Template Week 4 – Software

Student number:582777

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:

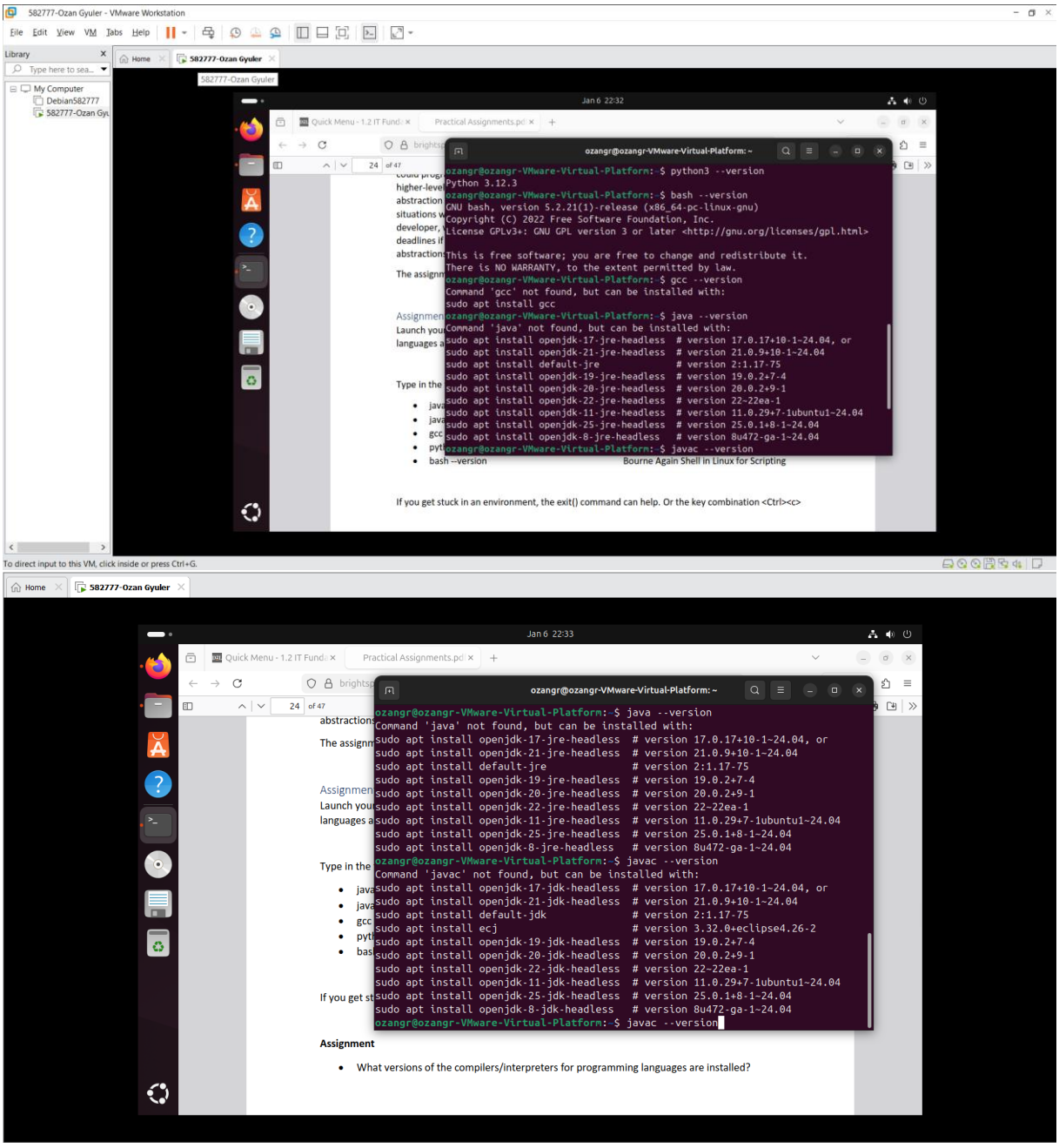
javac --version

java --version

gcc --version

python3 --version

bash --version


582777-Ozan Gyuler - VMware Workstation
ozangr@ozangr-VMware-Virtual-Platform: ~
ozangr@ozangr-VMware-Virtual-Platform:~$ python3 --version
Python 3.12.3
ozangr@ozangr-VMware-Virtual-Platform:~$ bash --version
GNU bash, version 5.2.21(1)-release (x86_64-pc-linux-gnu)
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
ozangr@ozangr-VMware-Virtual-Platform:~$ gcc --version
Command 'gcc' not found, but can be installed with:
sudo apt install gcc
ozangr@ozangr-VMware-Virtual-Platform:~$ java --version
Command 'java' not found, but can be installed with:
sudo apt install openjdk-17-jre-headless # version 17.0.17+10-1~24.04, or
sudo apt install openjdk-21-jre-headless # version 21.0.9+10-1~24.04
sudo apt install default-jre # version 2:1.17-75
sudo apt install openjdk-19-jre-headless # version 19.0.2+7-4
sudo apt install openjdk-20-jre-headless # version 20.0.2+9-1
sudo apt install openjdk-22-jre-headless # version 22~22ea-1
sudo apt install openjdk-11-jre-headless # version 11.0.29+7-1ubuntu1~24.04
sudo apt install openjdk-25-jre-headless # version 25.0.1+8-1~24.04
sudo apt install openjdk-8-jre-headless # version 8u472-ga-1~24.04
ozangr@ozangr-VMware-Virtual-Platform:~$ javac --version
