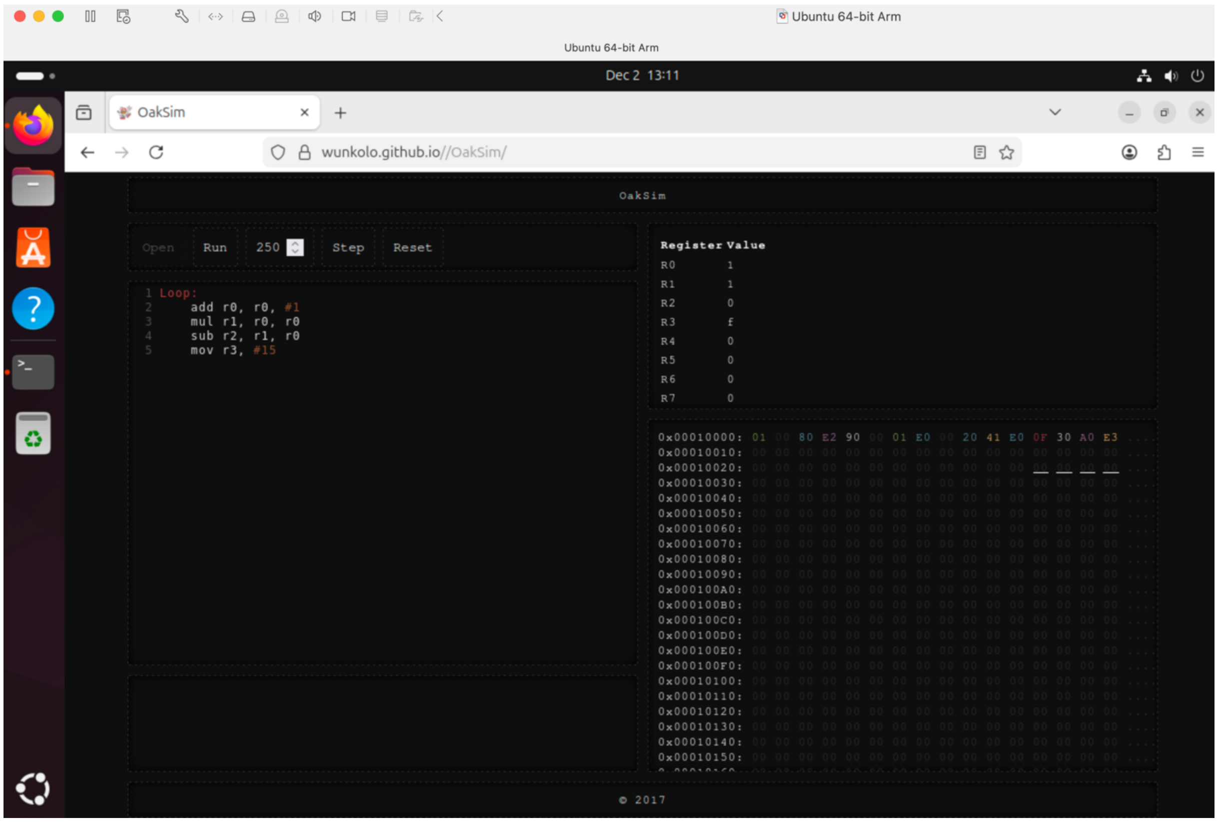
Template Week 4 – Software

Student number: 586461

Assignment 4.1: ARM assembly

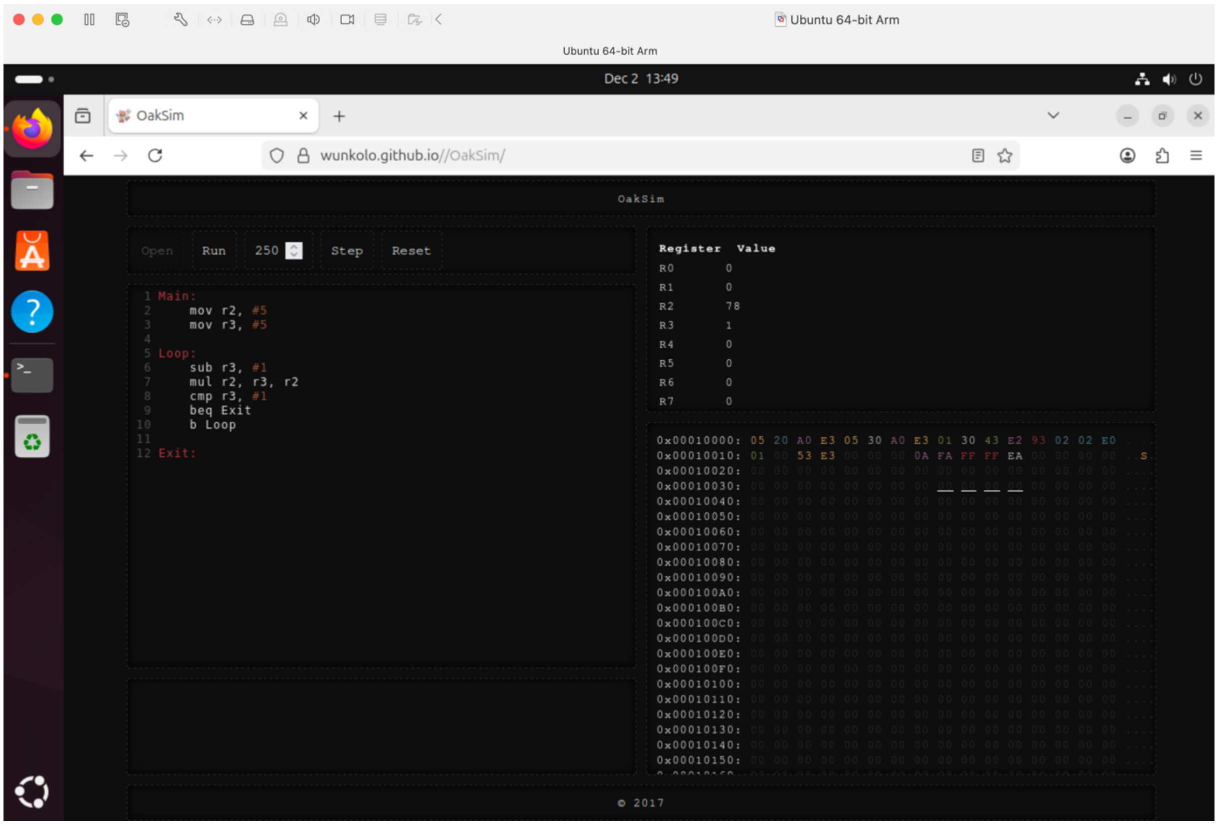
Screenshot of working assembly code of factorial calculation:



```
Loop:
    add r0, r0, #1
    mul r1, r0, r0
    sub r2, r1, r0
    mov r3, #15
```

```
Register Value
R0       1
R1       1
R2       0
R3       f
R4       0
R5       0
R6       0
R7       0
```

```
0x00010000: 01 00 80 E2 90 00 01 E0 00 20 41 E0 0F 30 A0 E3 ....
0x00010010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
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 ....
```



```
Main:
    mov r2, #5
    mov r3, #5

Loop:
    sub r3, #1
    mul r2, r3, r2
    cmp r3, #1
    beq Exit
    b Loop

Exit:
```

```
Register  Value
R0        0
R1        0
R2        78
R3        1
R4        0
R5        0
R6        0
R7        0
```

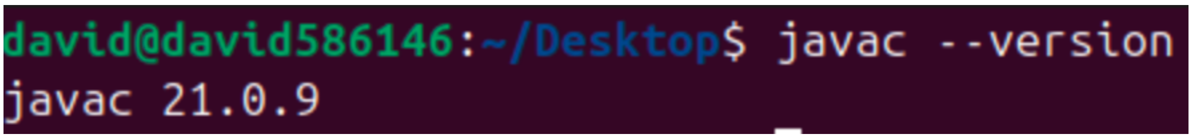
```
0x00010000: 05 20 A0 E3 05 30 A0 E3 01 30 43 E2 93 02 02 E0 ....
0x00010010: 01 00 53 E3 00 00 00 0A FA FF FF EA 00 00 00 00 ..S.
0x00010020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
0x00010030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
0x00010040: 00 00 00 00 00 00 00 00 00 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 ....
```

