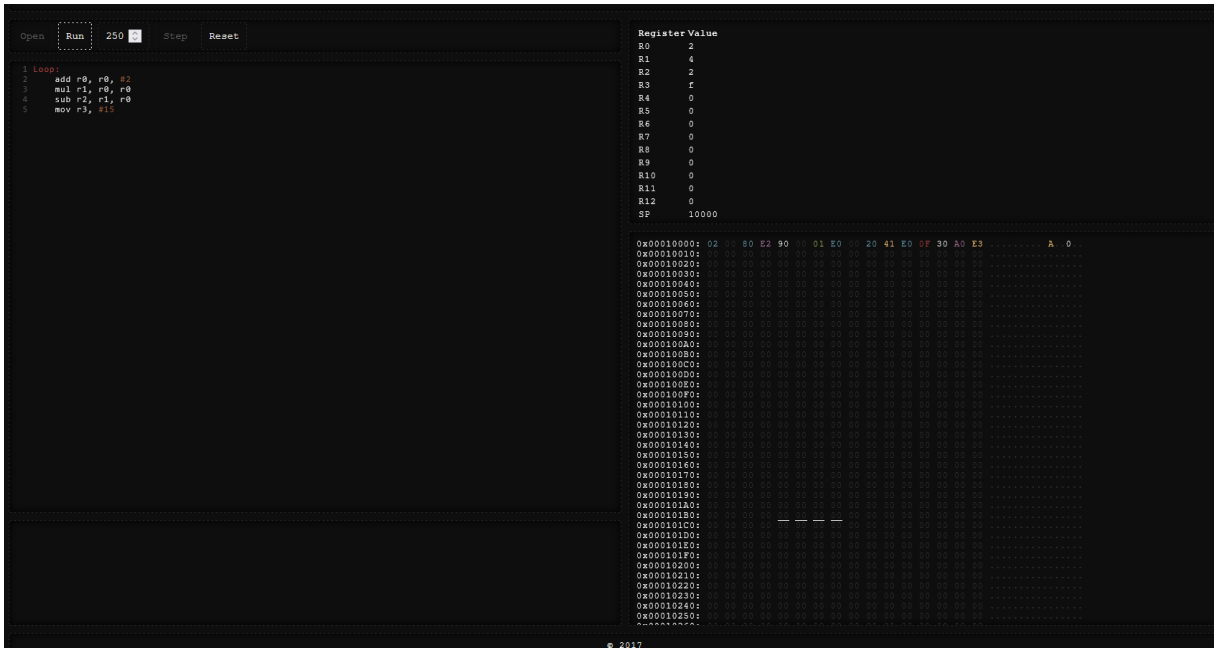
Template Week 4 – Software

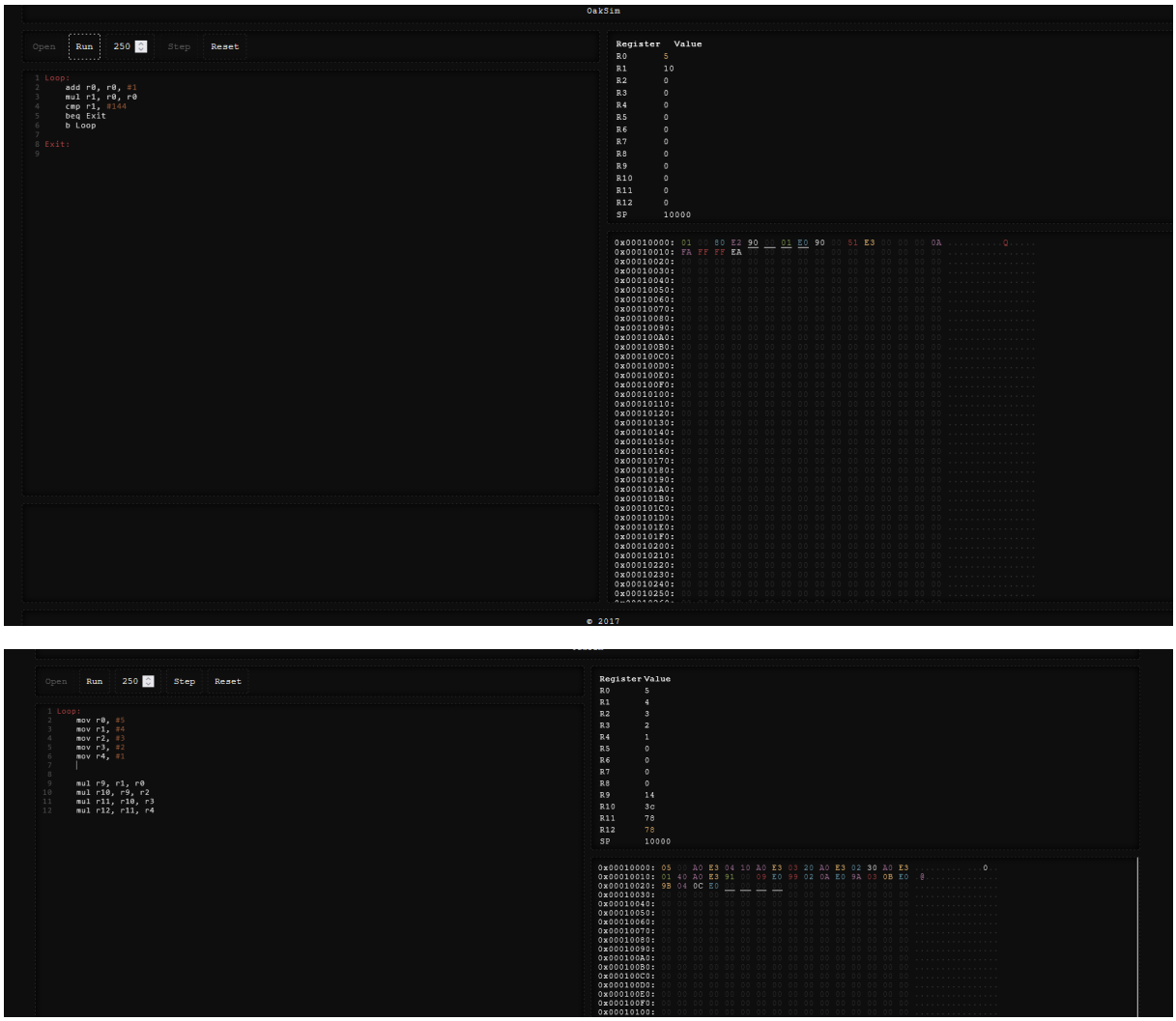
Student number: 571927

Assignment 4.1: ARM assembly

Screenshot of working assembly code of factorial calculation:



Values:
r0: 2
r1: 4
r2: 2
r3: f
r4: 0



Assignment 4.2: Programming languages

Take screenshots that the following commands work:

javac –version

```
stijn@ubuntu-591527:~$ javac --version
Command 'javac' not found, but can be installed with:
sudo apt install openjdk-17-jdk-headless  # version 17.0.17+10-1~24.04, or
sudo apt install openjdk-21-jdk-headless  # version 21.0.9+10-1~24.04
sudo apt install default-jdk              # version 2:1.17-75
sudo apt install openjdk-11-jdk-headless  # version 11.0.29+7-1ubuntu1~24.04
sudo apt install openjdk-25-jdk-headless  # version 25.0.1+8-1~24.04
sudo apt install openjdk-8-jdk-headless   # version 8u472-ga-1~24.04
sudo apt install ecj                      # version 3.32.0+eclipse4.26-2
sudo apt install openjdk-19-jdk-headless  # version 19.0.2+7-4
sudo apt install openjdk-20-jdk-headless  # version 20.0.2+9-1
sudo apt install openjdk-22-jdk-headless  # version 22~22ea-1
stijn@ubuntu-591527:~$
```

java –version

```
stijn@ubuntu-591527:~$ 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-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
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
stijn@ubuntu-591527:~$ S
```

gcc –version

```
stijn@ubuntu-591527:~$ gcc --version
gcc (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0
Copyright (C) 2023 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

stijn@ubuntu-591527:~$
```

python3 –version

```
stijn@ubuntu-591527:~$ python3 --version
Python 3.12.3
stijn@ubuntu-591527:~$
```

bash –version

```
stijn@ubuntu-591527:~$ 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.
stijn@ubuntu-591527:~$
```

Assignment 4.3: Compile

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

Fib.c, fib.py, Fibonacci.java

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?

Fibanacci.java, fib.py

Which source code files are interpreted by an interpreter?

Fib.sh, runall.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

How do I run a Java program?

Eerst compileren, dan daarna uitvoeren

How do I run a Python program?

