Template Week 4 – Software

Student number:574642

Assignment 4.1: ARM assembly

Screenshot of working assembly code of factorial calculation:

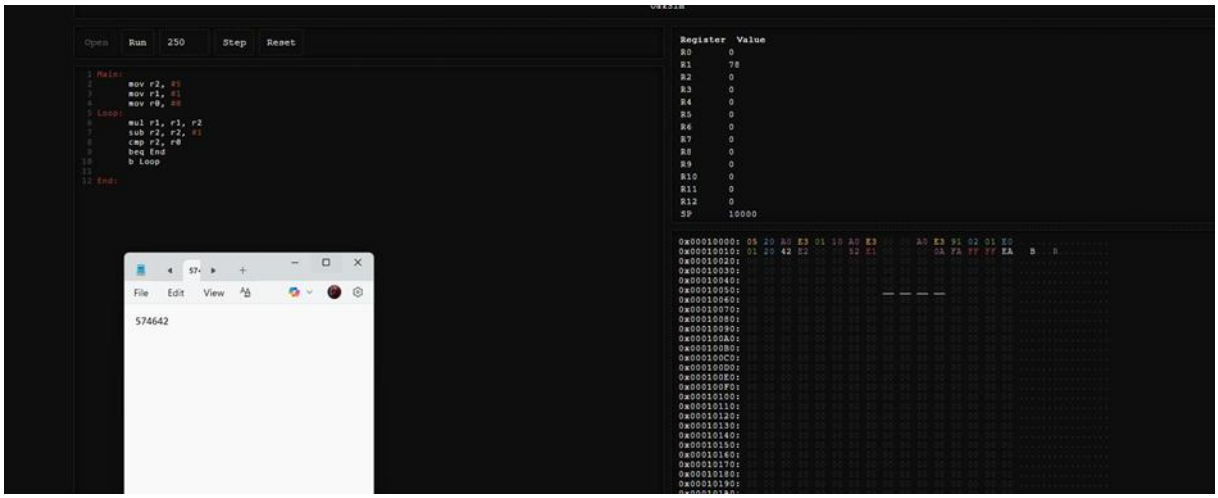
Na 1 keer runnen krijg ik

R0 = 2

R1 = 4

R2 = 2

R3 = F



Assignment 4.2: Programming languages

Take screenshots that the following commands work:

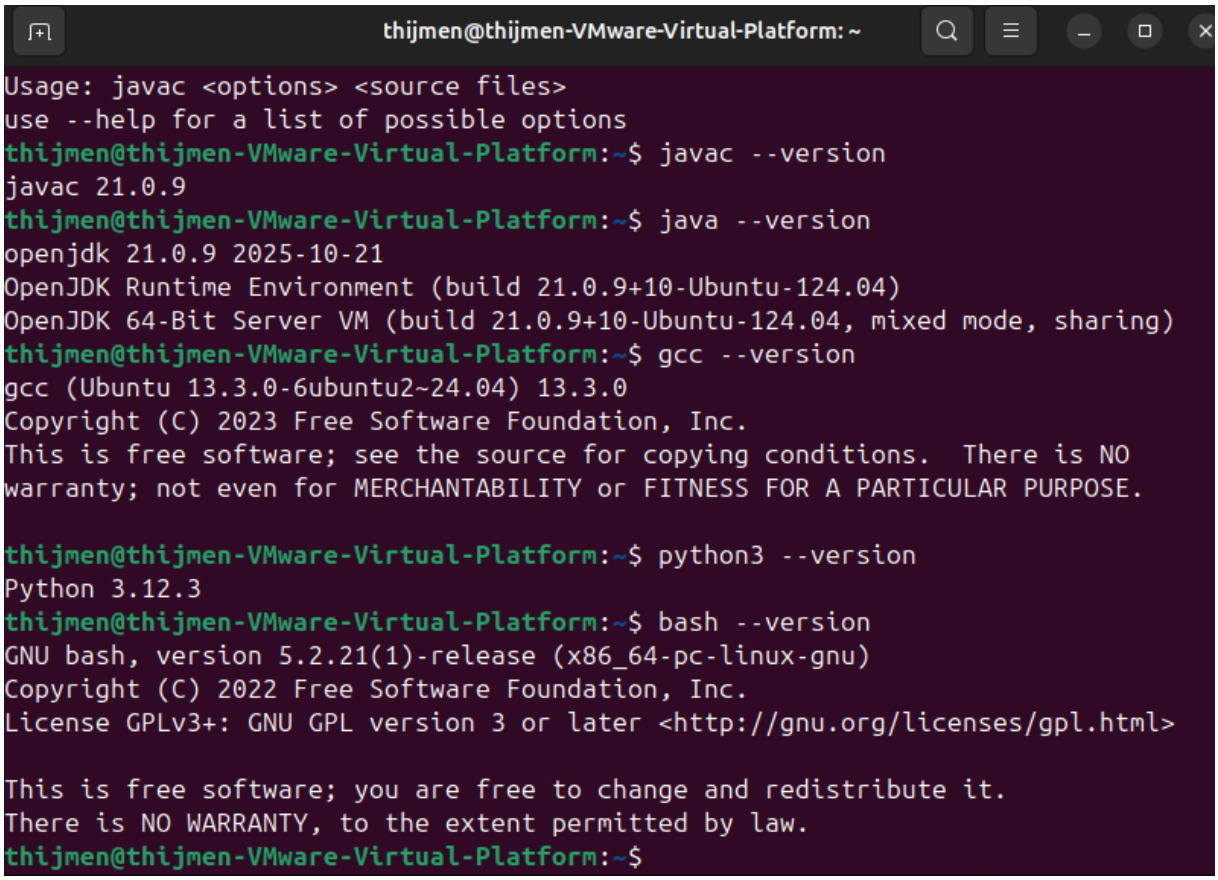
javac --version

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?

Fibonaci.java en 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 omdat die gelijk uitvoerbaar is door de proceccor na het compilen

How do I run a Java program?

Javac fibonacci.java compileren > java fibonacci runnen

How do I run a Python program?

Python3 fib.py

How do I run a C program?

Gcc fib.c -o fib compileren > ./fib runnen

How do I run a Bash script?