© 2017

Assignment 4.2: Programming languages

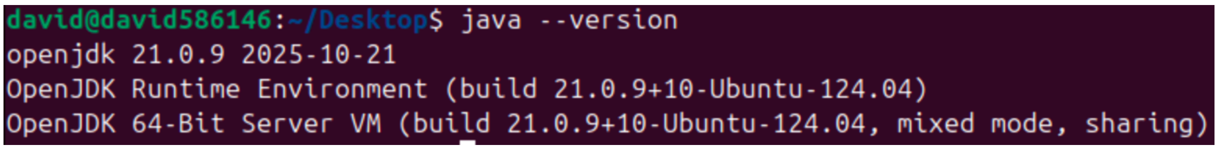
Take screenshots that the following commands work:

javac –version



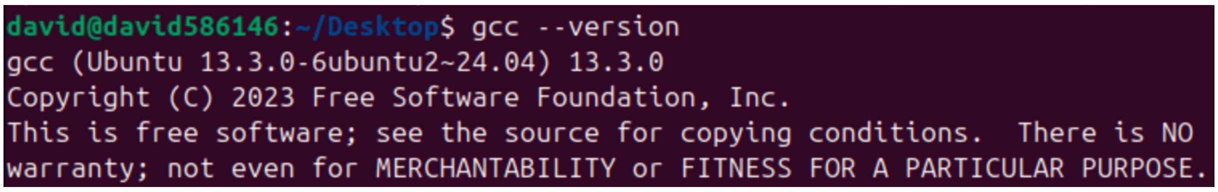
```
david@david586146:~/Desktop$ javac --version
javac 21.0.9
```

java –version



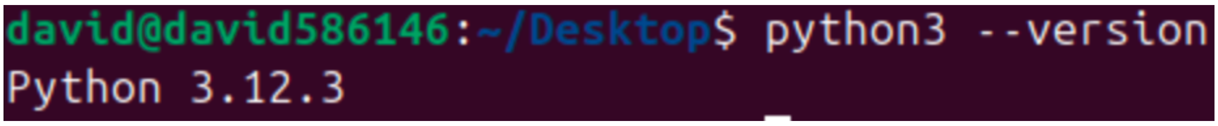
```
david@david586146:~/Desktop$ java --version
openjdk 21.0.9 2025-10-21
OpenJDK Runtime Environment (build 21.0.9+10-Ubuntu-124.04)
OpenJDK 64-Bit Server VM (build 21.0.9+10-Ubuntu-124.04, mixed mode, sharing)
```

gcc –version



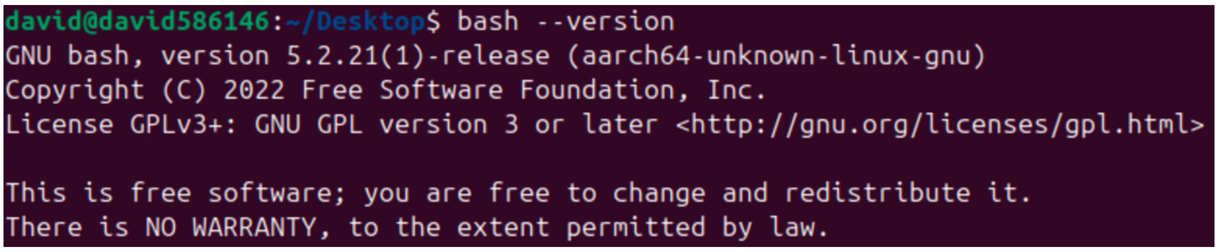
```
david@david586146:~/Desktop$ gcc --version
gcc (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0
Copyright (C) 2023 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

python3 –version



```
david@david586146:~/Desktop$ python3 --version
Python 3.12.3
```

bash --version