Je gaat eerst in de terminal naar het pad van het bestand. Daarna run je ‘python (bestandsnaam)’

How do I run a C program?

Bijna dezelfde maniera ls bij Java. Eerst compileer je de code, daarna voer je de gecompileerde code uit.

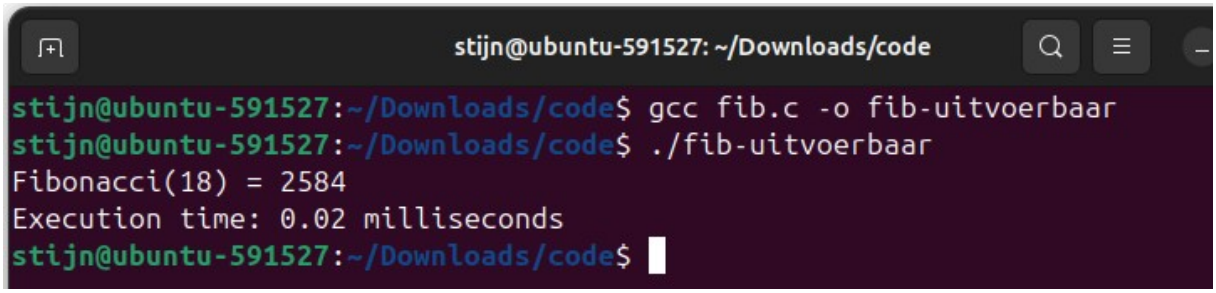
How do I run a Bash script?
Je voert hem letterlijk uit. Eventueel kan je zelfs de commando’s handmatig in CMD plakken. Ik kan er niet veel meer van maken.

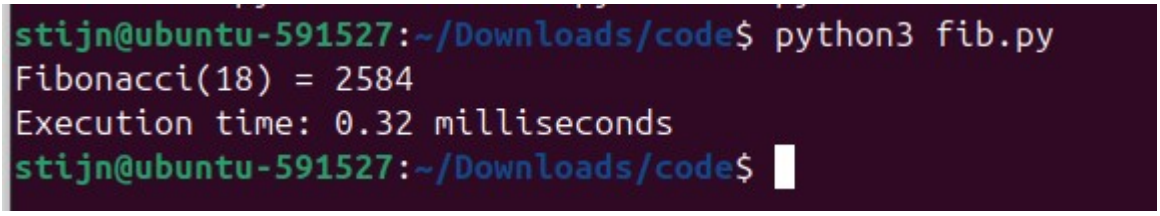
If I compile the above source code, will a new file be created? If so, which file?
Verschilt per code dat je wilt compileren.
Als je het .c bestand compileerd, krijg je waarschijnlijk een .exe bestand.

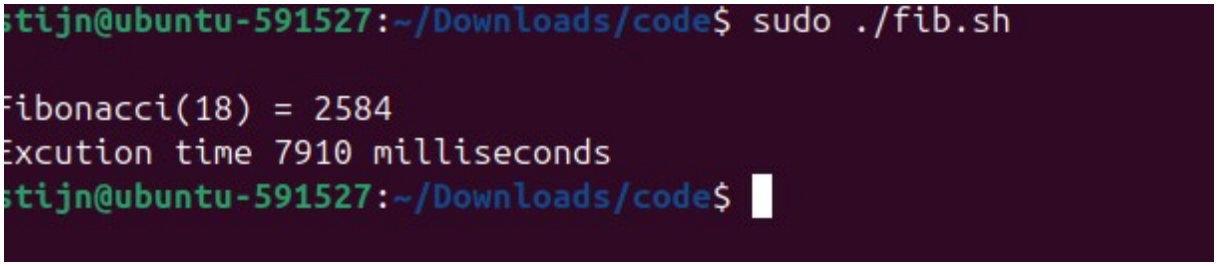
Als je het .java bestand compileerd, krijg je waarschijnlijk een (naam).class bestand. En dit is Java bytecode.

