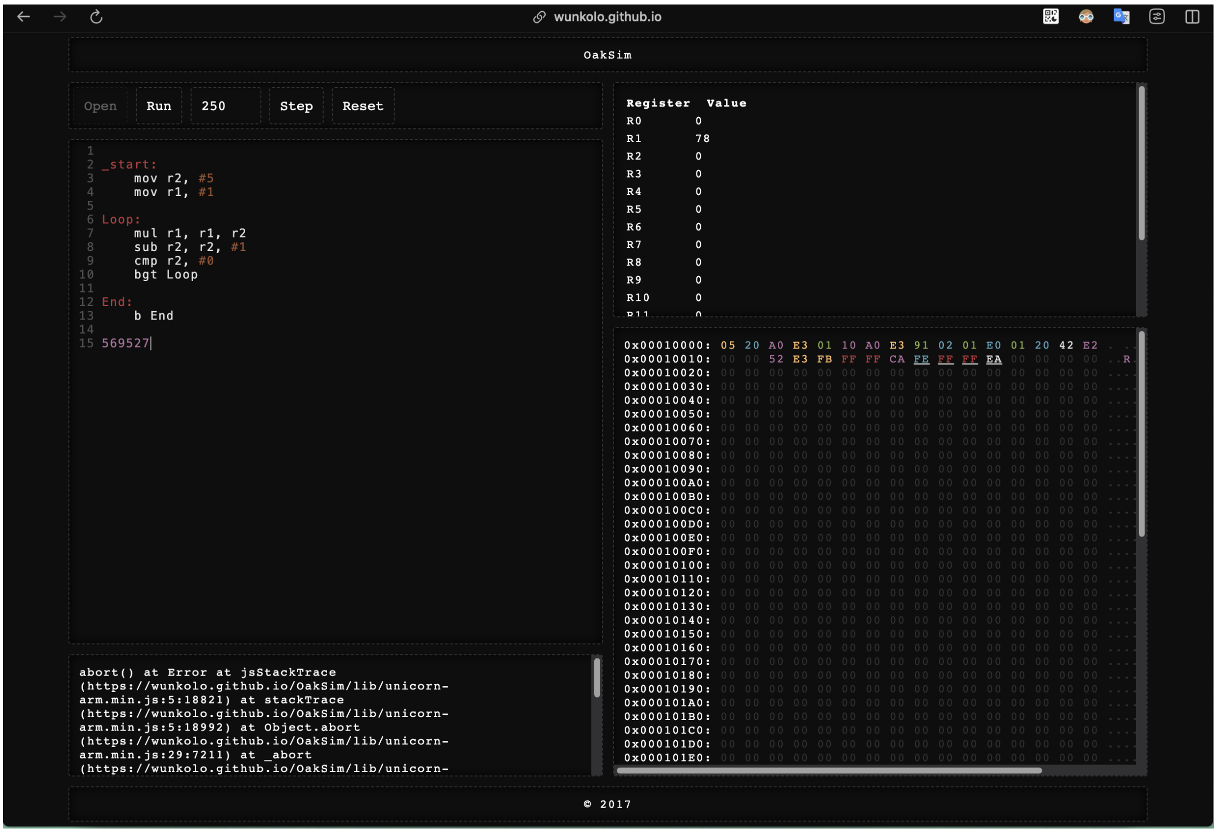
Template Week 4 – Software

Student number: 5695272

Assignment 4.1: ARM assembly

Screenshot of working assembly code of factorial calculation:



```

_start:
    mov r2, #5
    mov r1, #1

Loop:
    mul r1, r1, r2
    sub r2, r2, #1
    cmp r2, #0
    bgt Loop

End:
    b End

569527
```

```
Register  Value
R0        0
R1        78
R2        0
R3        0
R4        0
R5        0
R6        0
R7        0
R8        0
R9        0
R10       0
R11       0
```

```
0x00010000: 05 20 A0 E3 01 10 A0 E3 91 02 01 E0 01 20 42 E2  . .
0x00010010: 00 00 52 E3 FB FF FF CA FE FF FF EA 00 00 00 00  ..R
0x00010020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000100A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000100B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000100C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000100D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000100E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000100F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x00010190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000101A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000101B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000101C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000101D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
0x000101E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ....
```

```
abort() at Error at jsStackTrace
(https://wunkolo.github.io/OakSim/lib/unicorn-
arm.min.js:5:18821) at stackTrace
(https://wunkolo.github.io/OakSim/lib/unicorn-
arm.min.js:5:18992) at Object.abort
(https://wunkolo.github.io/OakSim/lib/unicorn-
arm.min.js:29:7211) at _abort
(https://wunkolo.github.io/OakSim/lib/unicorn-
```

Assignment 4.2: Programming languages

Take screenshots that the following commands work:

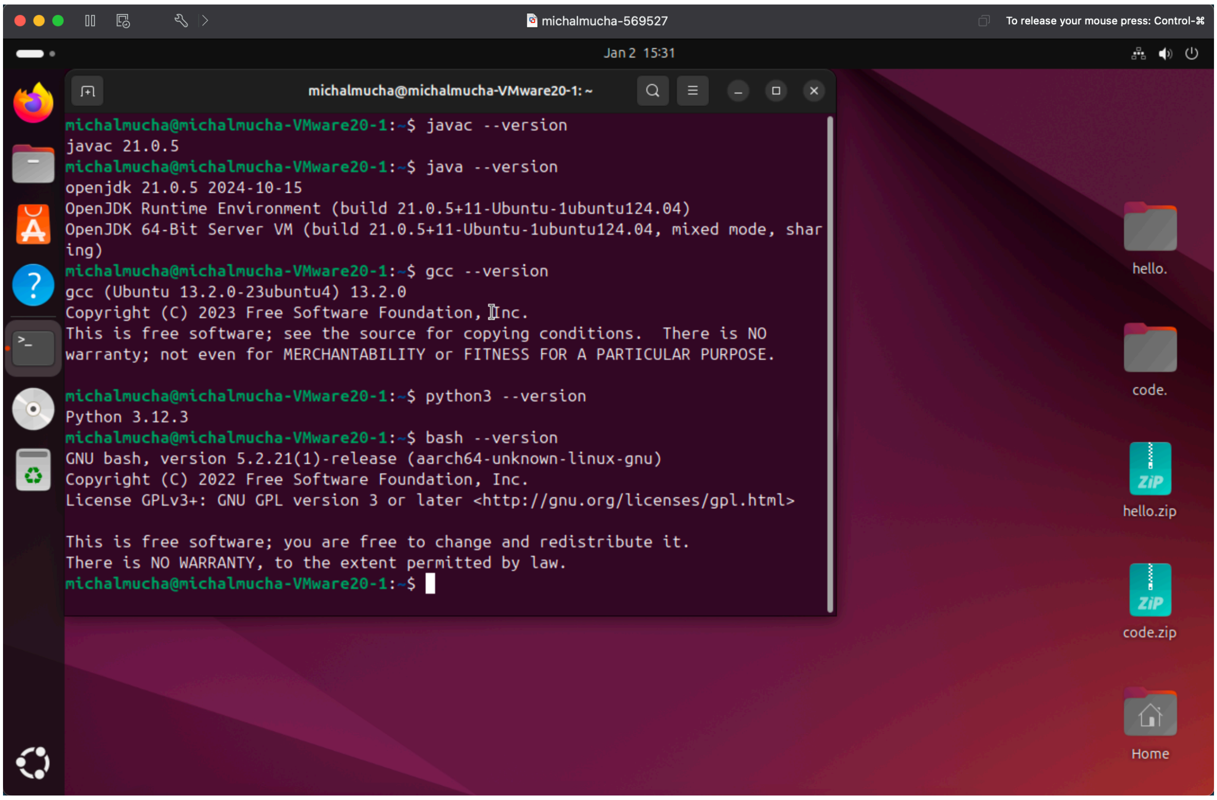
javac --version

java --version

gcc --version

python3 --version

bash –version


michalmucha-569527
To release your mouse press: Control-⌘
Jan 2 15:31
michalmucha@michalmucha-VMware20-1: ~
hello.
code.
hello.zip
code.zip
Home

```
michalmucha@michalmucha-VMware20-1:~$ javac --version
javac 21.0.5
michalmucha@michalmucha-VMware20-1:~$ java --version
openjdk 21.0.5 2024-10-15
OpenJDK Runtime Environment (build 21.0.5+11-Ubuntu-1ubuntu124.04)
OpenJDK 64-Bit Server VM (build 21.0.5+11-Ubuntu-1ubuntu124.04, mixed mode, shar
ing)
michalmucha@michalmucha-VMware20-1:~$ gcc --version
gcc (Ubuntu 13.2.0-23ubuntu4) 13.2.0
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.

michalmucha@michalmucha-VMware20-1:~$ python3 --version
Python 3.12.3
michalmucha@michalmucha-VMware20-1:~$ bash --version