Chmod a+x fib.sh uitvoerbaar maken > ./fib.sh runnen

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

Fibonacci.java > fibonacci.class

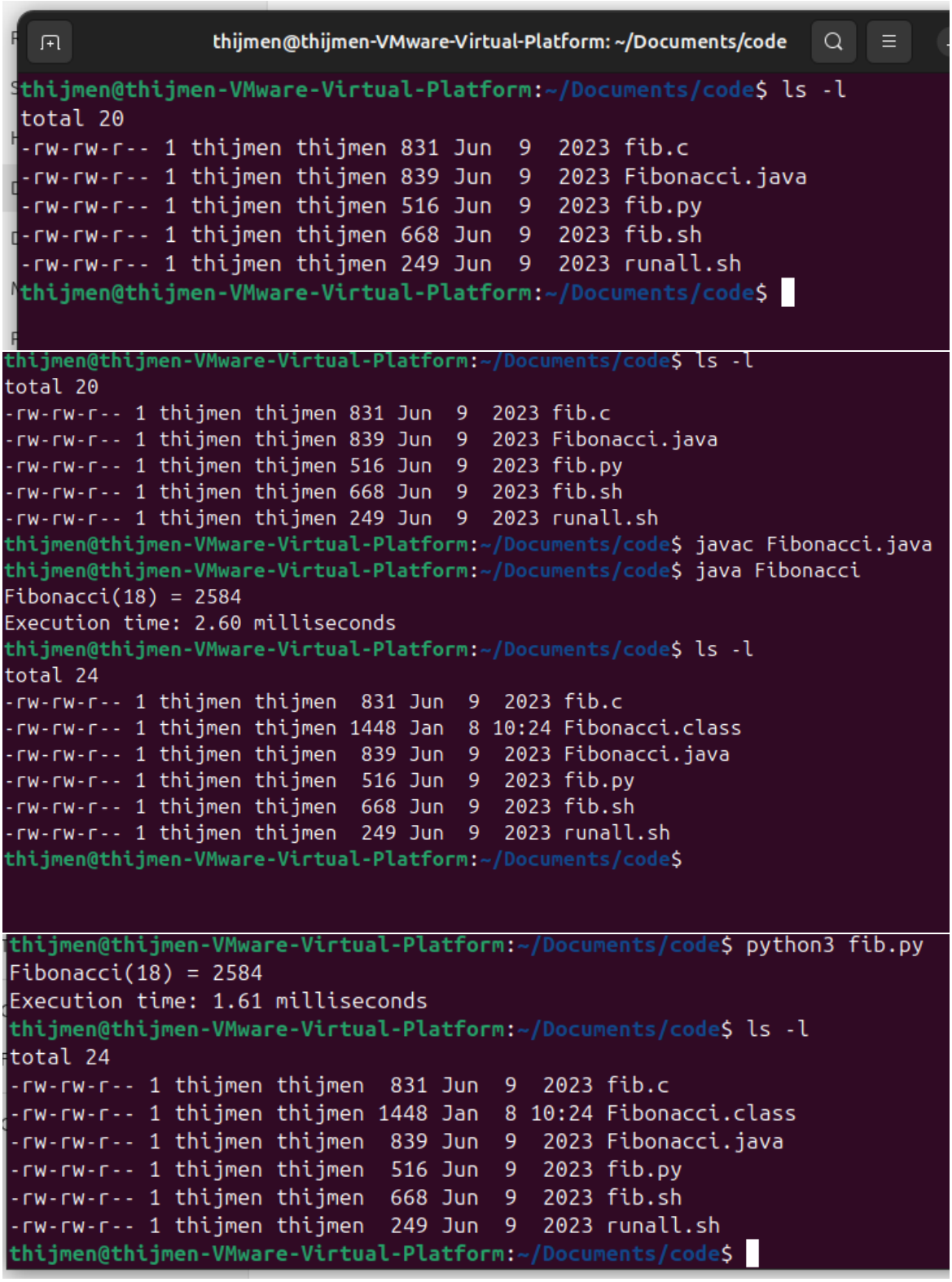
Fib.c > fib

Fib.py > zelfde naam

Fib.sh > zelfde naam

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?

- 
thijmen@thijmen-VMware-Virtual-Platform: ~/Documents/code
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ls -l
total 20
-rw-rw-r-- 1 thijmen thijmen 831 Jun 9 2023 fib.c
-rw-rw-r-- 1 thijmen thijmen 839 Jun 9 2023 Fibonacci.java
-rw-rw-r-- 1 thijmen thijmen 516 Jun 9 2023 fib.py
-rw-rw-r-- 1 thijmen thijmen 668 Jun 9 2023 fib.sh
-rw-rw-r-- 1 thijmen thijmen 249 Jun 9 2023 runall.sh
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ls -l
total 20
-rw-rw-r-- 1 thijmen thijmen 831 Jun 9 2023 fib.c
-rw-rw-r-- 1 thijmen thijmen 839 Jun 9 2023 Fibonacci.java
-rw-rw-r-- 1 thijmen thijmen 516 Jun 9 2023 fib.py
-rw-rw-r-- 1 thijmen thijmen 668 Jun 9 2023 fib.sh
-rw-rw-r-- 1 thijmen thijmen 249 Jun 9 2023 runall.sh
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ javac Fibonacci.java
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ java Fibonacci
Fibonacci(18) = 2584
Execution time: 2.60 milliseconds
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ls -l
total 24
-rw-rw-r-- 1 thijmen thijmen 831 Jun 9 2023 fib.c
-rw-rw-r-- 1 thijmen thijmen 1448 Jan 8 10:24 Fibonacci.class
-rw-rw-r-- 1 thijmen thijmen 839 Jun 9 2023 Fibonacci.java
-rw-rw-r-- 1 thijmen thijmen 516 Jun 9 2023 fib.py
-rw-rw-r-- 1 thijmen thijmen 668 Jun 9 2023 fib.sh
-rw-rw-r-- 1 thijmen thijmen 249 Jun 9 2023 runall.sh
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ python3 fib.py
Fibonacci(18) = 2584
Execution time: 1.61 milliseconds
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ls -l
total 24
-rw-rw-r-- 1 thijmen thijmen 831 Jun 9 2023 fib.c
-rw-rw-r-- 1 thijmen thijmen 1448 Jan 8 10:24 Fibonacci.class
-rw-rw-r-- 1 thijmen thijmen 839 Jun 9 2023 Fibonacci.java
-rw-rw-r-- 1 thijmen thijmen 516 Jun 9 2023 fib.py
-rw-rw-r-- 1 thijmen thijmen 668 Jun 9 2023 fib.sh
-rw-rw-r-- 1 thijmen thijmen 249 Jun 9 2023 runall.sh
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$

```
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ chmod a+x fib.sh
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ./fib.sh
Fibonacci(18) = 2584
Excution time 17526 milliseconds
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ls -l
total 40
-rwxrwxr-x 1 thijmen thijmen 16136 Jan  8 10:26 fib
-rw-rw-r-- 1 thijmen thijmen   831 Jun  9  2023 fib.c
-rw-rw-r-- 1 thijmen thijmen  1448 Jan  8 10:24 Fibonacci.class
-rw-rw-r-- 1 thijmen thijmen   839 Jun  9  2023 Fibonacci.java
-rw-rw-r-- 1 thijmen thijmen   516 Jun  9  2023 fib.py
-rwxrwxr-x 1 thijmen thijmen   668 Jun  9  2023 fib.sh
-rw-rw-r-- 1 thijmen thijmen   249 Jun  9  2023 runall.sh
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$
```

