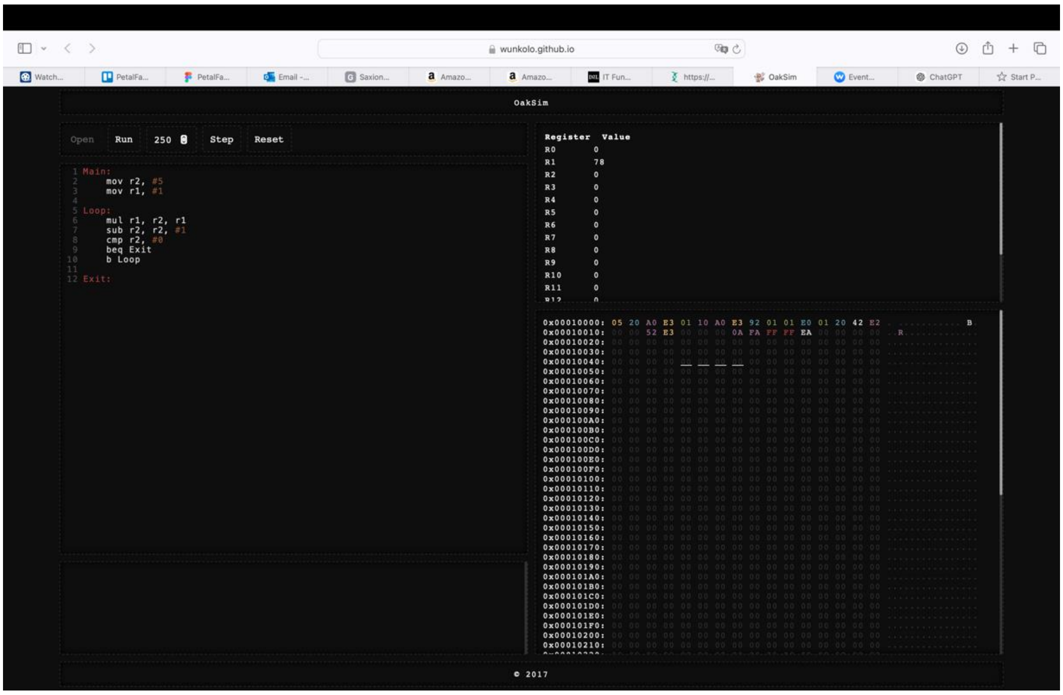
Template Week 4 – Software

Student number: 561004

Assignment 4.1: ARM assembly

Screenshot of working assembly code of factorial calculation:



```
Main:
    mov r2, #5
    mov r1, #1

Loop:
    mul r1, r2, r1
    sub r2, r2, #1
    cmp r2, #0
    beq Exit
    b Loop

Exit:
```

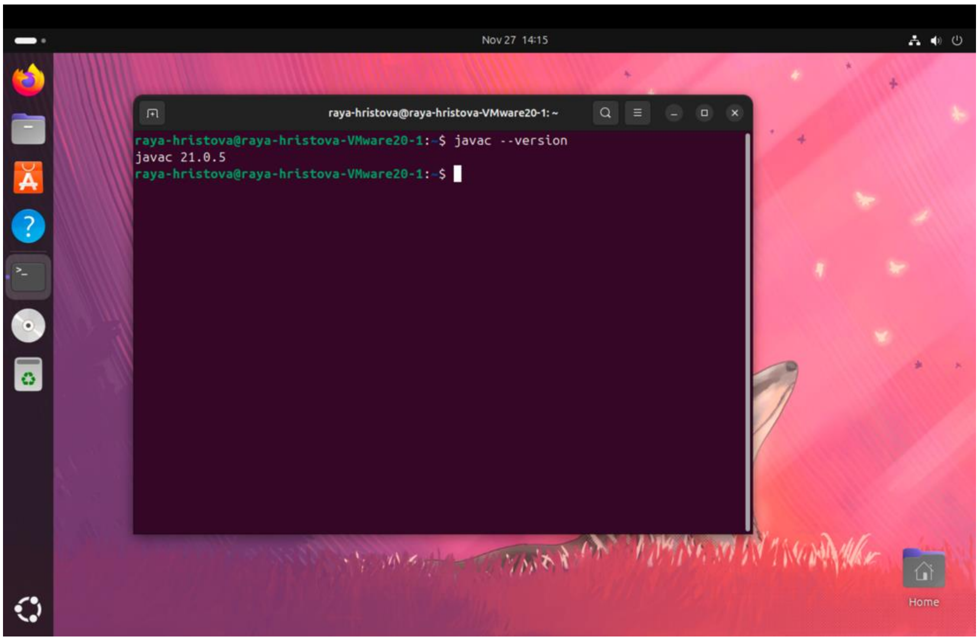
```
Register  Value
R0        0
R1        78
R2        0
R3        0
R4        0
R5        0
R6        0
R7        0
R8        0
R9        0
R10       0
R11       0
R12       0
```

```
0x00010000: 05 20 A0 E3 01 10 A0 E3 92 01 01 E0 01 20 42 E2 .............B.
0x00010010: 00 00 52 E3 00 00 00 0A FA 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 ................
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 ................
```