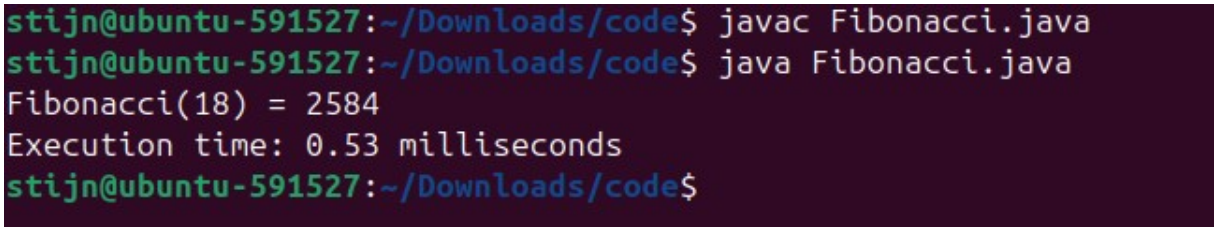
Take relevant screenshots of the following commands:

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

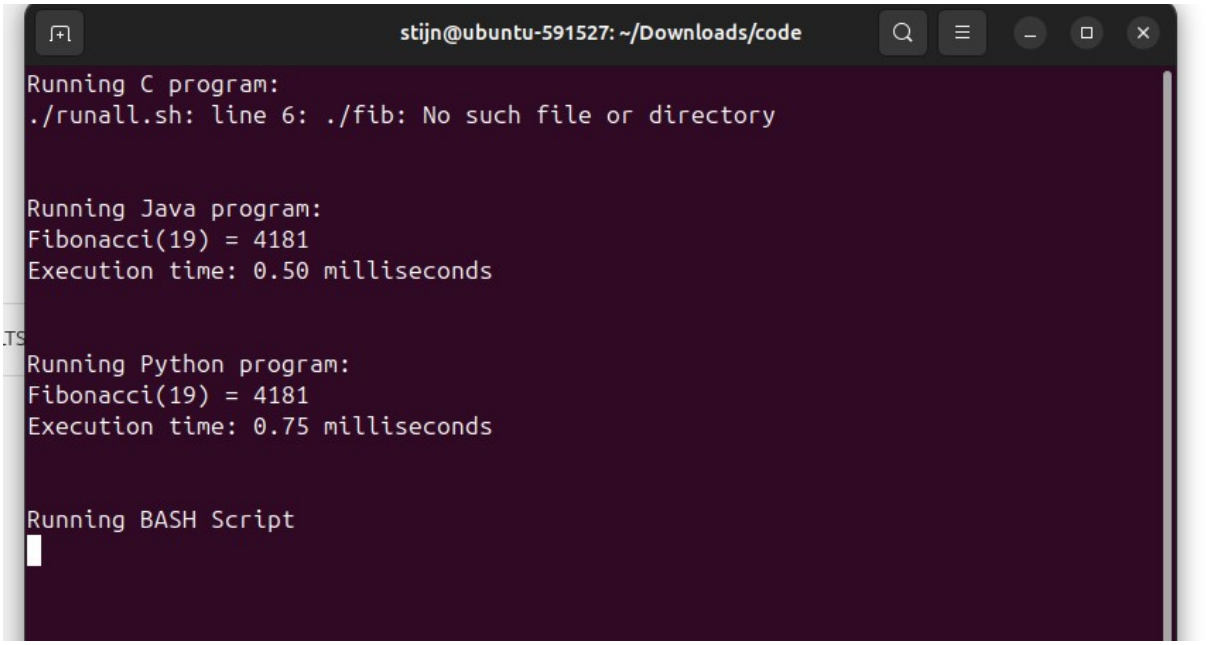








Runall.sh



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.

```
stijn@ubuntu-591527:~/Downloads/code$ ./fib-uitvoerbaar
Fibonacci(18) = 2584
Execution time: 0.03 milliseconds
stijn@ubuntu-591527:~/Downloads/code$
```