```
david@david586146:~/Desktop$ 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.
```

Assignment 4.3: Compile

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

- Fibonacci.java and 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

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 file_name

How do I run a Python program?

- python3 file_name.py

How do I run a C program?

- ./file_name

How do I run a Bash script?

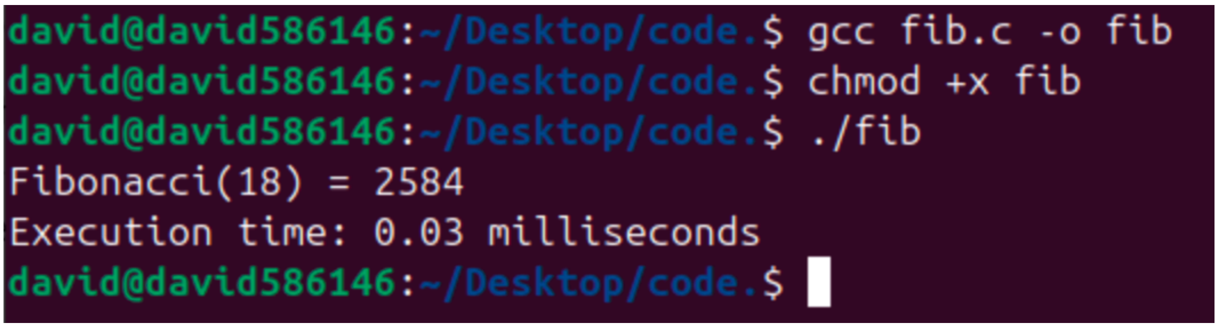
- ./file_name

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

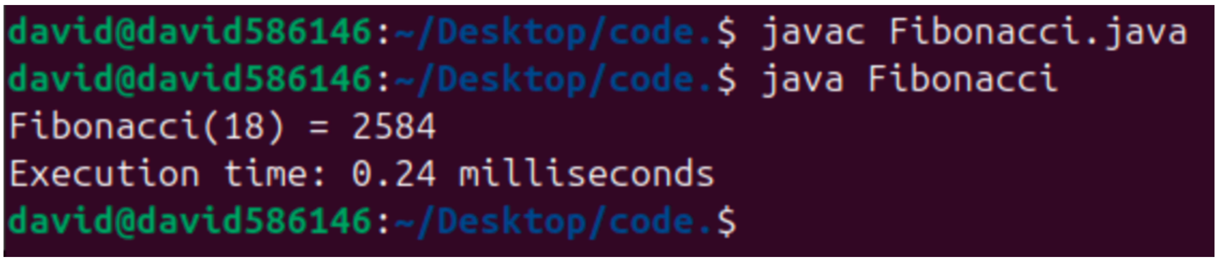
Java will create a new file called file_name.class and C will compile to a runnable file.

Take relevant screenshots of the following commands:

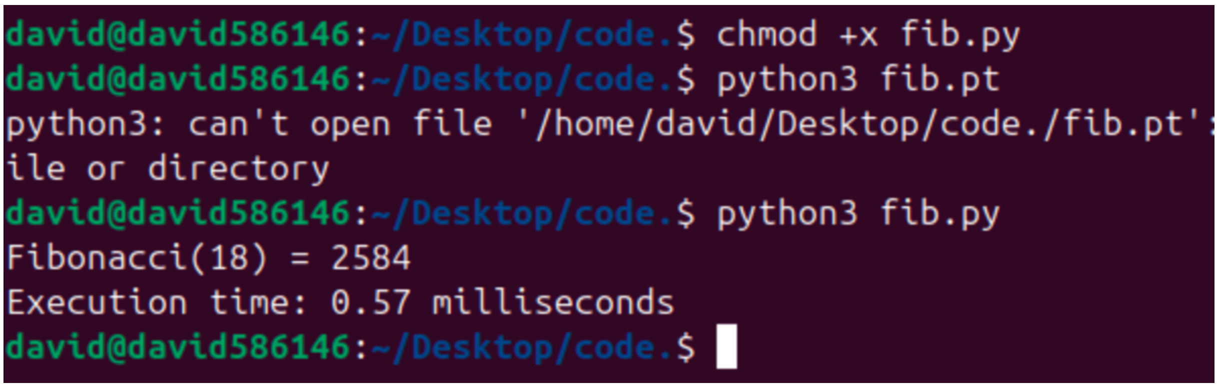
- Compile the source files where necessary
- Make them executable
- Run them
- Which (compiled) source code file performs the calculation the fastest?



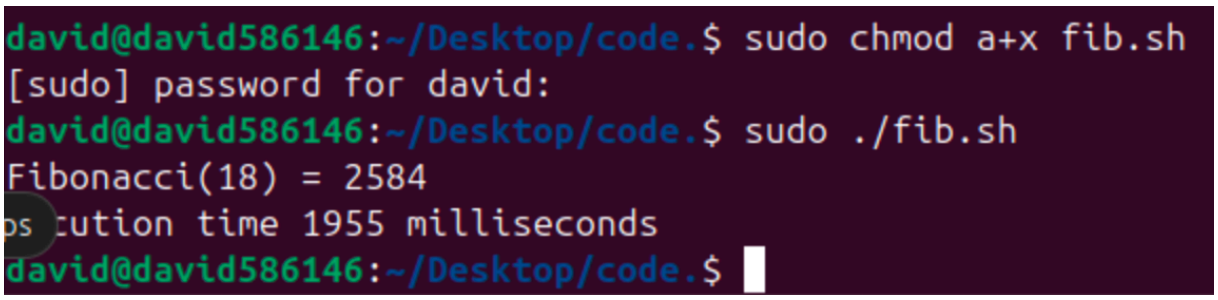
```
david@david586146:~/Desktop/code.$ gcc fib.c -o fib
david@david586146:~/Desktop/code.$ chmod +x fib
david@david586146:~/Desktop/code.$ ./fib
Fibonacci(18) = 2584
Execution time: 0.03 milliseconds
david@david586146:~/Desktop/code.$
```



