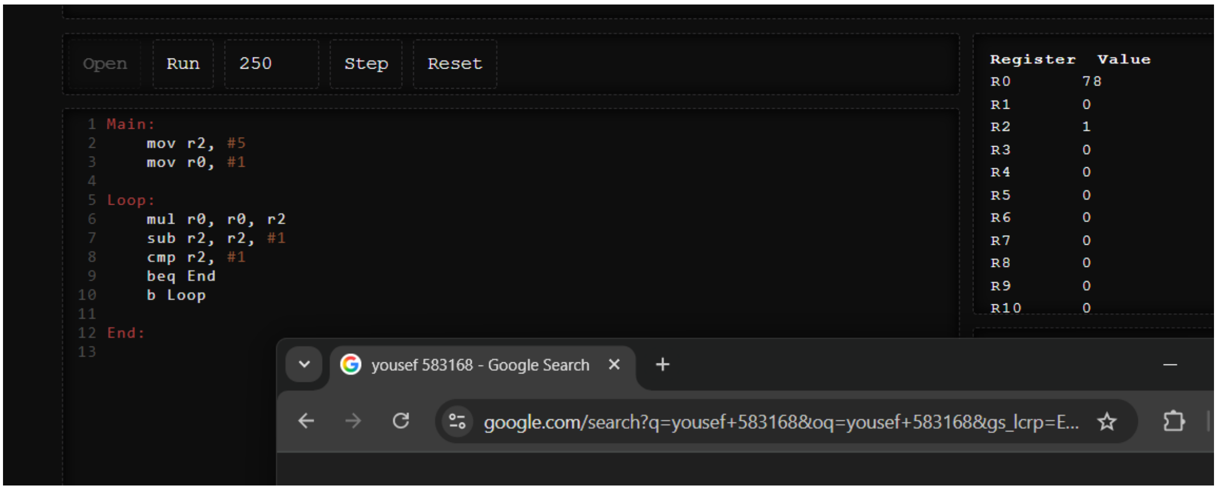
# Template Week 4 – Software

Student number:583168

### Assignment 4.1: ARM assembly

Screenshot of working assembly code of factorial calculation:


Open
Run
250
Step
Reset
yousef 583168 - Google Search
google.com/search?q=yousef+583168&oq=yousef+583168&gs_lcrp=E...

```
Main:
    mov r2, #5
    mov r0, #1

Loop:
    mul r0, r0, r2
    sub r2, r2, #1
    cmp r2, #1
    beq End
    b Loop

End:
```

| Register | Value |
|---|---|
| R0 | 78 |
| R1 | 0 |
| R2 | 1 |
| R3 | 0 |
| R4 | 0 |
| R5 | 0 |
| R6 | 0 |
| R7 | 0 |
| R8 | 0 |
| R9 | 0 |
| R10 | 0 |

### Assignment 4.2: Programming languages

Take screenshots that the following commands work:

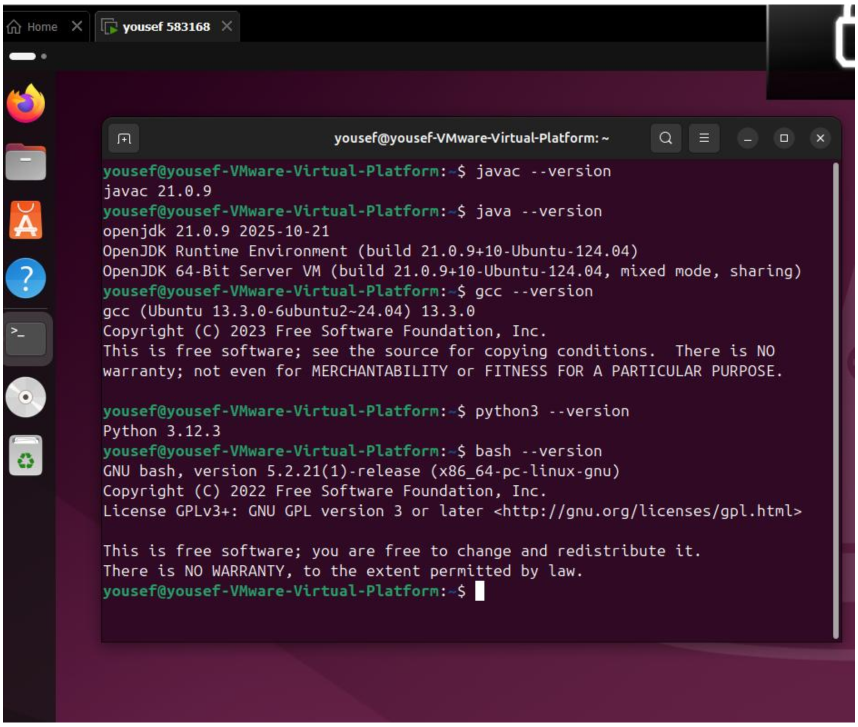
javac --version

java --version

gcc --version

python3 --version

bash –version



```
yousef@yousef-VMware-Virtual-Platform:~$ javac --version
javac 21.0.9
yousef@yousef-VMware-Virtual-Platform:~$ 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)
yousef@yousef-VMware-Virtual-Platform:~$ 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.

yousef@yousef-VMware-Virtual-Platform:~$ python3 --version
Python 3.12.3
yousef@yousef-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.
yousef@yousef-VMware-Virtual-Platform:~$
```

**Assignment 4.3: Compile**

**Which of the above files need to be compiled before you can run them?**
Fibonacci.java and fib.c need to be compiled before running, fib.py and fib.sh are interpreted.

**Which source code files are compiled into machine code and are then directly executable by a processor?**
fib.c is compiled into machine code and directly executable by the processor.

**Which source code files are compiled to byte code?**
Fibonacci.java is compiled to byte code.

**Which source code files are interpreted by an interpreter?**
fib.py is interpreted by the Python interpreter and fib.sh is interpreted by the Bash shell.

**These source code files perform the same calculation. Which one is expected to perform the calculation the fastest?**
fib.c is expected to perform the calculation the fastest.

**How do I run a Java program?**
Compile Java with javac Fibonacci.java and run with java Fibonacci.

**How do I run a Python program?**
Run Python with python3 fib.py.

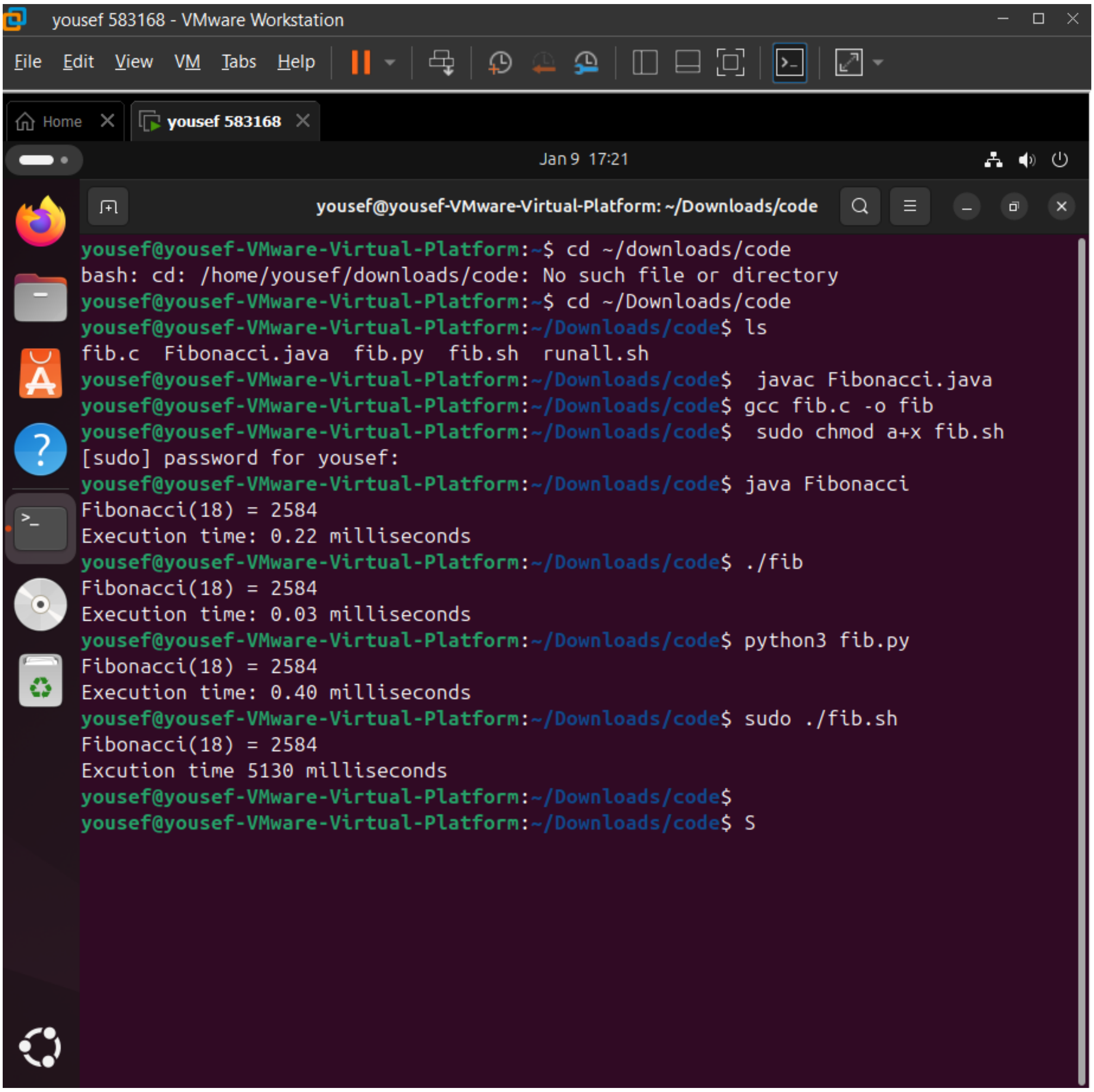
**How do I run a C program?**
Compile C with gcc fib.c -o fib and run with ./fib.