Assignment 4.2: Programming languages

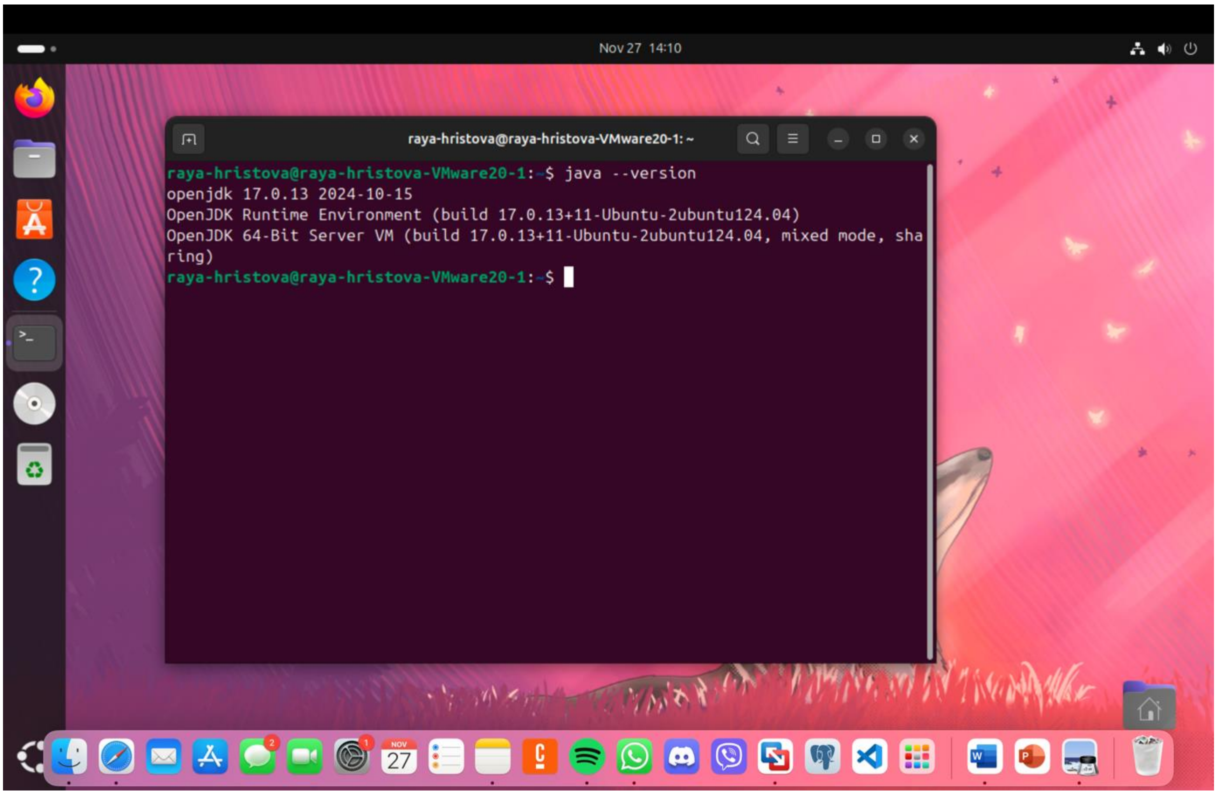
Take screenshots that the following commands work:

javac --version



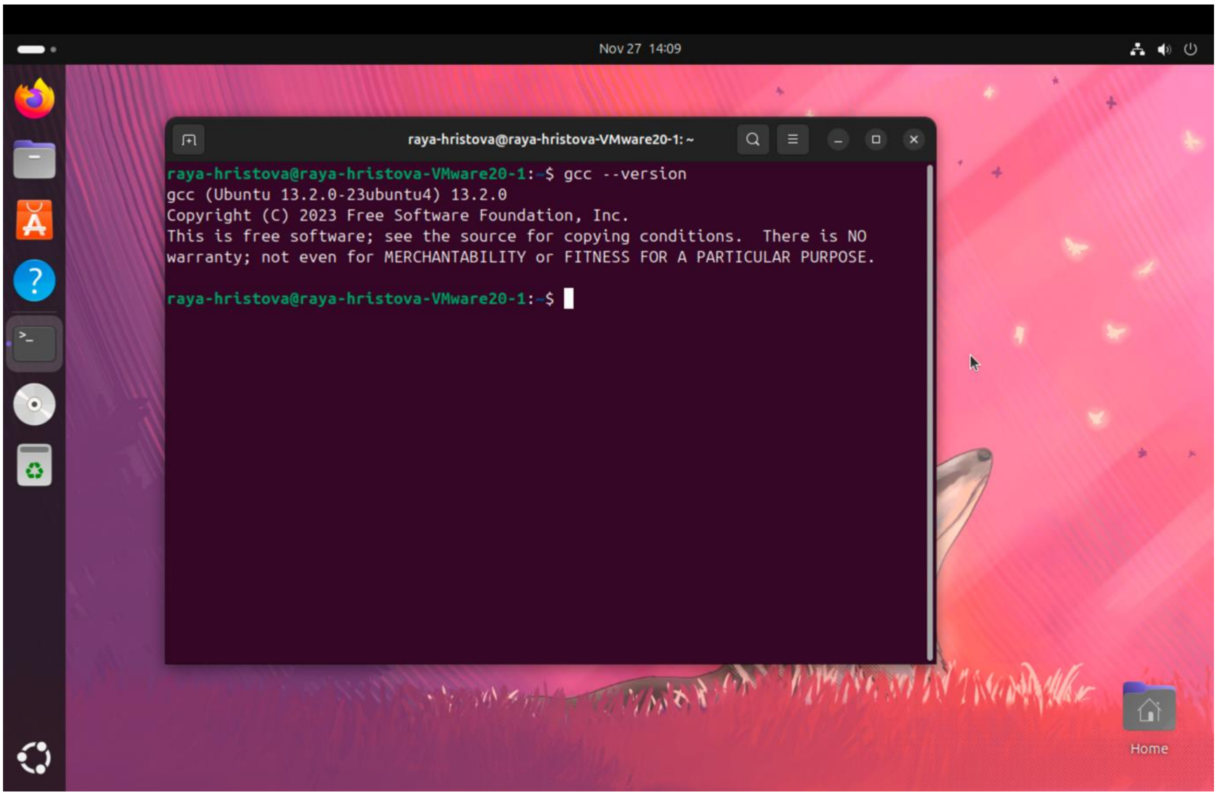
```
raya-hristova@raya-hristova-VMware20-1:~$ javac --version
javac 21.0.5
raya-hristova@raya-hristova-VMware20-1:~$ 
```