```
david@david586146:~/Desktop/code.$ javac Fibonacci.java
david@david586146:~/Desktop/code.$ java Fibonacci
Fibonacci(18) = 2584
Execution time: 0.24 milliseconds
david@david586146:~/Desktop/code.$
```



```
david@david586146:~/Desktop/code.$ chmod +x fib.py
david@david586146:~/Desktop/code.$ python3 fib.pt
python3: can't open file '/home/david/Desktop/code./fib.pt'
ile or directory
david@david586146:~/Desktop/code.$ python3 fib.py
Fibonacci(18) = 2584
Execution time: 0.57 milliseconds
david@david586146:~/Desktop/code.$
```



```
david@david586146:~/Desktop/code.$ sudo chmod a+x fib.sh
[sudo] password for david:
david@david586146:~/Desktop/code.$ sudo ./fib.sh
Fibonacci(18) = 2584
ps cution time 1955 milliseconds
david@david586146:~/Desktop/code.$
```

C is the fastest one of all of them.

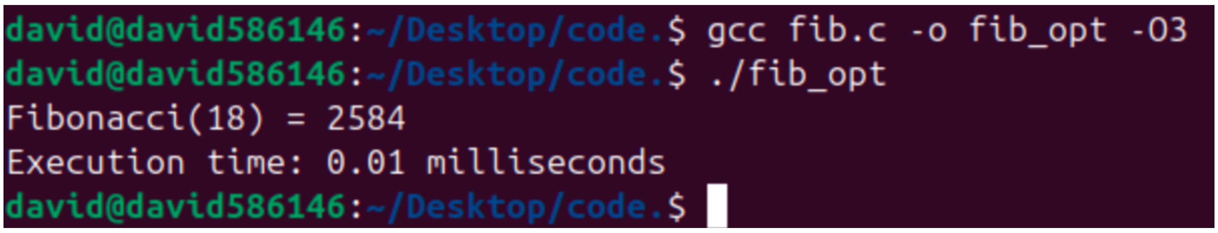
Assignment 4.4: Optimize

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 have chosen to use O3 for optimizing the script.

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

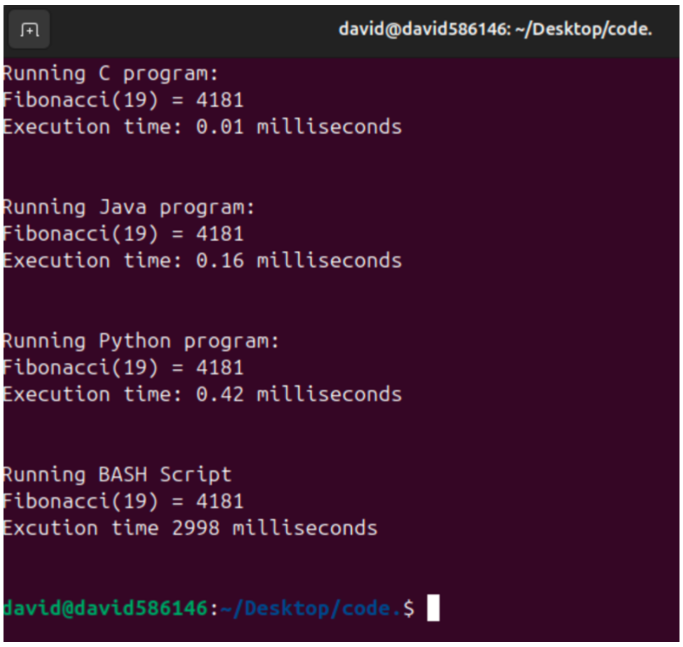


```
david@david586146:~/Desktop/code.$ gcc fib.c -o fib_opt -O3
david@david586146:~/Desktop/code.$ ./fib_opt
Fibonacci(18) = 2584
Execution time: 0.01 milliseconds
david@david586146:~/Desktop/code.$ 
```

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

Yes it’s by a mile faster, it now executes under 0.01 milliseconds.

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.