**How do I run a Bash Script?**
Make Bash script executable with sudo chmod a+x fib.sh and run with sudo ./fib.sh.

**If I compile the above source code, will a new file be created? If so, which file?**
Compiling creates Fibonacci.class for Java and fib for C.

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?


yousef 583168 - VMware Workstation
File Edit View VM Tabs Help
Home
yousef 583168
Jan 9 17:21
yousef@yousef-VMware-Virtual-Platform: ~/Downloads/code

```
yousef@yousef-VMware-Virtual-Platform:~$ cd ~/downloads/code
bash: cd: /home/yousef/downloads/code: No such file or directory
yousef@yousef-VMware-Virtual-Platform:~$ cd ~/Downloads/code
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ ls
fib.c  Fibonacci.java  fib.py  fib.sh  runall.sh
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$  javac Fibonacci.java
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ gcc fib.c -o fib
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$  sudo chmod a+x fib.sh
[sudo] password for yousef:
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ java Fibonacci
Fibonacci(18) = 2584
Execution time: 0.22 milliseconds
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ ./fib
Fibonacci(18) = 2584
Execution time: 0.03 milliseconds
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ python3 fib.py
Fibonacci(18) = 2584
Execution time: 0.40 milliseconds
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ sudo ./fib.sh
Fibonacci(18) = 2584
Excution time 5130 milliseconds
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ S
```

The C program fib.c runs the fastest

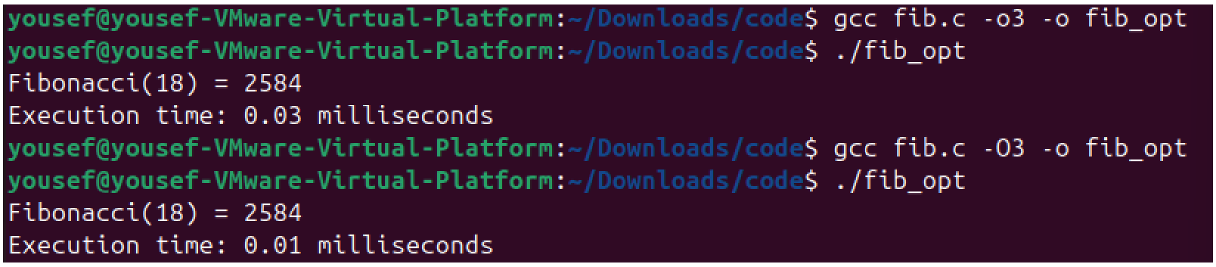
**Assignment 4.4: Optimize**

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.