java --version



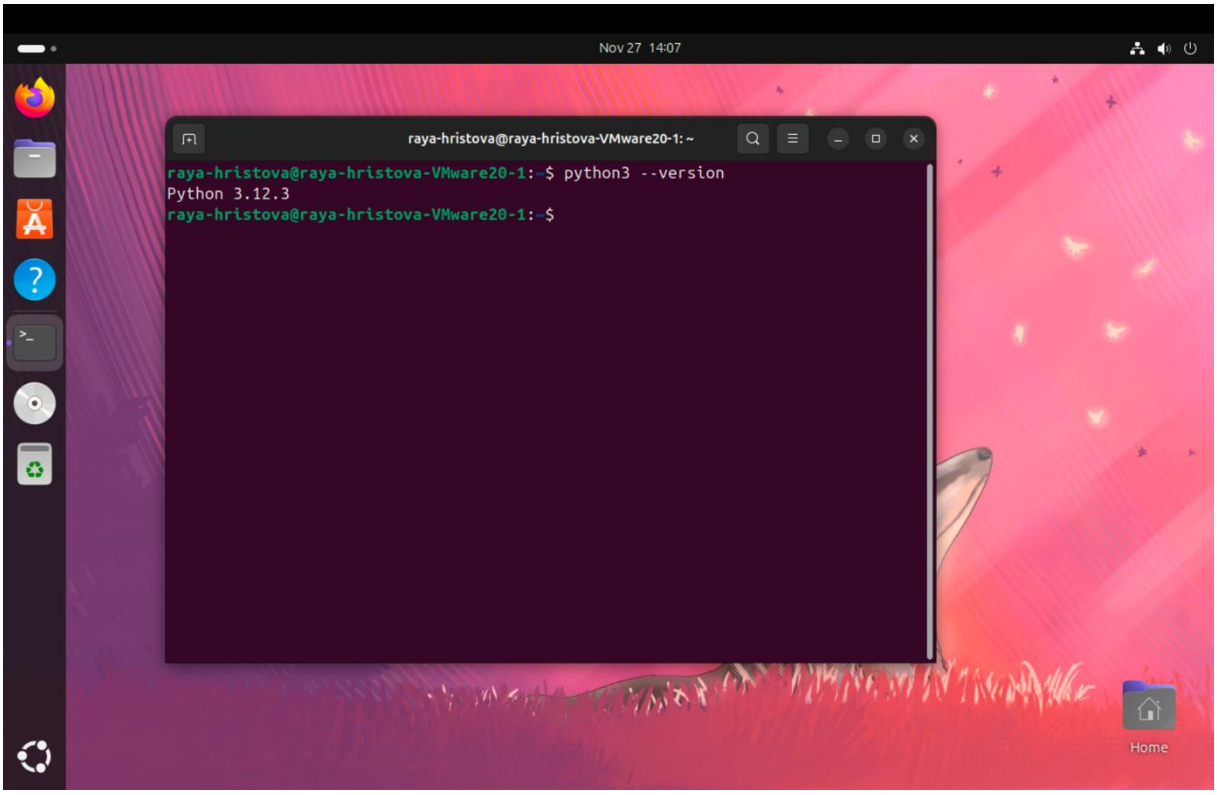
```
raya-hristova@raya-hristova-VMware20-1:~$ java --version
openjdk 17.0.13 2024-10-15
OpenJDK Runtime Environment (build 17.0.13+11-Ubuntu-2ubuntu124.04)
OpenJDK 64-Bit Server VM (build 17.0.13+11-Ubuntu-2ubuntu124.04, mixed mode, sha
ring)
raya-hristova@raya-hristova-VMware20-1:~$ 
```

gcc --version



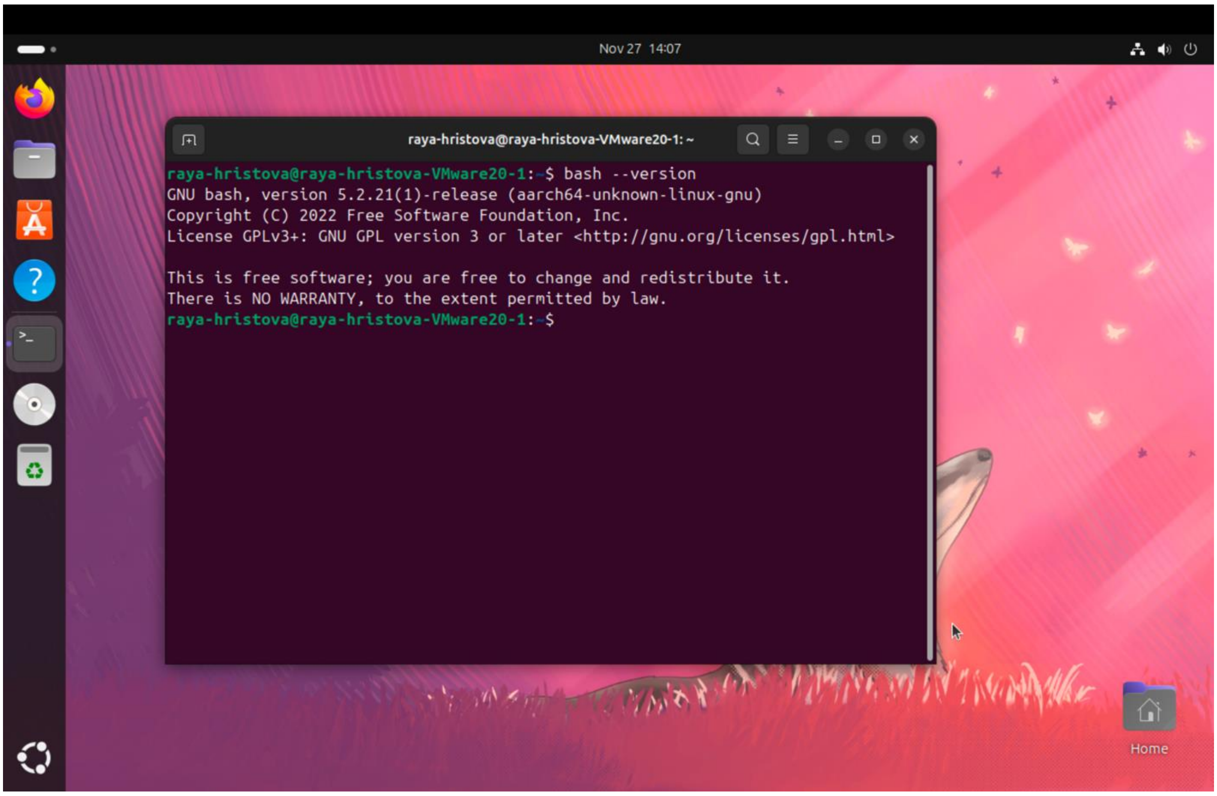
```
raya-hristova@raya-hristova-VMware20-1:~$ gcc --version
gcc (Ubuntu 13.2.0-23ubuntu4) 13.2.0
Copyright (C) 2023 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

raya-hristova@raya-hristova-VMware20-1:~$ 
```