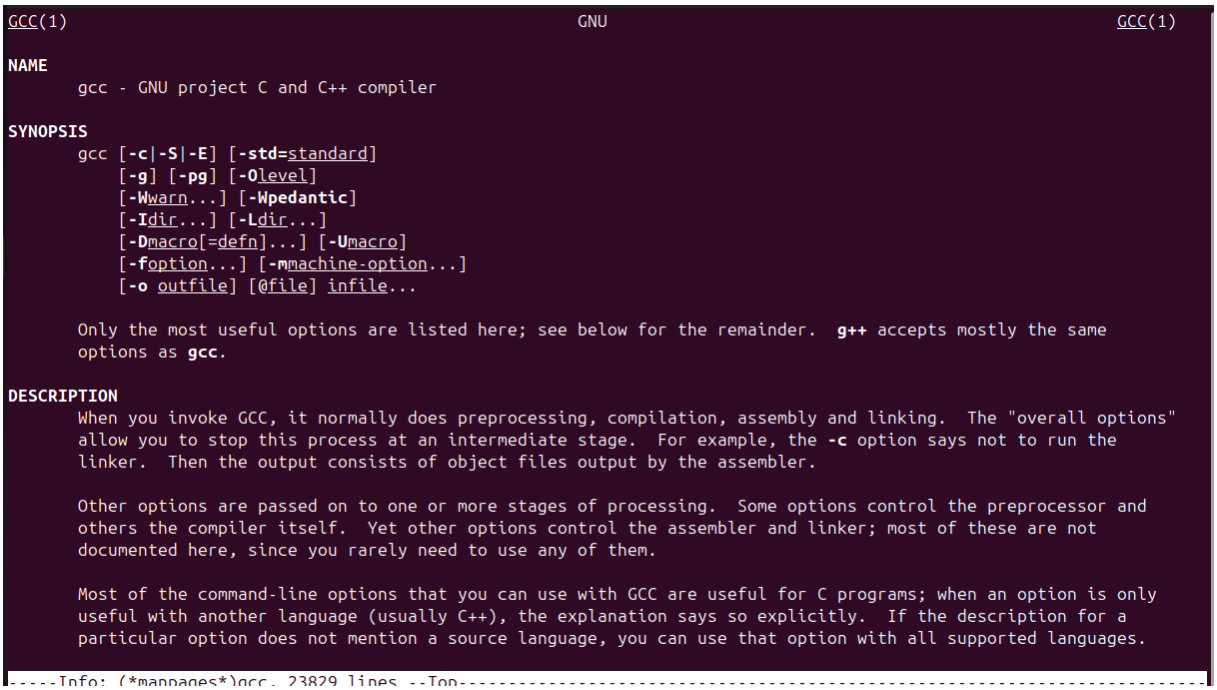
```
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ gcc fib.c -o fib
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ./fib
Fibonacci(18) = 2584
Execution time: 0.04 milliseconds
thijmen@thijmen-VMware-Virtual-Platform:~/Documents/code$ ls -l
total 40
-rwxrwxr-x 1 thijmen thijmen 16136 Jan  8 10:26 fib
-rw-rw-r-- 1 thijmen thijmen   831 Jun  9  2023 fib.c
-rw-rw-r-- 1 thijmen thijmen  1448 Jan  8 10:24 Fibonacci.class
-rw-rw-r-- 1 thijmen thijmen   839 Jun  9  2023 Fibonacci.java
-rw-rw-r-- 1 thijmen thijmen   516 Jun  9  2023 fib.py
-rw-rw-r-- 1 thijmen thijmen   668 Jun  9  2023 fib.sh
-rw-rw-r-- 1 thijmen thijmen   249 Jun  9  2023 runall.sh
```