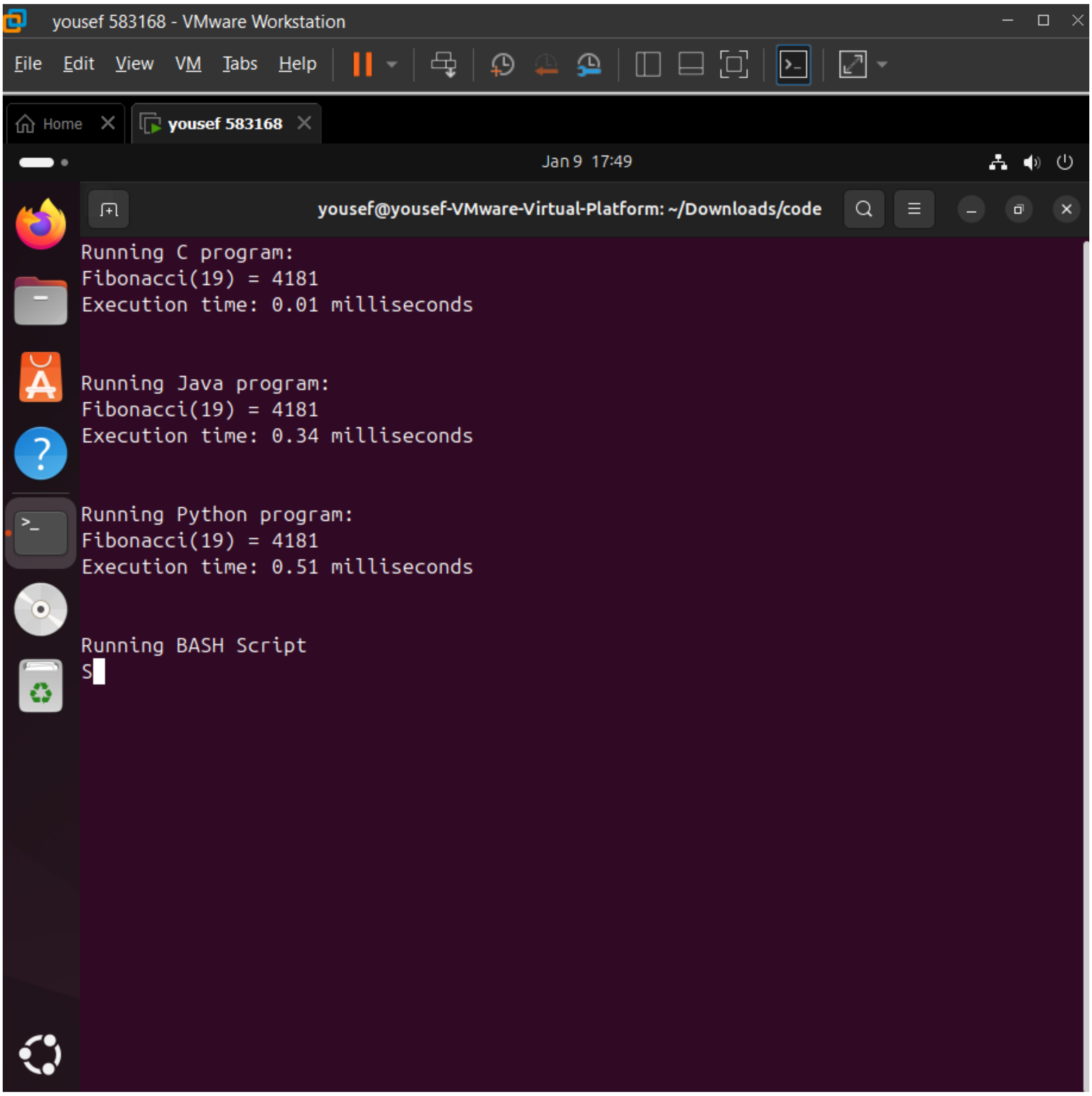
b) Compile **fib.c** again with the optimization parameters