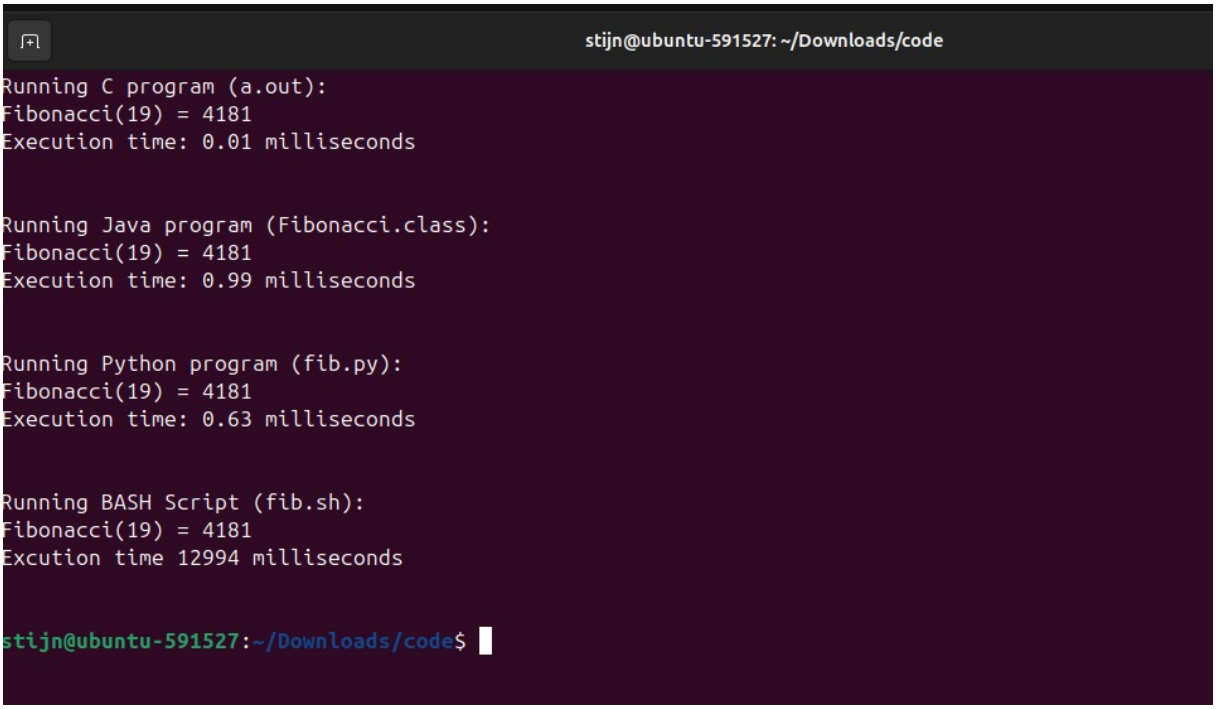
b) Compile **fib.c** again with the optimization parameters

```
stijn@ubuntu-591527:~/Downloads/code$ gcc -O3 fib.c
stijn@ubuntu-591527:~/Downloads/code$ ./a.out
Fibonacci(18) = 2584
Execution time: 0.01 milliseconds
stijn@ubuntu-591527:~/Downloads/code$
```

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

Yes. 0.02 milliseconds faster than before.

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.

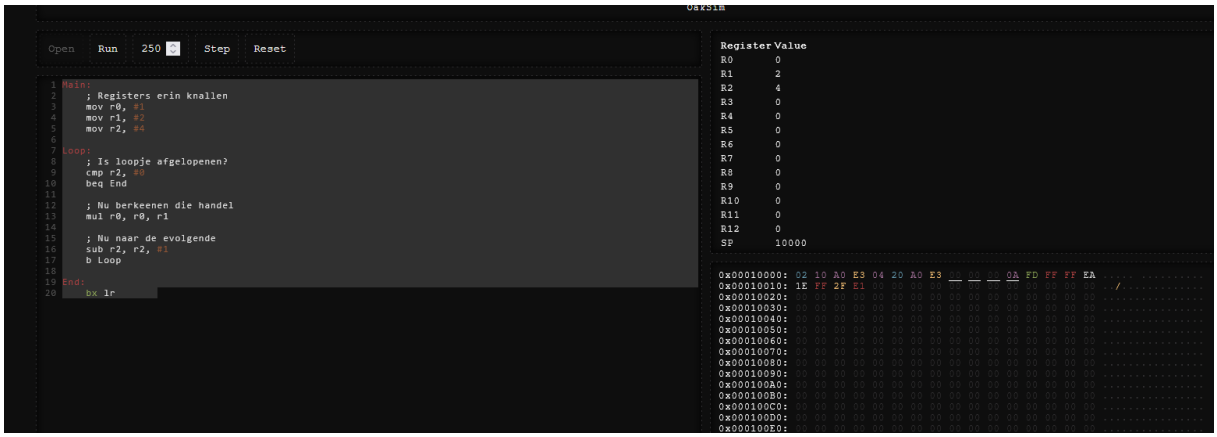
```
Main:
mov r1, #2
mov r2, #4

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**