bash --version
Bourne Again Shell in Linux for Scripting
If you get stuck in an environment, the exit() command can help. Or the key combination <Ctrl><c>
ozangr@ozangr-VMware-Virtual-Platform:~$ java --version
ozangr@ozangr-VMware-Virtual-Platform:~$ javac --version
Assignment
What versions of the compilers/interpreters for programming languages are installed?

Assignment 4.3: Compile

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

Fibonacci.java and fib.c needs compilation.

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

fib.c

Compiled by a C compiler (e.g., gcc) directly into native machine code.

The resulting executable runs directly on the CPU.

Which source code files are compiled to byte code?

Fibonacci.java compiled by javac into Java bytecode.

Which source code files are interpreted by an interpreter?

fib.py **-** Interpreted by the Python interpreter

fib.sh -Interpreted by a shell interpreter (e.g., bash)

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

C (fib.c) - Compiled to native machine code

How do I run a Java program?

java Fibonacci

How do I run a Python program?

python3 fib.py - python fib.py depends on the system

How do I run a C program?

./fib

How do I run a Bash script?

chmod +x fib.sh make executable (once)

./fib.sh run or bash fib.sh run

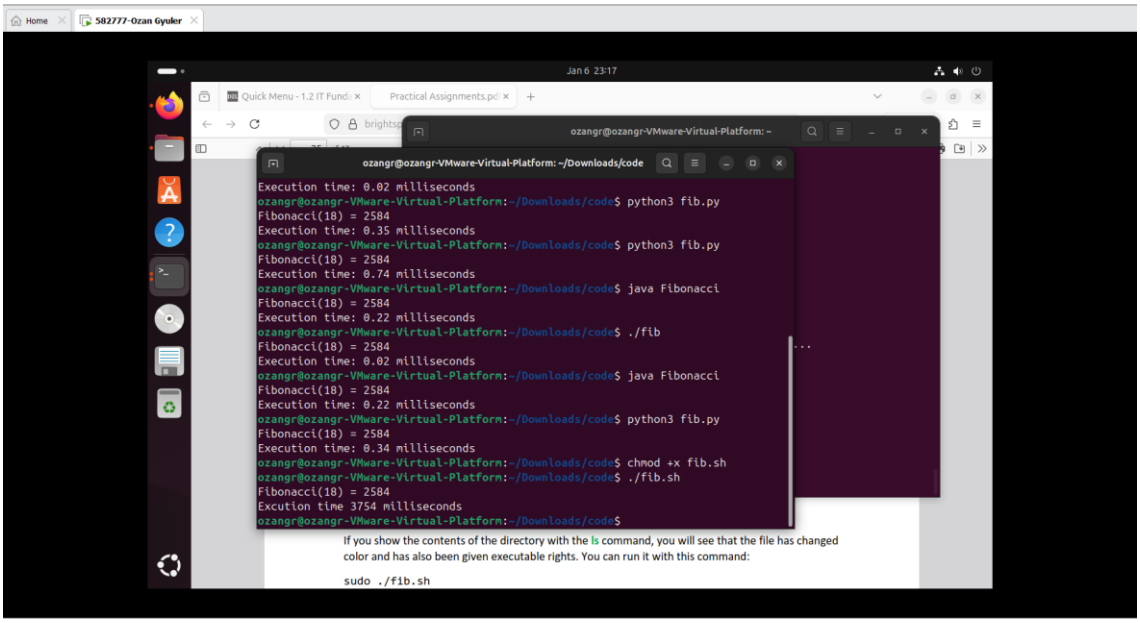
If I compile the above source code, will a new file be created? If so, which file?