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



```
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ gcc fib.c -o3 -o fib_opt
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ ./fib_opt
Fibonacci(18) = 2584
Execution time: 0.03 milliseconds
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ gcc fib.c -O3 -o fib_opt
yousef@yousef-VMware-Virtual-Platform:~/Downloads/code$ ./fib_opt
Fibonacci(18) = 2584
Execution time: 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.


yousef 583168 - VMware Workstation
File Edit View VM Tabs Help
Home
yousef 583168
Jan 9 17:49
yousef@yousef-VMware-Virtual-Platform: ~/Downloads/code

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


Running Java program:
Fibonacci(19) = 4181
Execution time: 0.34 milliseconds


Running Python program:
Fibonacci(19) = 4181
Execution time: 0.51 milliseconds


Running BASH Script
S
```

**Assignment 4.5: More ARM Assembly**

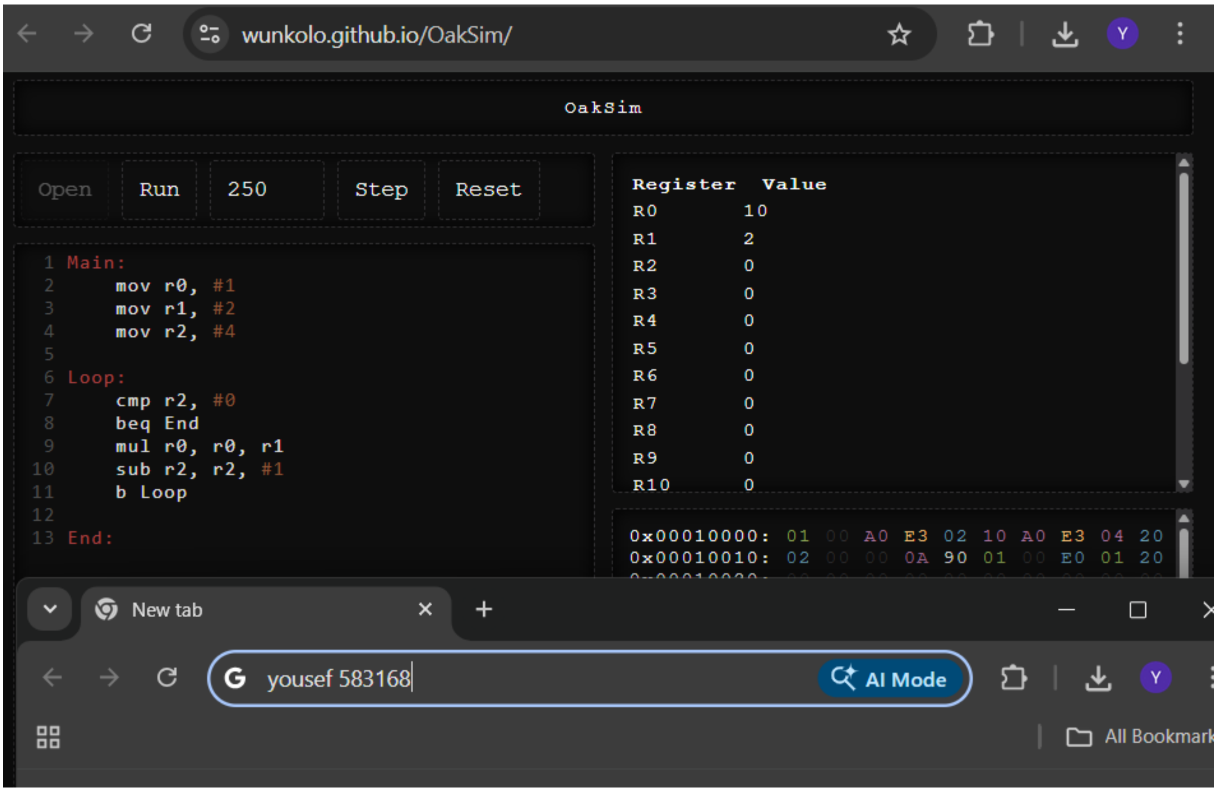
Like the factorial example, you can also implement the calculation of a power of 2 in assembly. For example you want to calculate $2^4$ = 16. Use iteration to calculate the result. Store the result in r0.

```
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.



```
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
```

```
0x00010000: 01 00 A0 E3 02 10 A0 E3 04 20
0x00010010: 02 00 00 0A 90 01 00 E0 01 20
```

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