python3 --version



```
raya-hristova@raya-hristova-VMware20-1:~$ python3 --version
Python 3.12.3
raya-hristova@raya-hristova-VMware20-1:~$
```

bash --version



```
raya-hristova@raya-hristova-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.
raya-hristova@raya-hristova-VMware20-1:~$
```

Assignment 4.3: Compile

Which of the above files need to be compiled before you can run them? – fib.c, 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? – fib.py, 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, because it's compiled in machine code.

How do I run a Java program? – Compile the file with javac command and run with java command

How do I run a Python program? – Run program with python3 command

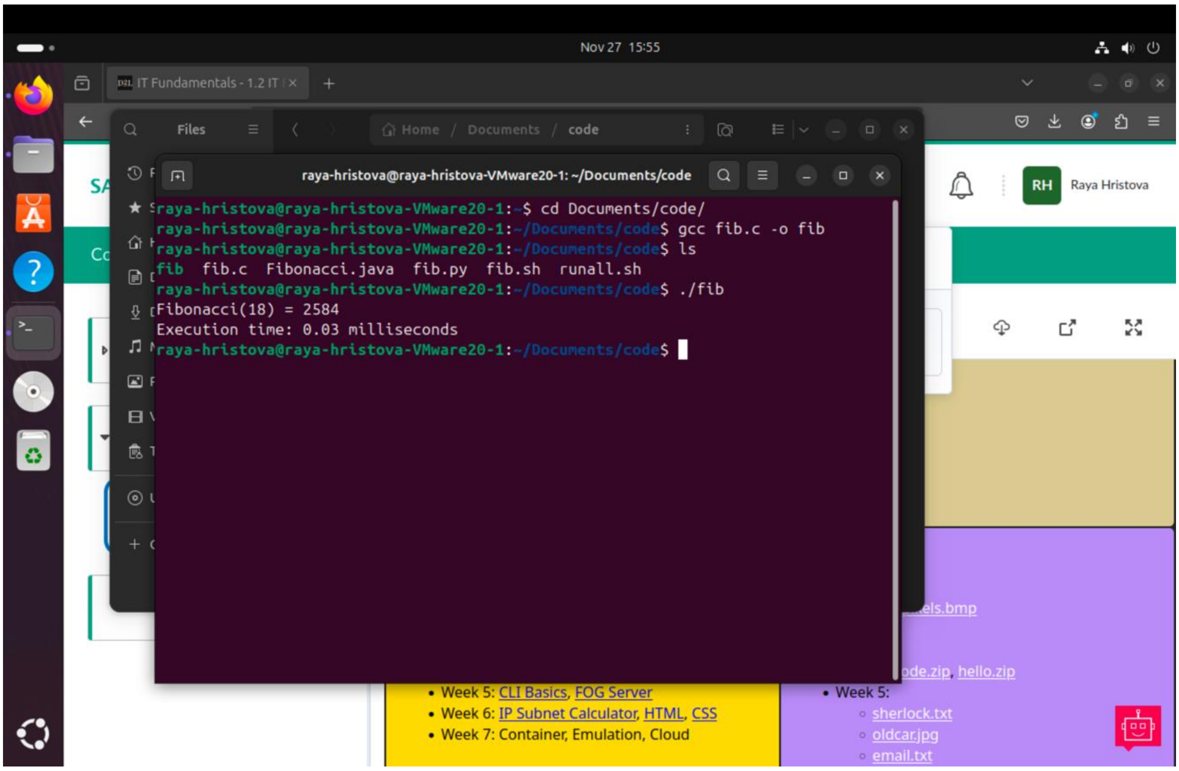
How do I run a C program? – Compile the file into machine code with gcc command and execute

How do I run a Bash script? – Make the script executable and run it

If I compile the above source code, will a new file be created? If so, which file? – A new runnable file is created when compiling the C program into machine code.

