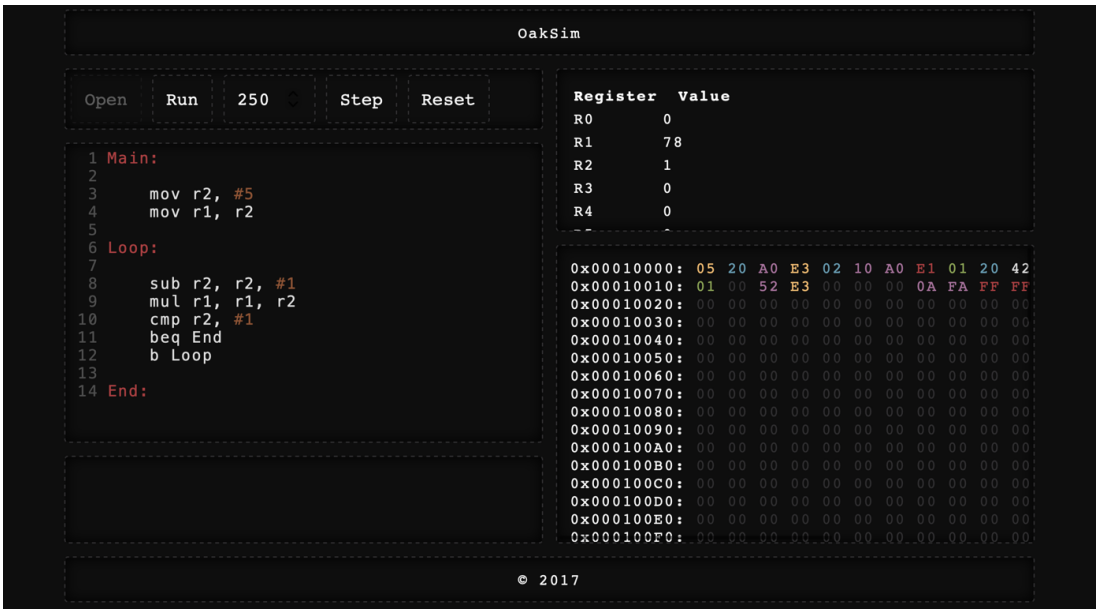
Template Week 4 – Software

Student number: 585303

Assignment 4.1: ARM assembly

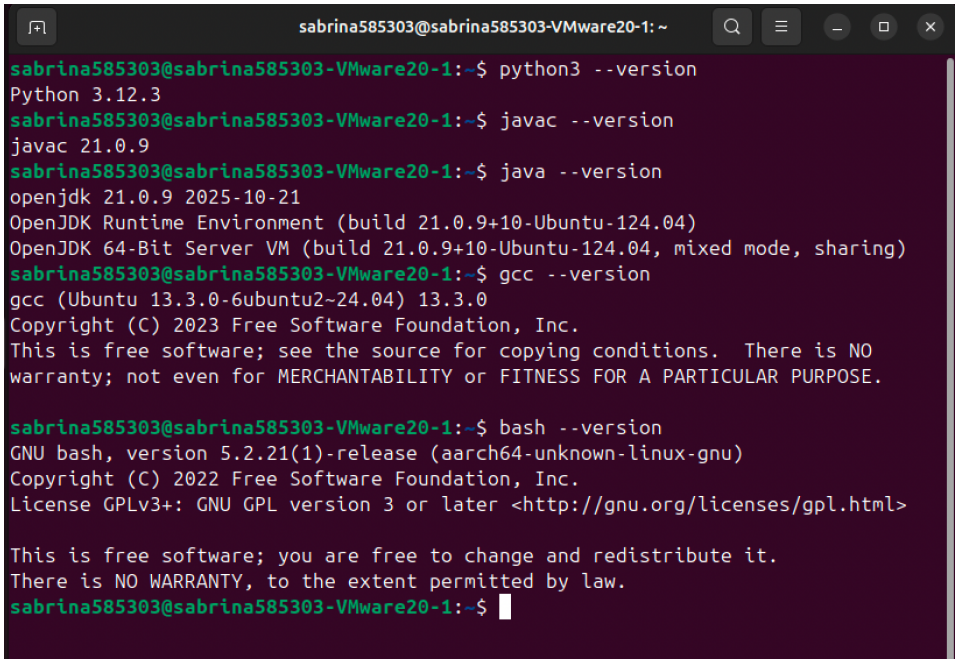
Screenshot of working assembly code of factorial calculation:



Assignment 4.2: Programming languages

Take screenshots that the following commands work:

javac –version, java –version, gcc –version, python3 –version, bash --version



Assignment 4.3: Compile

Which of the above files need to be compiled before you can run them?