```
david@david586146: ~/Desktop/code.

Running C program:
Fibonacci(19) = 4181
Execution time: 0.01 milliseconds


Running Java program:
Fibonacci(19) = 4181
Execution time: 0.16 milliseconds


Running Python program:
Fibonacci(19) = 4181
Execution time: 0.42 milliseconds


Running BASH Script
Fibonacci(19) = 4181
Excution time 2998 milliseconds


david@david586146:~/Desktop/code.$ 
```

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.

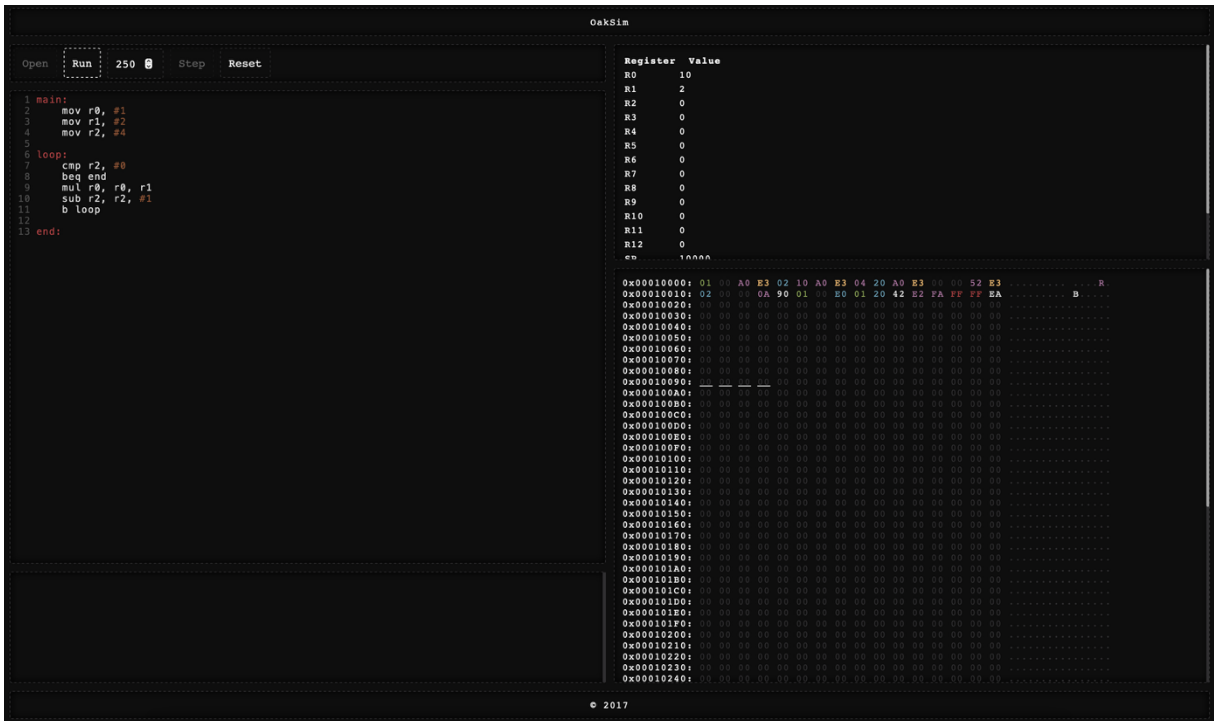
```
Main:
mov r0, #1
mov r1, #2
mov r2, #4

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.



```
main:
    mov r0, #1
    mov r1, #2
    mov r2, #4

loop:
    cmp r2, #0
    beq end
    mul r0, r0, r1
    sub r2, r2, #1
    b loop

end:
```

```
Register  Value
R0        10
R1        2
R2        0
R3        0
R4        0
R5        0
R6        0
R7        0
R8        0
R9        0
R10       0
R11       0
R12       0
SP        10000
```

```
0x00010000: 01 00 A0 E3 02 10 A0 E3 04 20 A0 E3 00 00 52 E3 ..............R.
0x00010010: 02 00 00 0A 90 01 00 E0 01 20 42 E2 FA FF FF EA ...........B....
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 ................
0x000101F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0x00010240: 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**