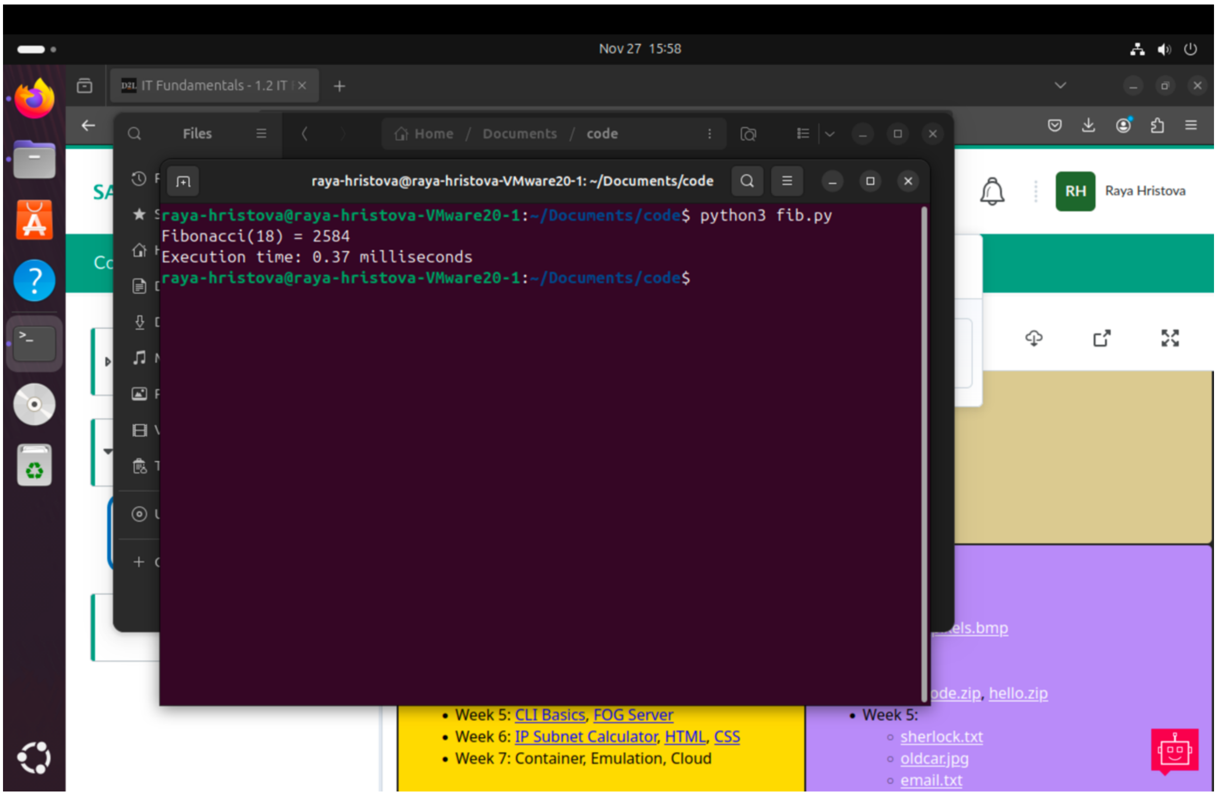
Take relevant screenshots of the following commands:

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



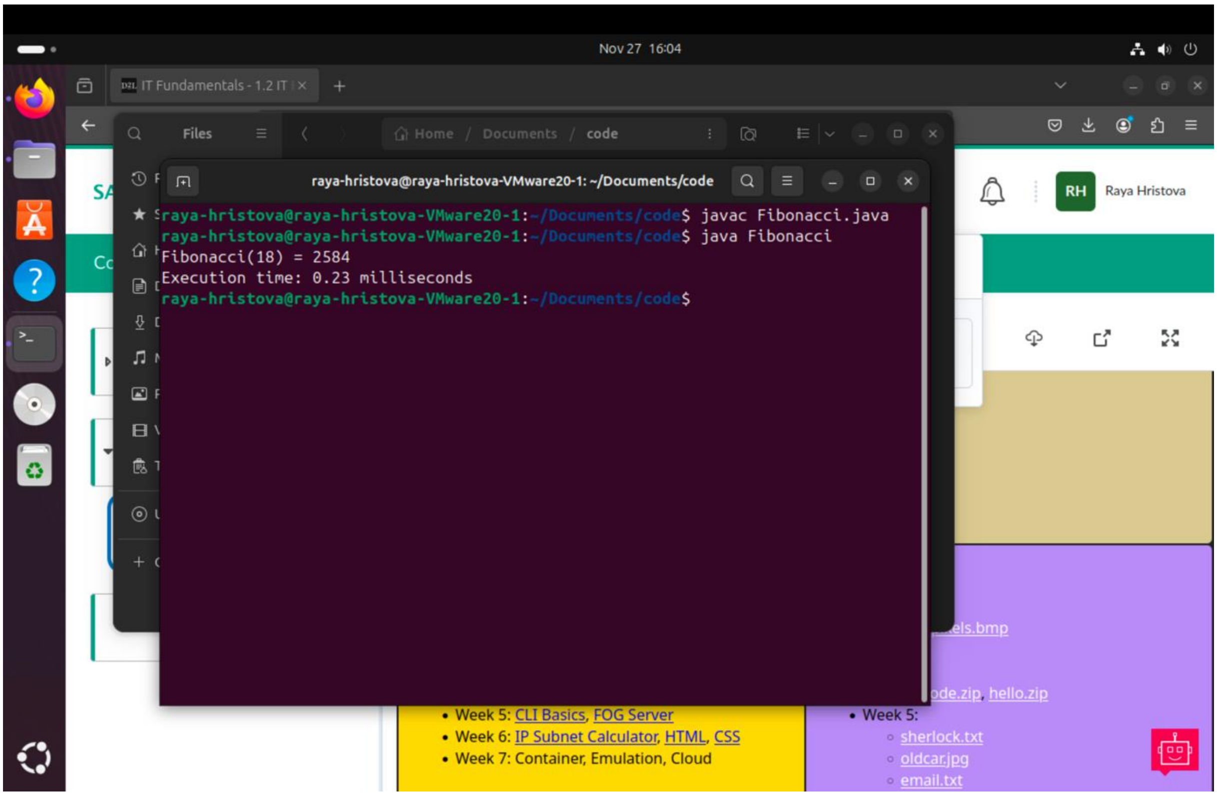
raya-hristova@raya-hristova-VMware20-1: ~/Documents/code

```
raya-hristova@raya-hristova-VMware20-1:~$ cd Documents/code/
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ gcc fib.c -o fib
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ ls
fib  fib.c  Fibonacci.java  fib.py  fib.sh  runall.sh
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ ./fib
Fibonacci(18) = 2584
Execution time: 0.03 milliseconds
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ 
```



