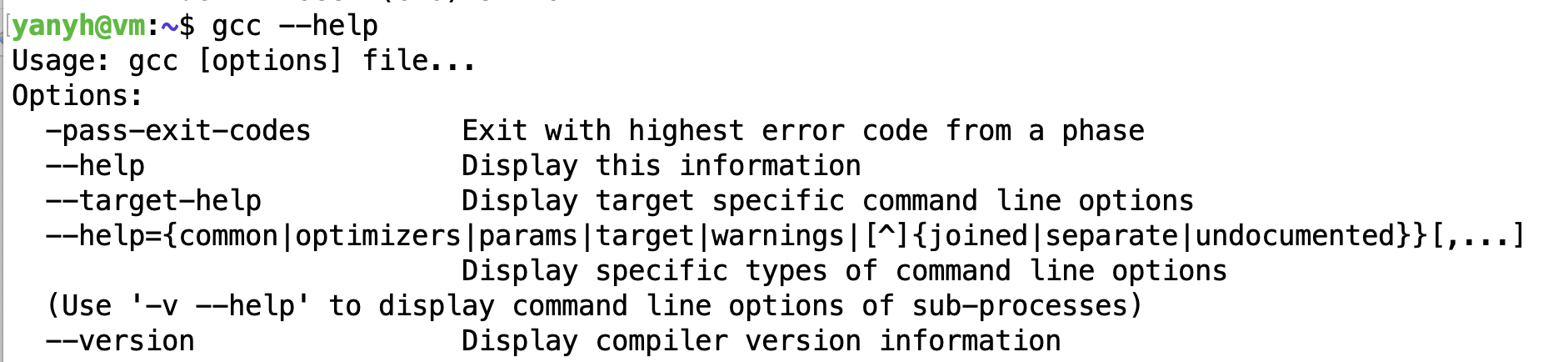
**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lab for ITSC 3181 Introduction to Computer Architecture, Spring 2023**

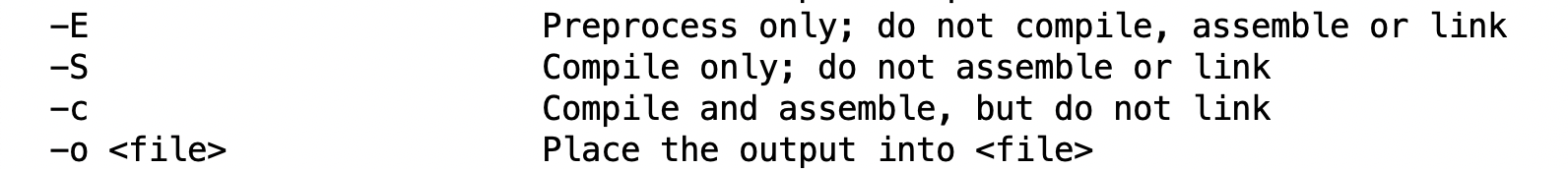
**Lab #01: Compiling a C program to produce its assembly output. Due Wednesday 1/18 and Total points: 1% of the final accumulated percentage grade.**

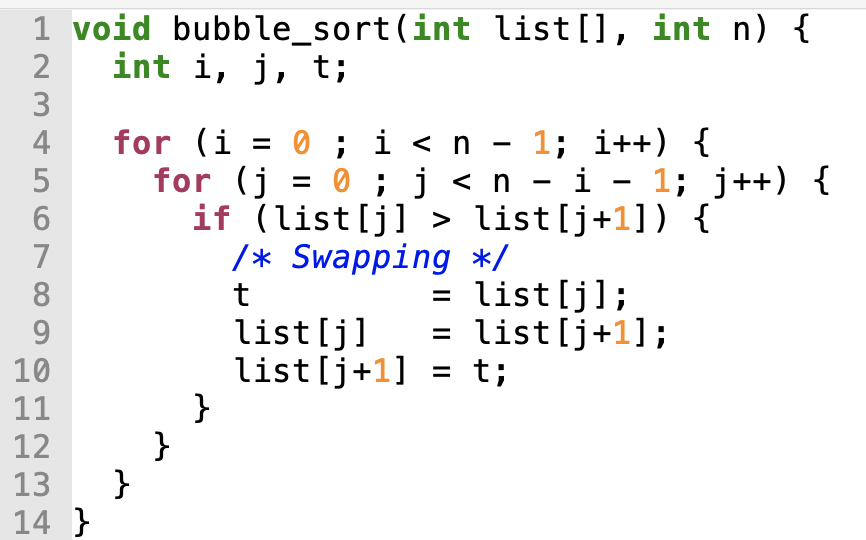
The development and lab environment <https://passlab.github.io/ITSC3181/resources/devenv.html>