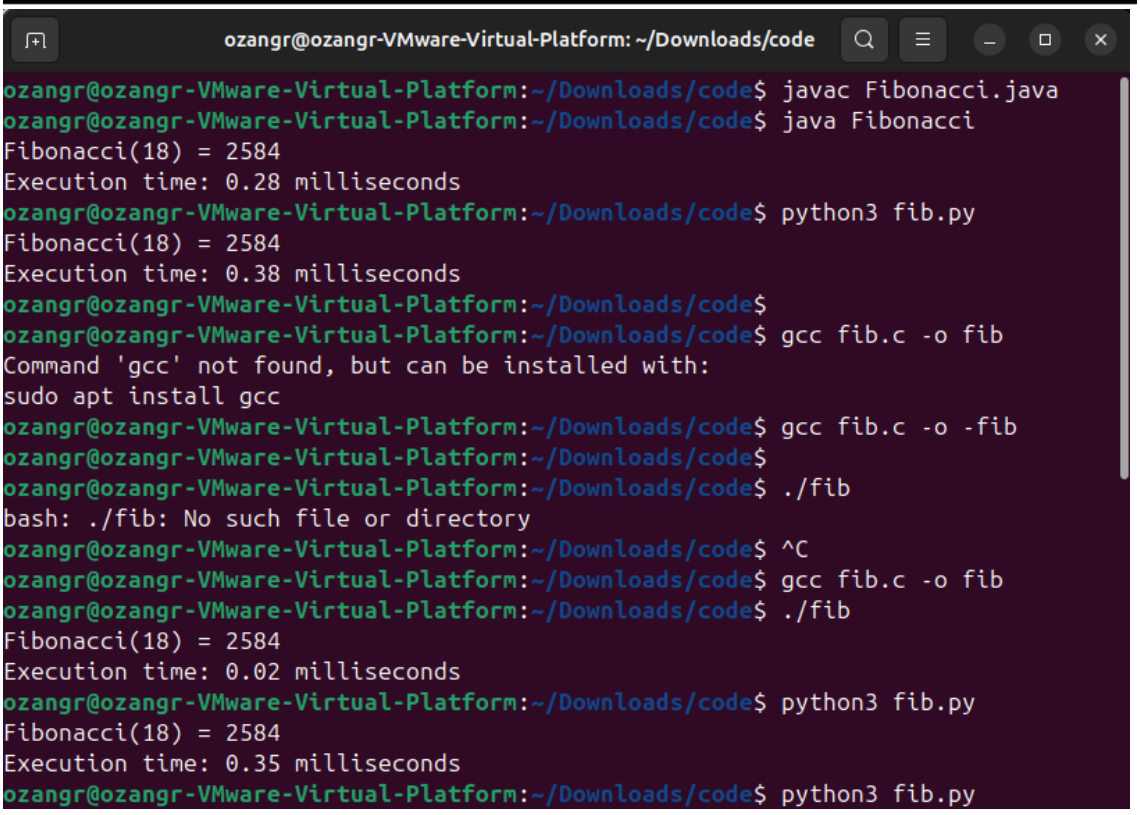
fib.c -- fib (or fib.exe on Windows)

Fibonacci.java -- Fibonacci.class

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?
  fib.c is the fastest.