GNU bash, version 5.2.21(1)-release (aarch64-unknown-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.
michalmucha@michalmucha-VMware20-1:~$
```

Assignment 4.3: Compile

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

Fibonacci.java (Java): Needs to be compiled to bytecode using javac.

fib.c (C): Needs to be compiled to machine code using gcc.

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: Interpreted by the Python interpreter.

fib.sh: Interpreted by the Bash shell.

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?

```
Java Fibonacci
```

How do I run a Python program?

```
python3 fib.py
```

How do I run a C program?

```
./fib
```

How do I run a Bash script?

```
chmod +x fib.sh
./fib.sh
```

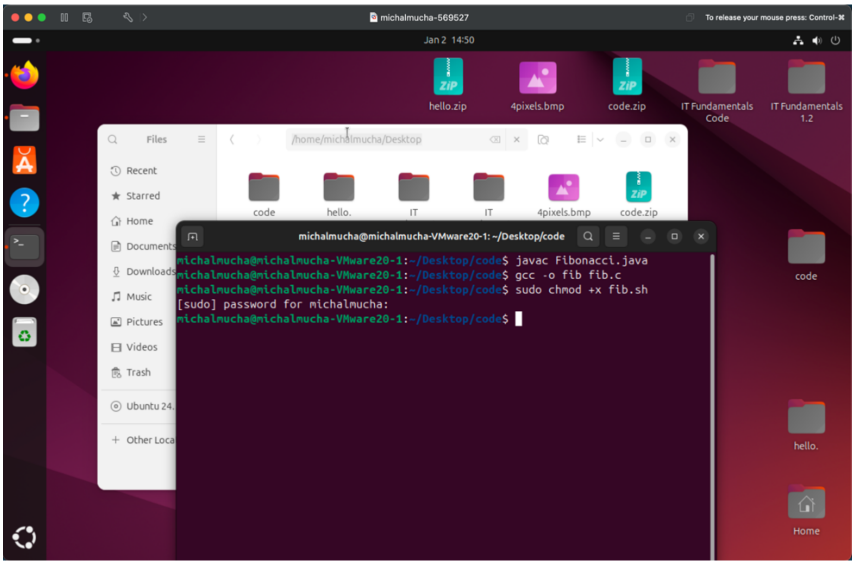
If I compile the above source code, will a new file be created? If so, which file?

Fibonacci.java: Compilation creates a bytecode file named Fibonacci.class.