1. Open <https://repl.it/languages/c> from a web browser. The left side of the interface is for you to type in your program. The Linux terminal environment is on the right side of the web interface, and get familiar with yourself using Linux commands from command lines (<https://passlab.github.io/ITSC3181/notes/lecture00_LinuxBasicsCompilation.pdf>). Inspect the output of the commands you type for execution.
2. Try “gcc –help” and “man gcc” command, which show how to use the gcc compiler to compile high-level program to machine code.



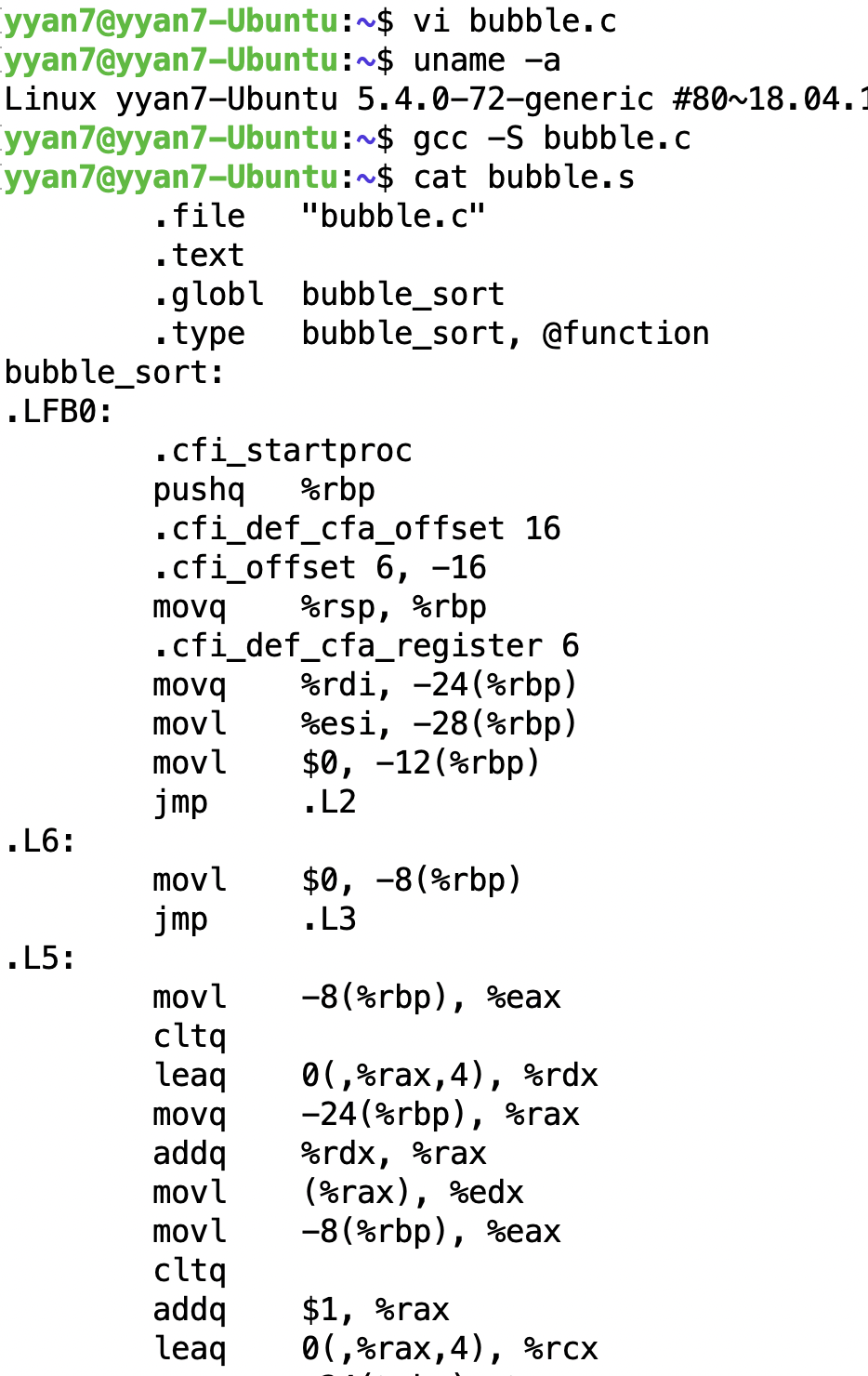
…



1. Create hello.c file from either the terminal (using vim editor or other editor that you are comfortable with) or from the <https://replit.com/languages/c> web interface, and type in the content of the file:
2. Compile and execute hello.c program.

gcc hello.c -o hello

./hello

1. Create bubble.c file and type in the content:
2. Compile the bubble.c file with -S option, which generate a bubble.s file for the ISA architecture of the machine (x86). Check the content of the bubble.s file. **Note: those lines with symbols that start with . (e.g. .file, .text, .cfi\_startproc), the label lines (those that end with :, e.g. bubble:, .LFB0: ), or those lines for function signature or comments are NOT instructions.**
3. Explore other ISA assembly from Compiler Explorer at <https://godbolt.org/> for the bubble.c example.

**Submission: Submit your work using the submission page that you can find from canvas for this lab. The submission page has the following table. You need count the number of instructions for the specified compiler and ISAs for bubble.c program and input the counts in the table. Labels (those ends with :) and directive (those starts with .) are not instructions, so do not count them. Instructions are normally highlighted with color and you can apply Filter in the right-side Pan of the web interface to sort out so only instructions show. You should count manually the number of instructions from the output when selecting the specified compilers from** [**https://godbolt.org**](https://godbolt.org)**.**

|  |  |  |