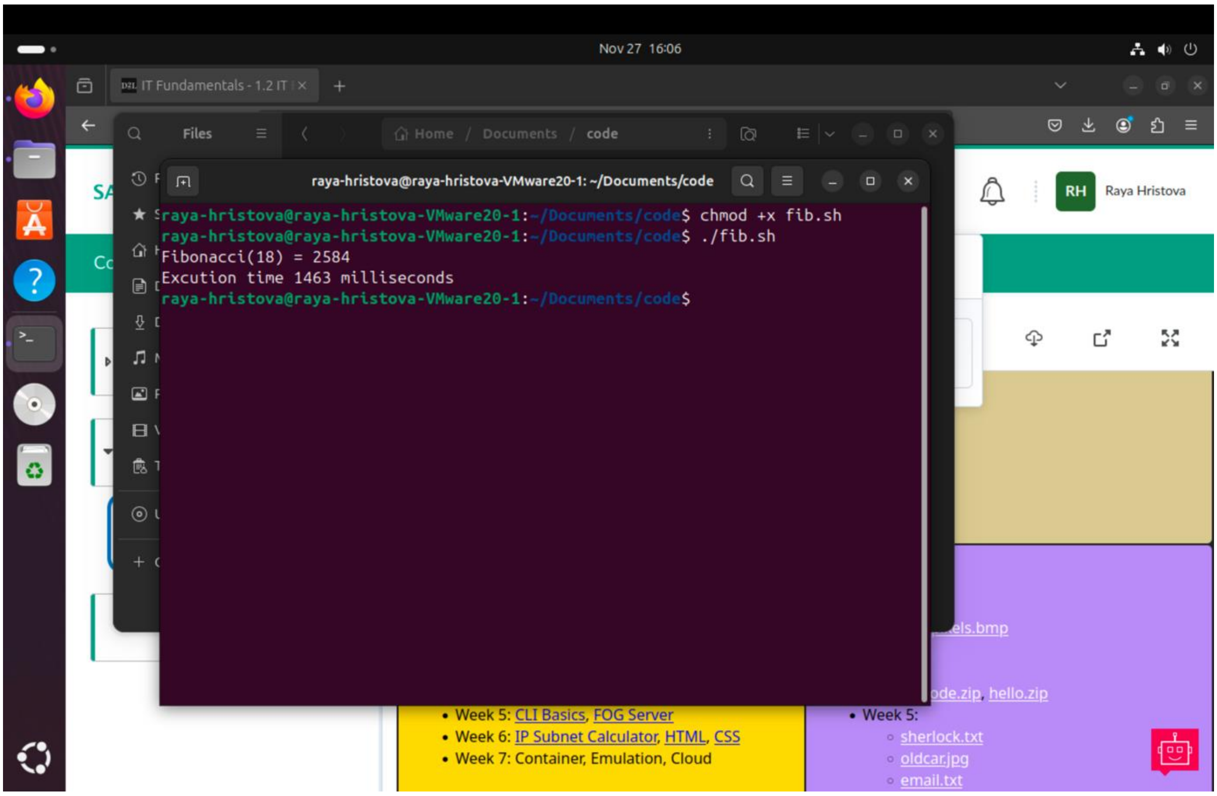
Nov 27 15:58
IT Fundamentals - 1.2 IT
Files
Home / Documents / code
raya-hristova@raya-hristova-VMware20-1: ~/Documents/code
RH
Raya Hristova
Week 5: CLI Basics, FOG Server
Week 6: IP Subnet Calculator, HTML, CSS
Week 7: Container, Emulation, Cloud
els.bmp
ode.zip, hello.zip
Week 5:
sherlock.txt
oldcar.jpg
email.txt

```
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ python3 fib.py
Fibonacci(18) = 2584
Execution time: 0.37 milliseconds
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$
```



Nov 27 16:04
IT Fundamentals - 1.2 IT
Files
Home / Documents / code
raya-hristova@raya-hristova-VMware20-1: ~/Documents/code
RH
Raya Hristova
Week 5: CLI Basics, FOG Server
Week 6: IP Subnet Calculator, HTML, CSS
Week 7: Container, Emulation, Cloud
els.bmp
ode.zip, hello.zip
Week 5:
sherlock.txt
oldcar.jpg
email.txt

```
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ javac Fibonacci.java
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ java Fibonacci
Fibonacci(18) = 2584
Execution time: 0.23 milliseconds
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$
```



```
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ chmod +x fib.sh
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ ./fib.sh
Fibonacci(18) = 2584
Excution time 1463 milliseconds
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$
```

Nov 27 16:06
IT Fundamentals - 1.2 IT
Files
Home / Documents / code
raya-hristova@raya-hristova-VMware20-1: ~/Documents/code
RH
Raya Hristova
Week 5: CLI Basics, FOG Server
Week 6: IP Subnet Calculator, HTML, CSS
Week 7: Container, Emulation, Cloud
Week 5:
sherlock.txt
oldcar.jpg
email.txt