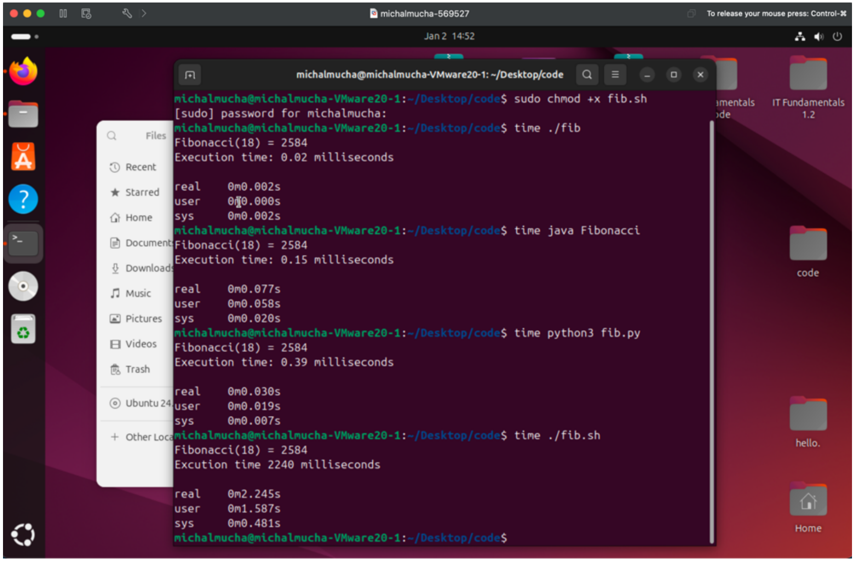
fib.c: Compilation creates a machine code executable, typically named a.out (default) or a custom name specified with -o, such as fib.

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?
  Fastest is C code



```
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ javac Fibonacci.java
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ gcc -o fib fib.c
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ sudo chmod +x fib.sh
[sudo] password for michalmucha:
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ 
```



```
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ sudo chmod +x fib.sh
[sudo] password for michalmucha:
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ time ./fib
Fibonacci(18) = 2584
Execution time: 0.02 milliseconds

real	0m0.002s
user	0m0.000s
sys	0m0.002s
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ time java Fibonacci
Fibonacci(18) = 2584
Execution time: 0.15 milliseconds

real	0m0.077s
user	0m0.058s
sys	0m0.020s
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ time python3 fib.py
Fibonacci(18) = 2584
Execution time: 0.39 milliseconds

real	0m0.030s
user	0m0.019s
sys	0m0.007s
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ time ./fib.sh
Fibonacci(18) = 2584
Excution time 2240 milliseconds

real	0m2.245s
user	0m1.587s
sys	0m0.481s
michalmucha@michalmucha-VMware20-1:~/Desktop/code$
```

Assignment 4.4: Optimize

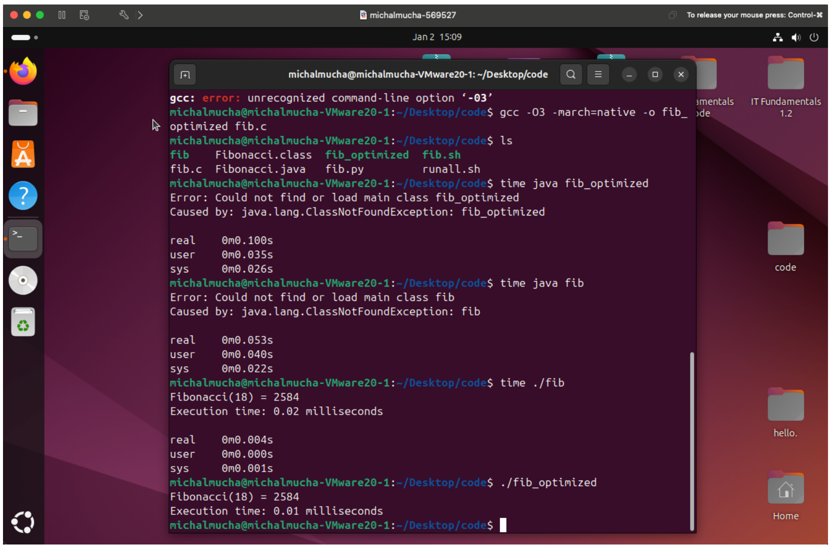
Take relevant screenshots of the following commands:

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.

gcc -O3 -march=native -o fib_optimized fib.c

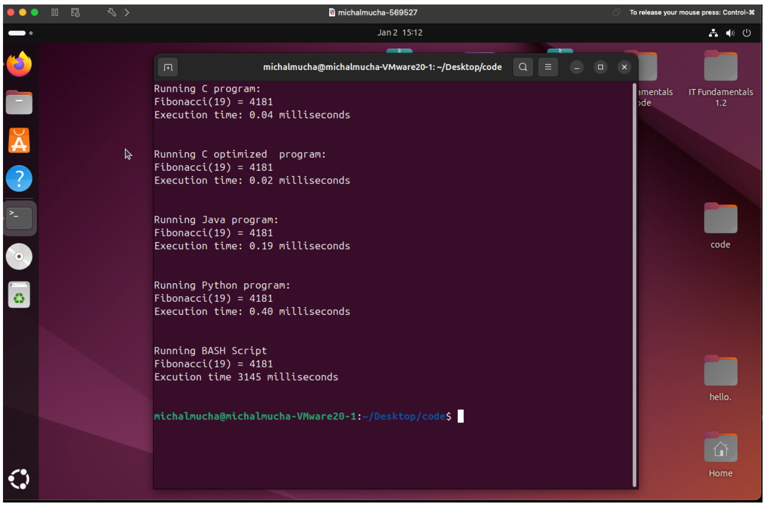
b) Compile **fib.c** again with the optimization parameters

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


michalmucha-569527
To release your mouse press: Control-⌘
Jan 2 15:09
michalmucha@michalmucha-VMware20-1: ~/Desktop/code
IT Fundamentals 1.2
code
hello.
Home

```
gcc: error: unrecognized command-line option '-03'
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ gcc -03 -march=native -o fib_
optimized fib.c
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ ls
fib    Fibonacci.class  fib_optimized  fib.sh
fib.c  Fibonacci.java   fib.py         runall.sh
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ time java fib_optimized
Error: Could not find or load main class fib_optimized
Caused by: java.lang.ClassNotFoundException: fib_optimized

real    0m0.100s
user    0m0.035s
sys     0m0.026s
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ time java fib
Error: Could not find or load main class fib
Caused by: java.lang.ClassNotFoundException: fib

real    0m0.053s
user    0m0.040s
sys     0m0.022s
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ time ./fib
Fibonacci(18) = 2584
Execution time: 0.02 milliseconds

real    0m0.004s
user    0m0.000s
sys     0m0.001s
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ ./fib_optimized
Fibonacci(18) = 2584
Execution time: 0.01 milliseconds
michalmucha@michalmucha-VMware20-1:~/Desktop/code$ 
```

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.


michalmucha-569527
To release your mouse press: Control-⌘
Jan 2 15:12
michalmucha@michalmucha-VMware20-1: ~/Desktop/code
IT Fundamentals 1.2
code
hello.
Home

```
Running C program:
Fibonacci(19) = 4181
Execution time: 0.04 milliseconds


Running C optimized  program:
Fibonacci(19) = 4181
Execution time: 0.02 milliseconds


Running Java program:
Fibonacci(19) = 4181
Execution time: 0.19 milliseconds


Running Python program:
Fibonacci(19) = 4181
Execution time: 0.40 milliseconds


Running BASH Script
Fibonacci(19) = 4181
Excution time 3145 milliseconds


michalmucha@michalmucha-VMware20-1:~/Desktop/code$ 
```