- Fibonacci.java
- Fib.c

Which source code files are compiled into machine code and then directly executable by a processor?

- Fib.c

Which source code files are compiled to byte code?

- Fibonacci.java

Which source code files are interpreted by an interpreter?

- Fib.py
- Fib.sh

These source code files will perform the same calculation after compilation/interpretation. Which one is expected to do the calculation the fastest?

- Fib.c will be the fastest because is being translated into machine code by a compiler, which makes it able to be executed directly by the processor.

How do I run a Java program?

- Javac Fibonacci.java
- Java Fibonnaci

How do I run a Python program?

- Python3 fib.py

How do I run a C program?

- gcc fib.c -o fib
- ./fib

How do I run a Bash script?

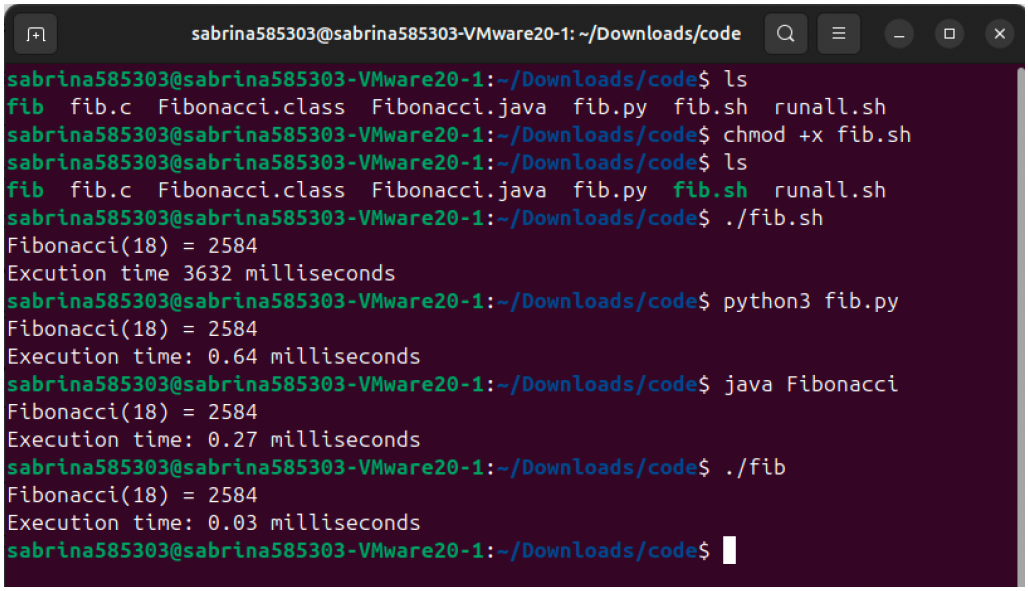
- chmod +x fib.sh
- ./fib.sh

If I compile the above source code, will a new file be created? If so, which file?

- Fibonacci.class for Fibonacci.java
- An executable file for fib.c and we name it fib

Take relevant screenshots of the following commands:

- Compile the source files where necessary
- Make them executable
- Run them
- Which (compiled) source code file performs the calculation the fastest?: the fib.c



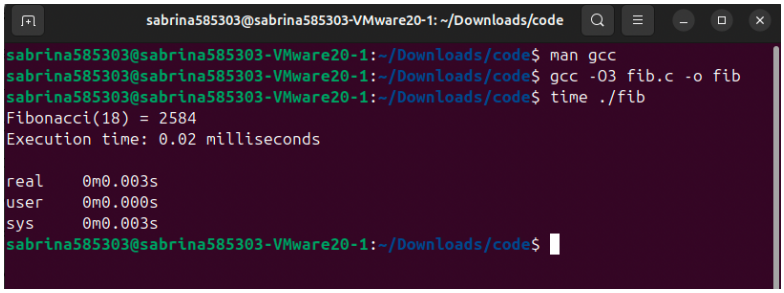
Assignment 4.4: Optimize

Take relevant screenshots of the following commands:

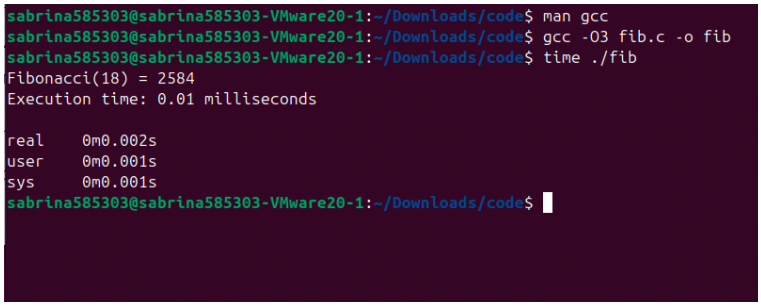
a) Figure out which parameters you need to pass to **the gcc** compiler so that the compiler performs a number of optimizations that will ensure that the compiled source code will run faster. **Tip!** The parameters are usually a letter followed by a number. Also read **page 191** of your book, but find a better optimization in the man pages. Please note that Linux is case sensitive.

- I will use the parameter gcc: -O3.
- It is a strong optimization level that tries to maximize the speed of the compiled program.
- **gcc -O3 fib.c -o fib**

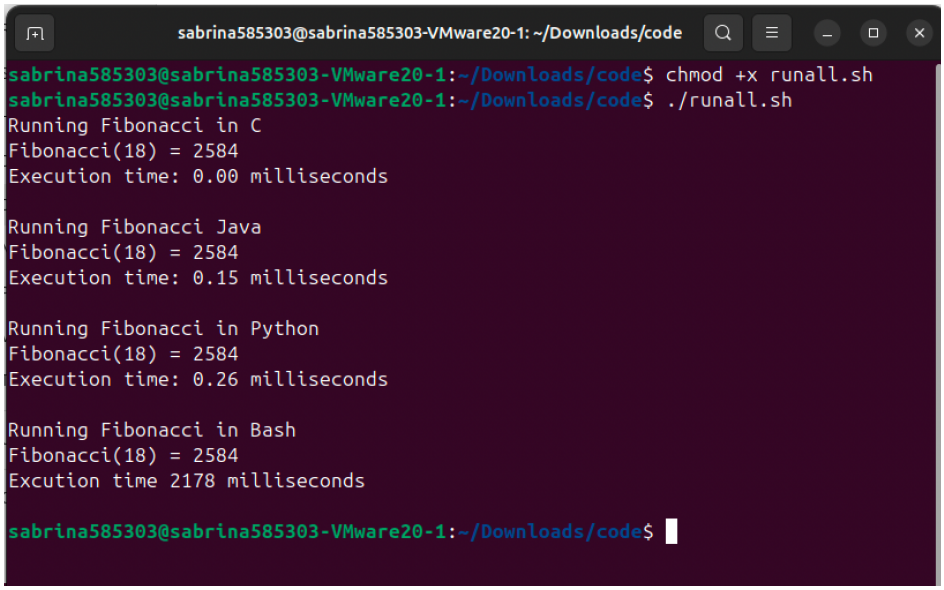
b) Compile **fib.c** again with the optimization parameters



c) Run the newly compiled program. Is it true that it now performs the calculation faster?



d) Edit the file **runall.sh**, so you can perform all four calculations in a row using this Bash script. So the (compiled/interpreted) C, Java, Python and Bash versions of Fibonacci one after the other.



Assignment 4.5: More ARM Assembly

Like the factorial example, you can also implement the calculation of a power of 2 in assembly. For example you want to calculate $2^4 = 16$. Use iteration to calculate the result. Store the result in r0.

```
Main:

mov r1, #2

mov r2, #4


Loop:


End:
```