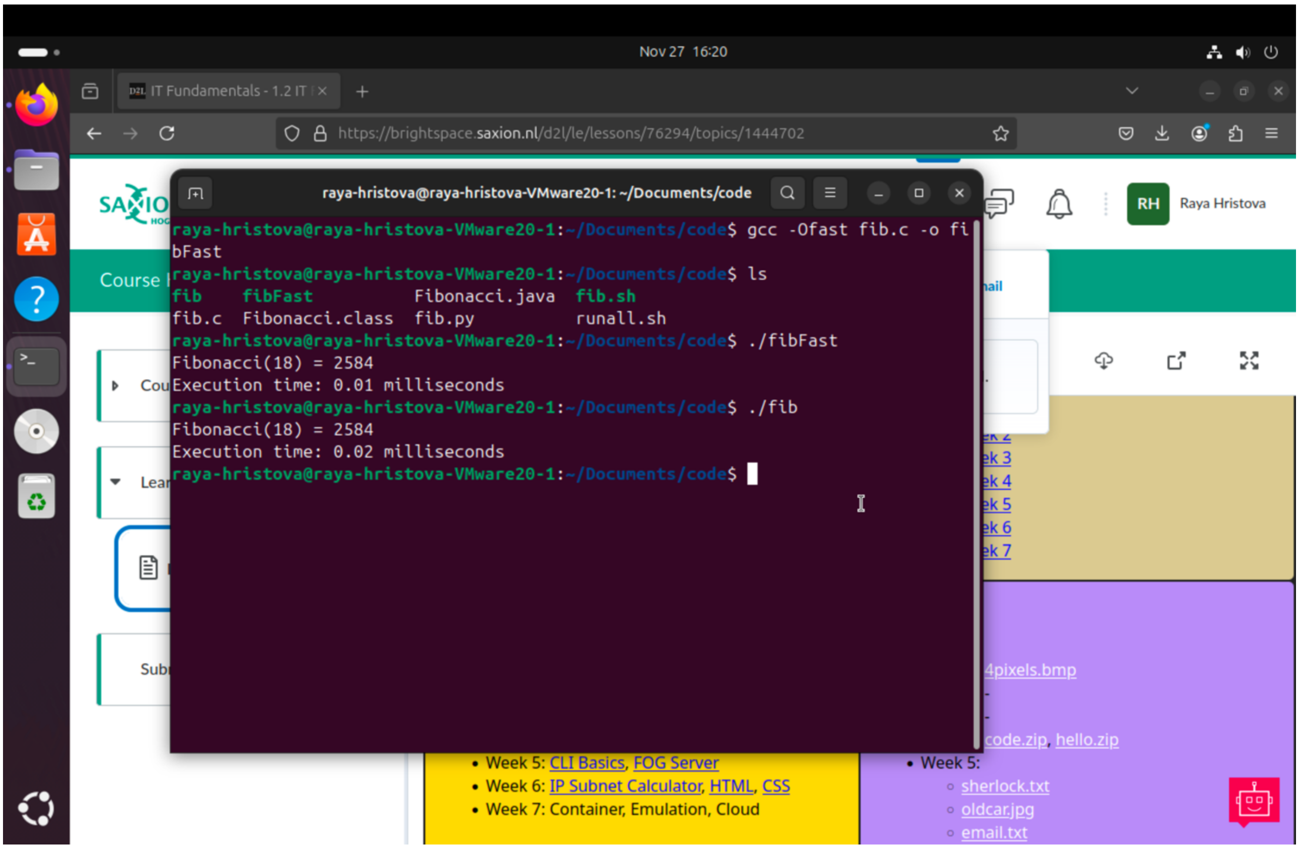
Assignment 4.4: Optimize

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

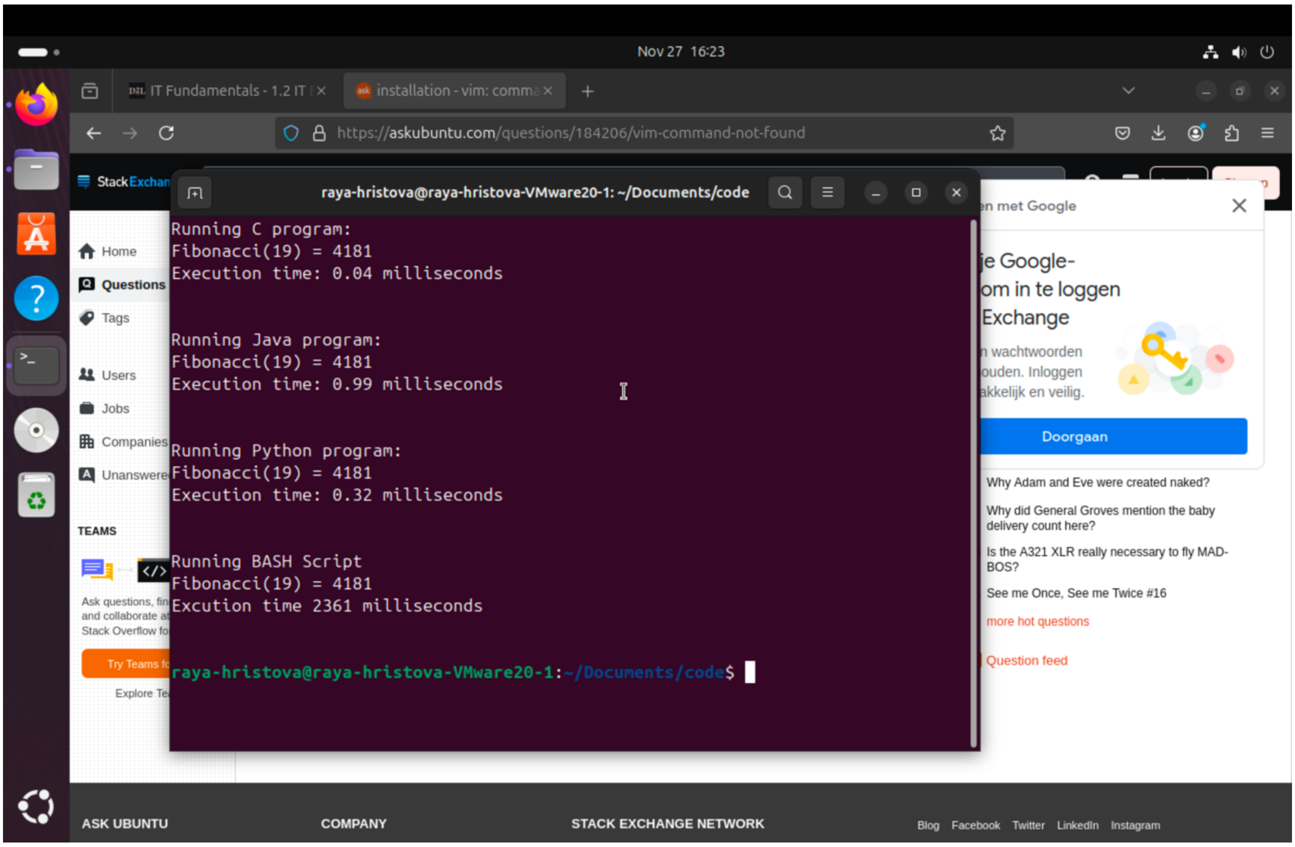
c) Run the newly compiled program. Is it true that it now performs the calculation faster?