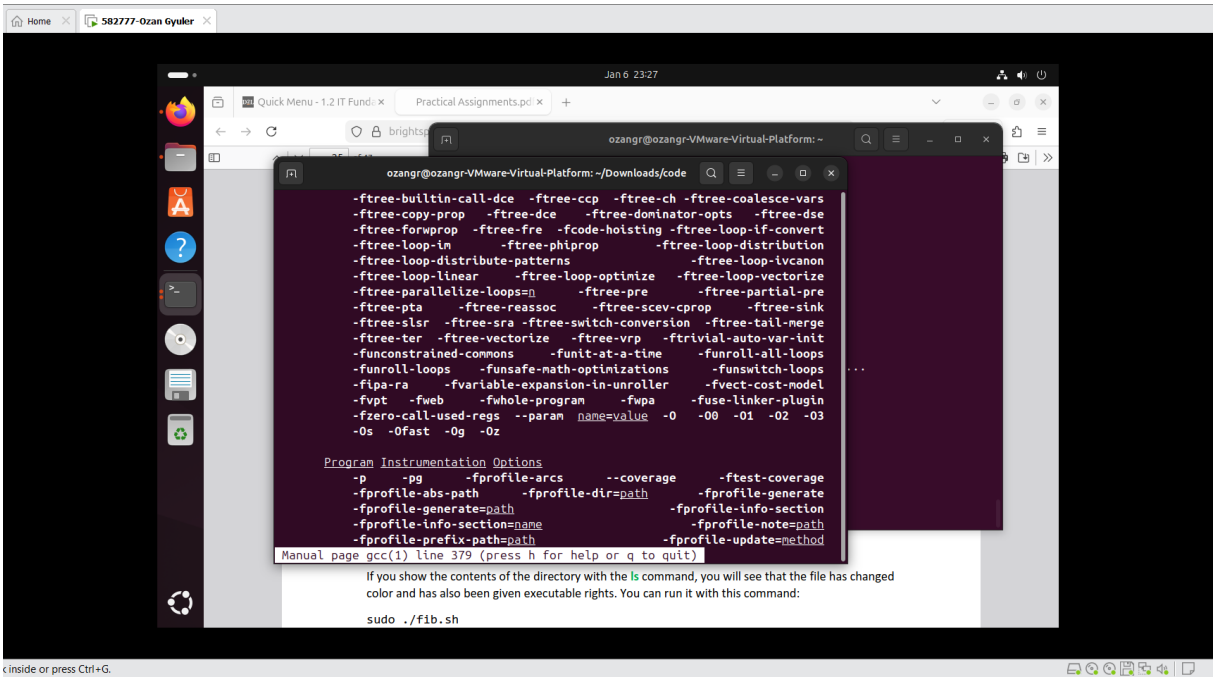




Assignment 4.4: Optimize

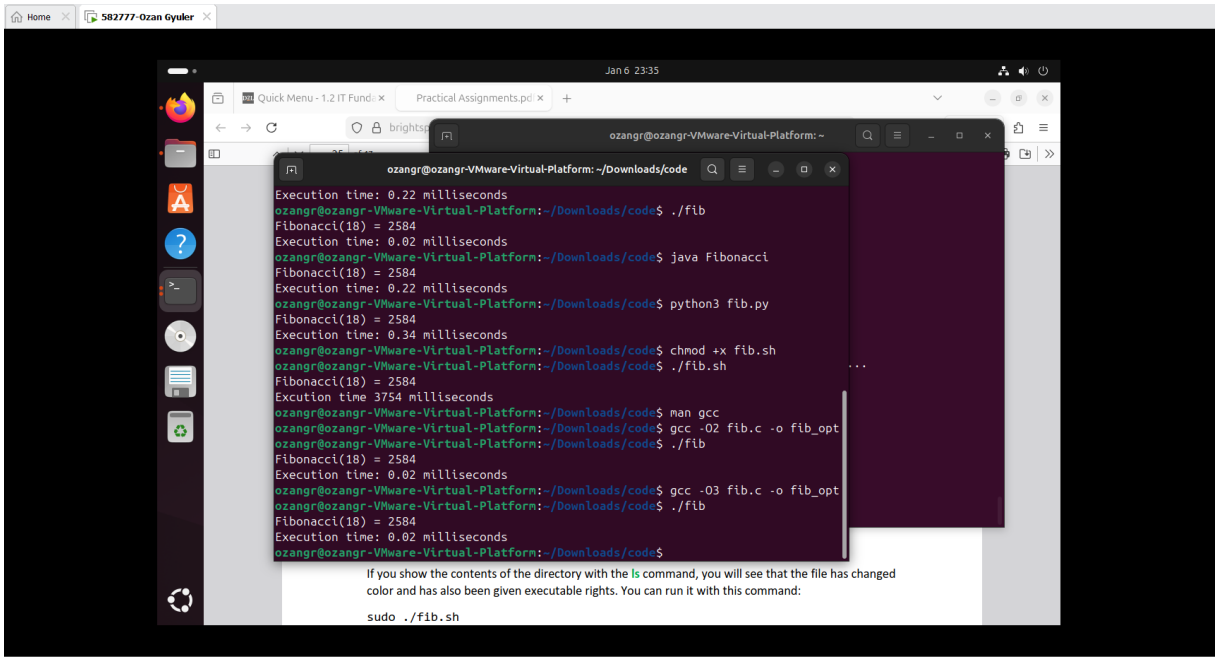
Take relevant screenshots of the following commands:

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.

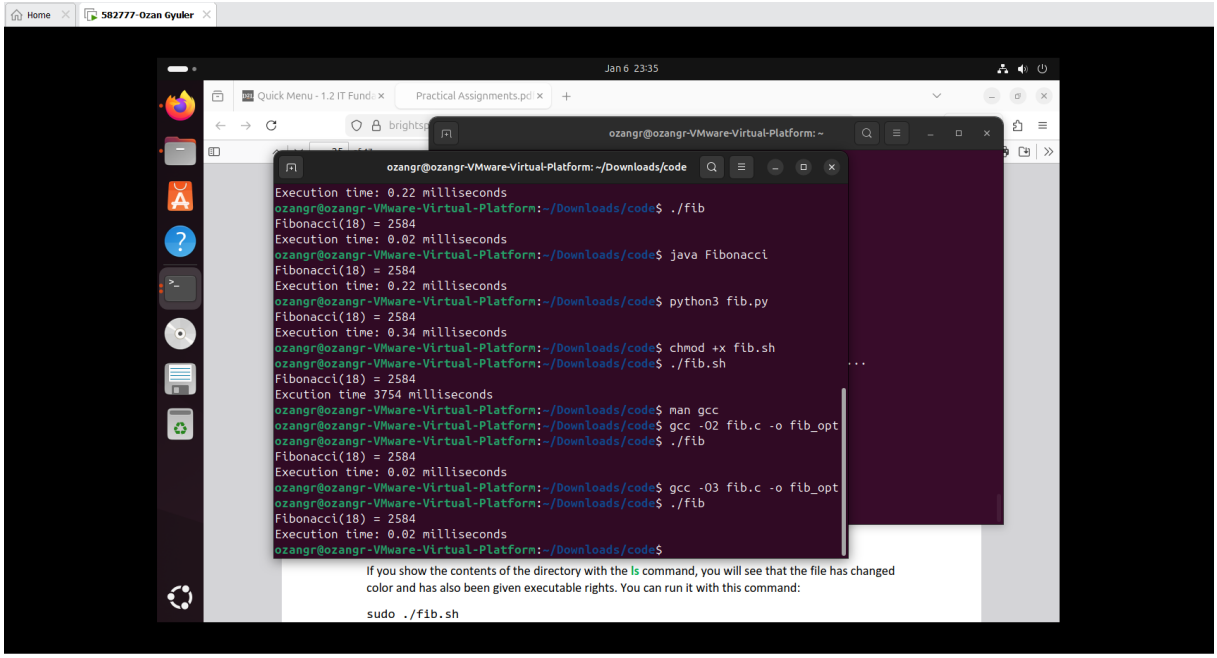


Ofast is fast but not reliable and O2-O3 is more safe way to make it faster

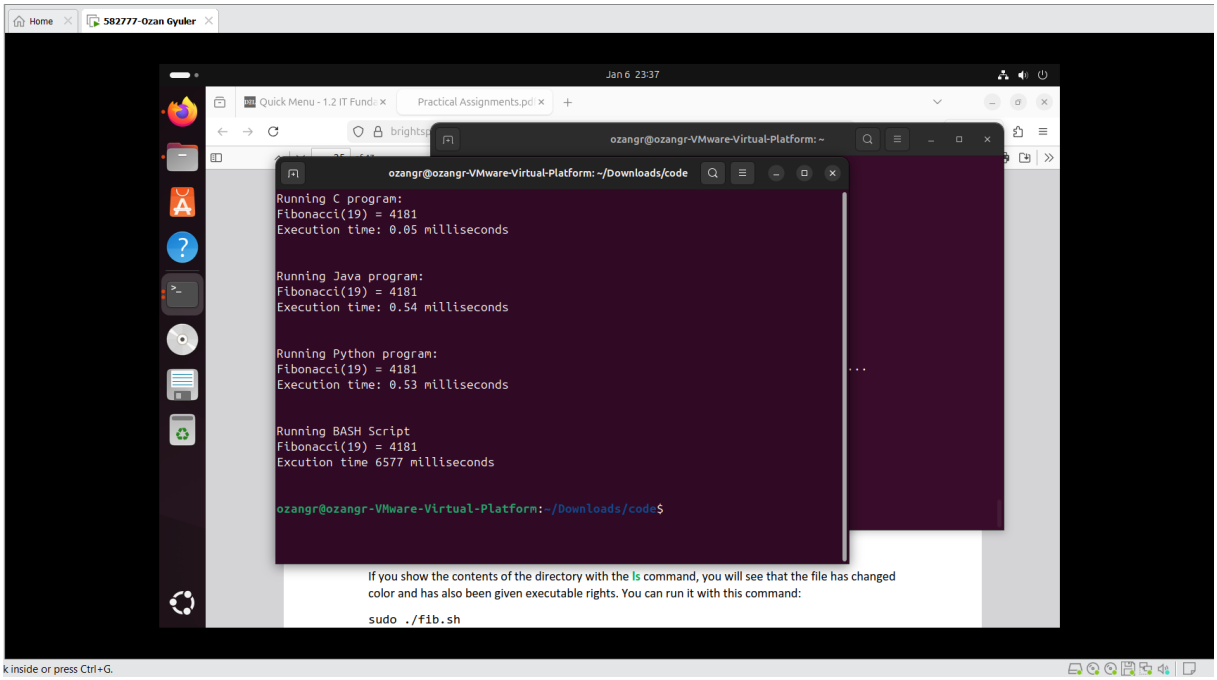
b) Compile **fib.c** again with the optimization parameters



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.



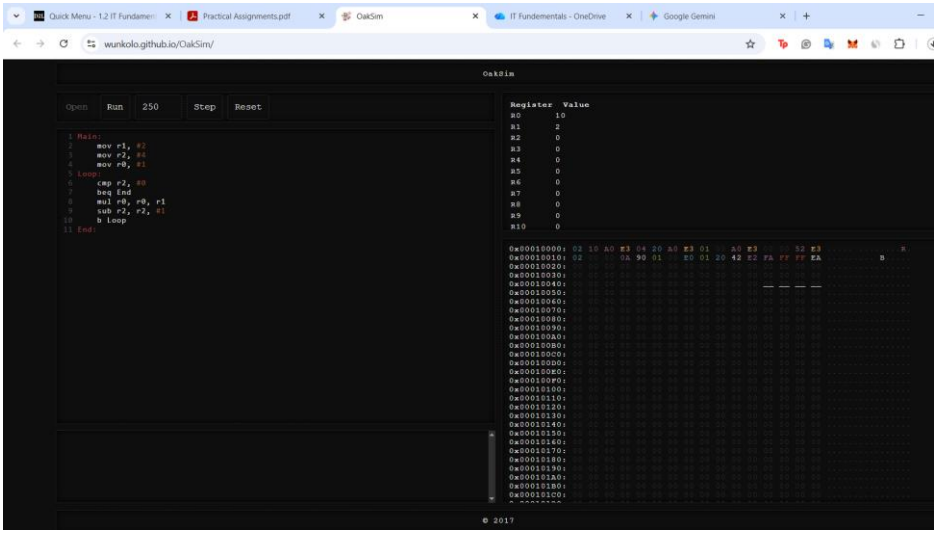
Assignment 4.5: More ARM Assembly

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
    mov r0, #1
Loop:
    cmp r2, #0
    beq End
    mul r0, r0, r1
    sub r2, r2, #1
    b Loop
End:
```

Complete the code. See the PowerPoint slides of week 4.

Screenshot of the completed code here.



Ready? Save this file and export it as a pdf file with the name: **week4.pdf**