a

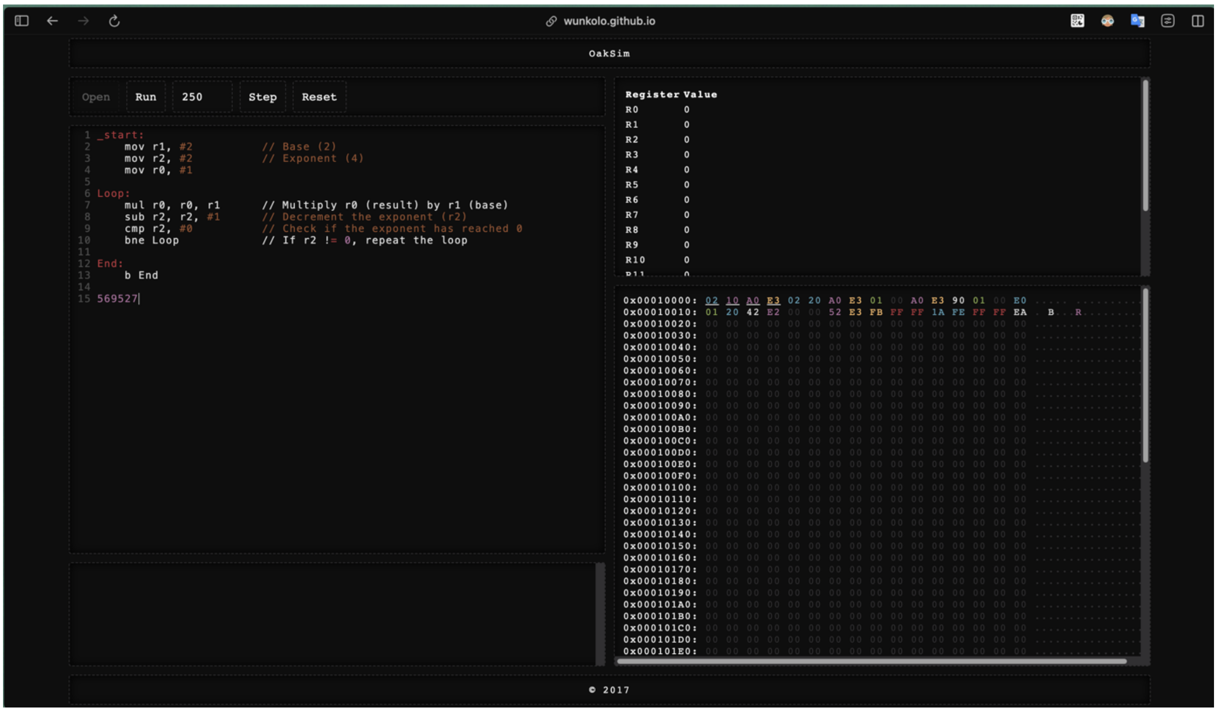
Bonus point assignment – week 4

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.



```
_start:
    mov r1, #2          // Base (2)
    mov r2, #2          // Exponent (4)
    mov r0, #1

Loop:
    mul r0, r0, r1      // Multiply r0 (result) by r1 (base)
    sub r2, r2, #1      // Decrement the exponent (r2)
    cmp r2, #0          // Check if the exponent has reached 0
    bne Loop            // If r2 != 0, repeat the loop

End:
    b End

569527
```

```
Register Value
R0       0
R1       0
R2       0
R3       0
R4       0
R5       0
R6       0
R7       0
R8       0
R9       0
R10      0
R11      0
```

```
0x00010000: 02 10 A0 E3 02 20 A0 E3 01 00 A0 E3 90 01 00 E0 ................
0x00010010: 01 20 42 E2 00 00 52 E3 FB FF FF 1A FE FF FF EA . B..R..........
0x00010020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000100A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000100B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000100C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000100D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000100E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000100F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000101A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000101B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000101C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000101D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x000101E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

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