| --- | --- | --- |
|  | **Compiler and ISA** | **Number of instructions** |
| <https://godbolt.org/> | RISC-V rv32gc gcc latest |  |
| <https://godbolt.org/> | MIPS64 gcc 5.4 |  |
| <https://godbolt.org/> | x86-64 clang 12.0.0 |  |
| <https://godbolt.org/> | ARM gcc 8.2 |  |
| <https://godbolt.org/> | RISC-V rv32gc clang(trunk) |  |
| <https://godbolt.org/> | RISC-V rv64gc clang(trunk) |  |

The study shows that for the same high-level program, the instruction sequence for different compiler and different machine architecture (represented by its ISA) are very different. Even for the same ISA, the instruction sequences vary from different compilers. Instruction sequences also vary between 32-bit and 64-bit machines of the same ISA and of the same compiler.

**For TA to prepare students for this lab:**

1. Help students to be familiar with using <https://repl.it/languages/c> to create and edit a C program file, compiler and execute it. Explain that there are three panes in the interface and what each pane does.
2. Go through the Linux commands from command lines (<https://passlab.github.io/ITSC3181/notes/lecture00_LinuxBasicsCompilation.pdf>)
3. Help students to be familiar with compiler explorer <https://godbolt.org>. E.g. how to select the compiler used to compile a program.

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lab for ITSC 3181 Introduction to Computer Architecture, Spring 2023**

**Lab #01: Compiling a C program to produce its assembly output. Due Monday 9/05 and Total points: 1% of the final accumulated percentage grade.**

|  |  |  |
| --- | --- | --- |
|  | **Compiler and ISA** | **Number of instructions** |
| <https://godbolt.org/> | RISC-V rv32gc gcc latest |  |
| <https://godbolt.org/> | MIPS64 gcc 5.4 |  |
| <https://godbolt.org/> | x86-64 clang 12.0.0 |  |
| <https://godbolt.org/> | ARM gcc 8.2 |  |
| <https://godbolt.org/> | RISC-V rv32gc clang(trunk) |  |
| <https://godbolt.org/> | RISC-V rv64gc clang(trunk) |  |