Hierboven alles gecompiled met bash + alle loadtimes

Assignment 4.4: Optimize

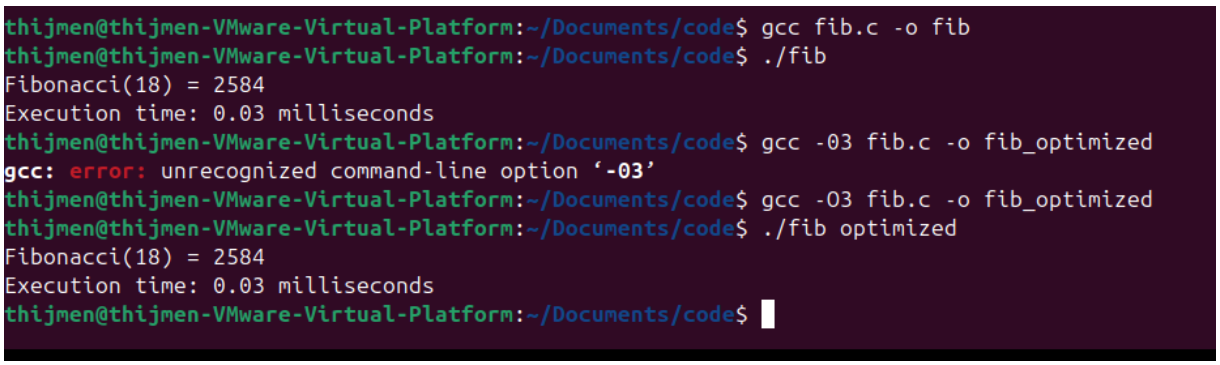
Take relevant screenshots of the following commands:

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.



Ik heb info gcc gedaan om te kijken welke paramiter ik moest hebben, dit kon ik zo snel niet vinden dus ik heb het opgezocht in dit geval is 03 het snelst.

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?

Nee, het is bij mij niet sneller

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
```

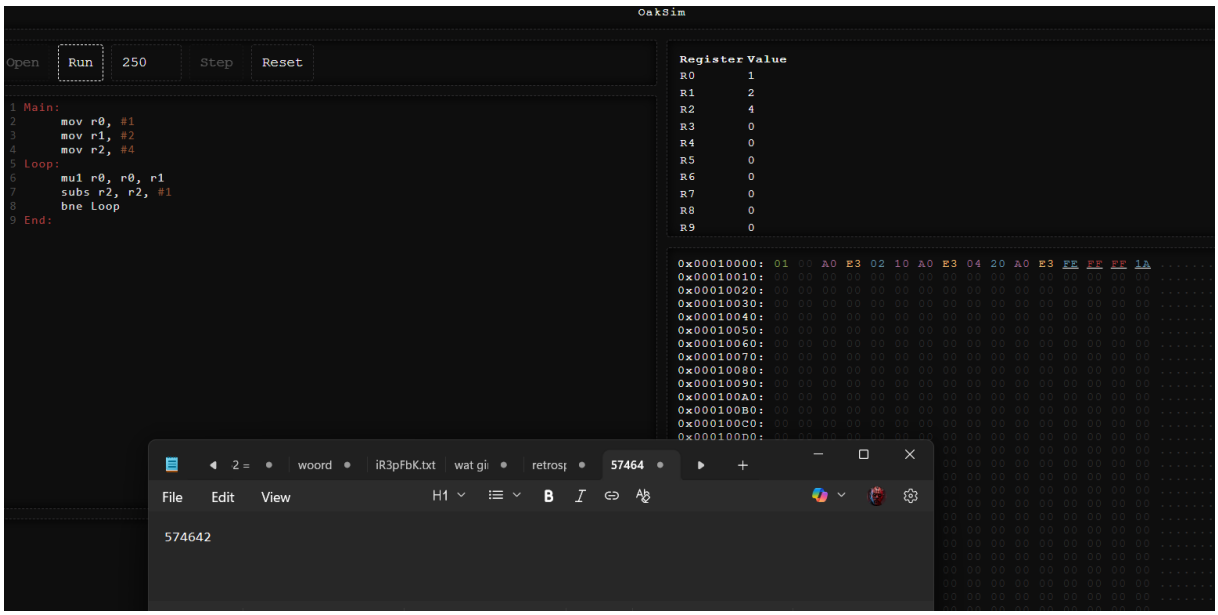
```
mov r2, #4

Loop:

End:
```

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**