Nov 27 16:20
IT Fundamentals - 1.2 IT
https://brightspace.saxion.nl/d2l/le/lessons/76294/topics/1444702
raya-hristova@raya-hristova-VMware20-1: ~/Documents/code
RH
Raya Hristova
Course
Week 5: CLI Basics, FOG Server
Week 6: IP Subnet Calculator, HTML, CSS
Week 7: Container, Emulation, Cloud
4pixels.bmp
code.zip, hello.zip
Week 5:
sherlock.txt
oldcar.jpg
email.txt

```
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ gcc -Ofast fib.c -o fi
bFast
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ ls
fib    fibFast          Fibonacci.java  fib.sh
fib.c  Fibonacci.class  fib.py          runall.sh
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ ./fibFast
Fibonacci(18) = 2584
Execution time: 0.01 milliseconds
raya-hristova@raya-hristova-VMware20-1:~/Documents/code$ ./fib
Fibonacci(18) = 2584
Execution time: 0.02 milliseconds
raya-hristova@raya-hristova-VMware20-1:~/Documents/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.



Nov 27 16:23
IT Fundamentals - 1.2 IT
installation - vim: comma
https://askubuntu.com/questions/184206/vim-command-not-found
raya-hristova@raya-hristova-VMware20-1: ~/Documents/code
Home
Questions
Tags
Users
Jobs
Companies
Unanswere
TEAMS
Doorgaan
more hot questions
Question feed
ASK UBUNTU
COMPANY
STACK EXCHANGE NETWORK
Blog Facebook Twitter LinkedIn Instagram

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


Running Java program:
Fibonacci(19) = 4181
Execution time: 0.99 milliseconds


Running Python program:
Fibonacci(19) = 4181
Execution time: 0.32 milliseconds


Running BASH Script
Fibonacci(19) = 4181
Excution time 2361 milliseconds

raya-hristova@raya-hristova-VMware20-1:~/Documents/code$
```

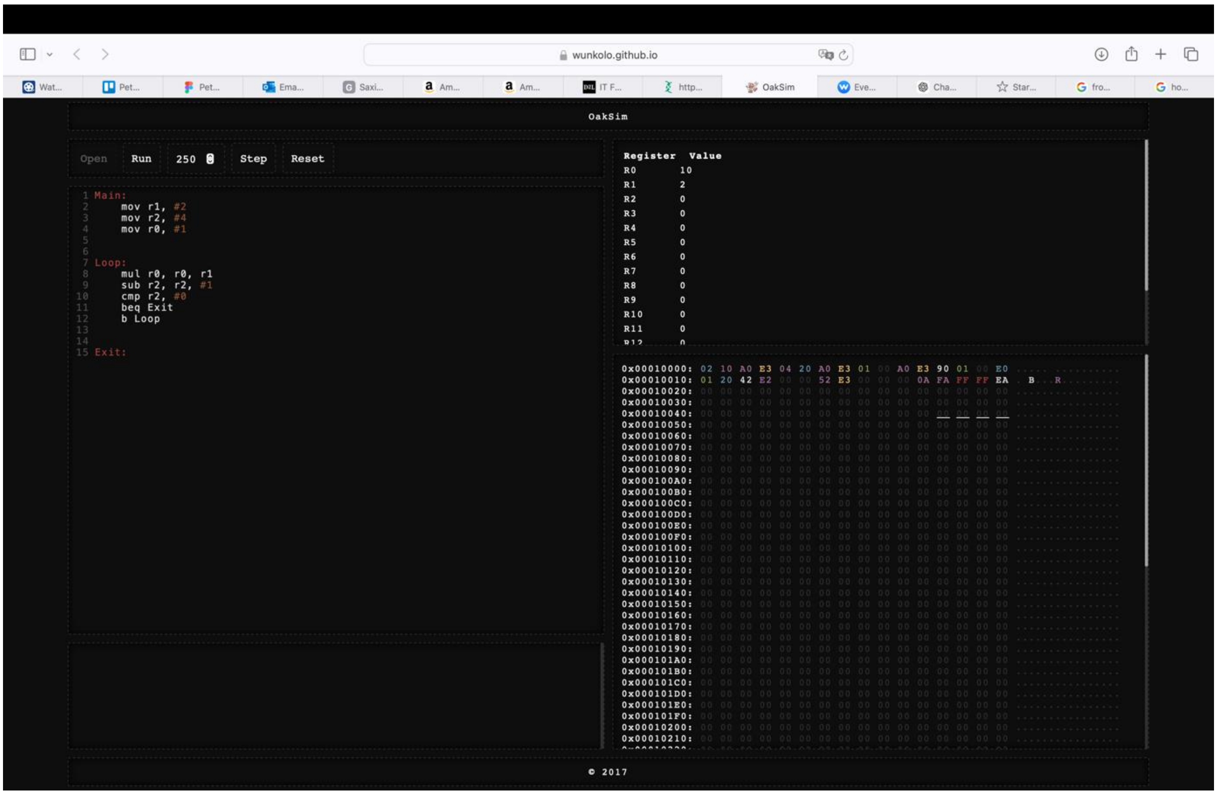
Bonus point assignment – week 4

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.



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


Loop:
    mul r0, r0, r1
    sub r2, r2, #1
    cmp r2, #0
    beq Exit
    b Loop


Exit:
```

```
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
```

```
0x00010000: 02 10 A0 E3 04 20 A0 E3 01 00 A0 E3 90 01 00 E0 ................
0x00010010: 01 20 42 E2 00 00 52 E3 00 00 00 0A FA 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 ................
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 ................
```