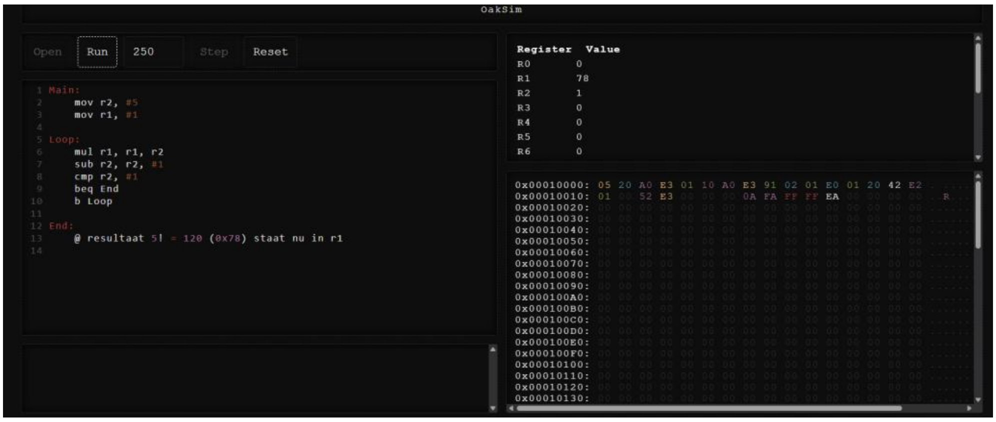
Template Week 4 – Software

Student number: 589948

Assignment 4.1: ARM assembly

Screenshot of working assembly code of factorial calculation:



Assignment 4.2: Programming languages

Take screenshots that the following commands work:

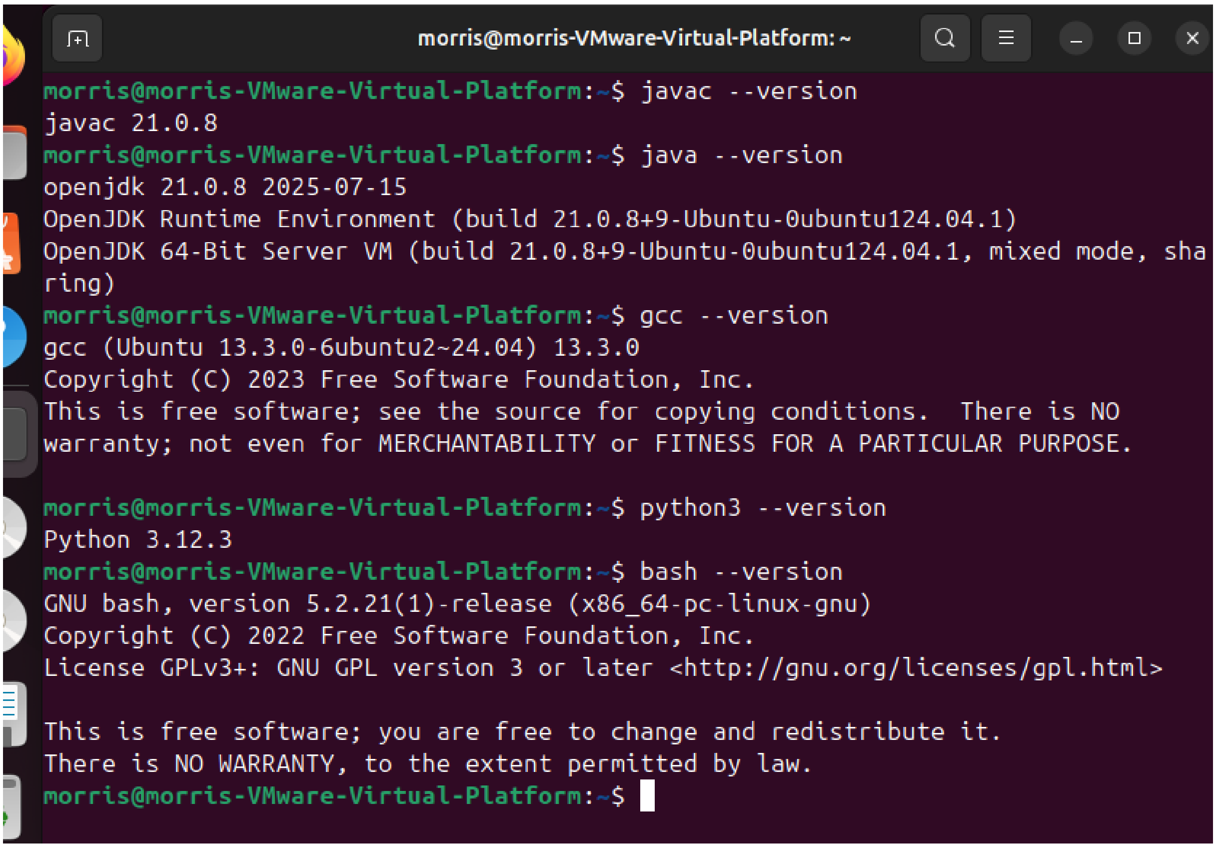
javac --version

java --version

gcc --version

python3 --version

bash –version


morris@morris-VMware-Virtual-Platform: ~
morris@morris-VMware-Virtual-Platform:~$ javac --version
javac 21.0.8
morris@morris-VMware-Virtual-Platform:~$ java --version
openjdk 21.0.8 2025-07-15
OpenJDK Runtime Environment (build 21.0.8+9-Ubuntu-0ubuntu124.04.1)
OpenJDK 64-Bit Server VM (build 21.0.8+9-Ubuntu-0ubuntu124.04.1, mixed mode, sharing)
morris@morris-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.
morris@morris-VMware-Virtual-Platform:~$ python3 --version
Python 3.12.3
morris@morris-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.
morris@morris-VMware-Virtual-Platform:~$

Assignment 4.3: Compile

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

- Fibonacci.java
- fib.c

Which source code files are compiled into machine code and then directly executable by a processor?

- fib.c → na compileren krijg je een native executable (bijv. a.out of fib_c)

Which source code files are compiled to byte code?

- Fibonacci.java → Fibonacci.class (Java bytecode, draait in de JVM)

Which source code files are interpreted by an interpreter?

- fib.py → door de Python-interpreter
- fib.sh → door de bash-shell

These source code files will perform the same calculation after compilation/interpretation. Which one is expected to do the calculation the fastest?

- Het C-programma (fib.c) is het snels, omdat C wordt direct gecompileerd naar machinecode en CPU voert dit meteen uit. Java draait via de JVM echt snel, maar met extra lagen. Python is geïnterpreteerd, dus langzamer. Bash is veel langzamer, dus slecht geschikt voor zware berekeningen.

How do I run a Java program?

-Een Java-programma moet eerst gecompileerd worden. Je voert hiervoor het commando javac uit op het .java-bestand. Hierdoor ontstaat er een nieuw .class-bestand. Daarna kun je het programma uitvoeren door java te gebruiken gevolgd door de bestandsnaam zonder extensie.

How do I run a Python program?

-Een Python-programma hoeft niet gecompileerd te worden. Het wordt rechtstreeks uitgevoerd door de Python-interpreter. Je voert het uit door python3 te gebruiken gevolgd door de naam van het .py-bestand.

How do I run a C program?

-Een C-programma moet eerst gecompileerd worden met een C-compiler, bijvoorbeeld GCC. Het compileren zet de C-broncode om naar een uitvoerbaar bestand. Na het compileren kun je het programma uitvoeren door het gegenereerde uitvoerbestand te starten.

How do I run a Bash script?

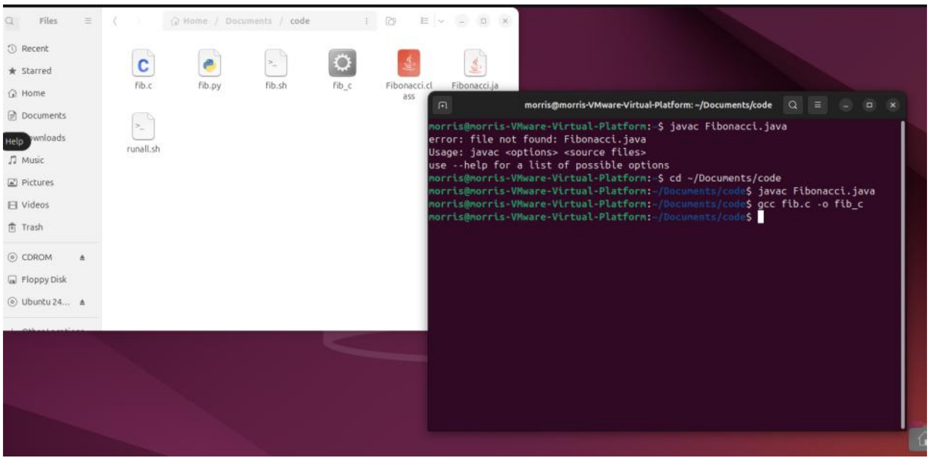
-Een Bash-script hoeft niet gecompileerd te worden, maar het moet wel uitvoerrechten krijgen. Je geeft het script uitvoerrechten met het commando chmod. Daarna kun je het uitvoeren door het scriptbestand direct te starten.

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

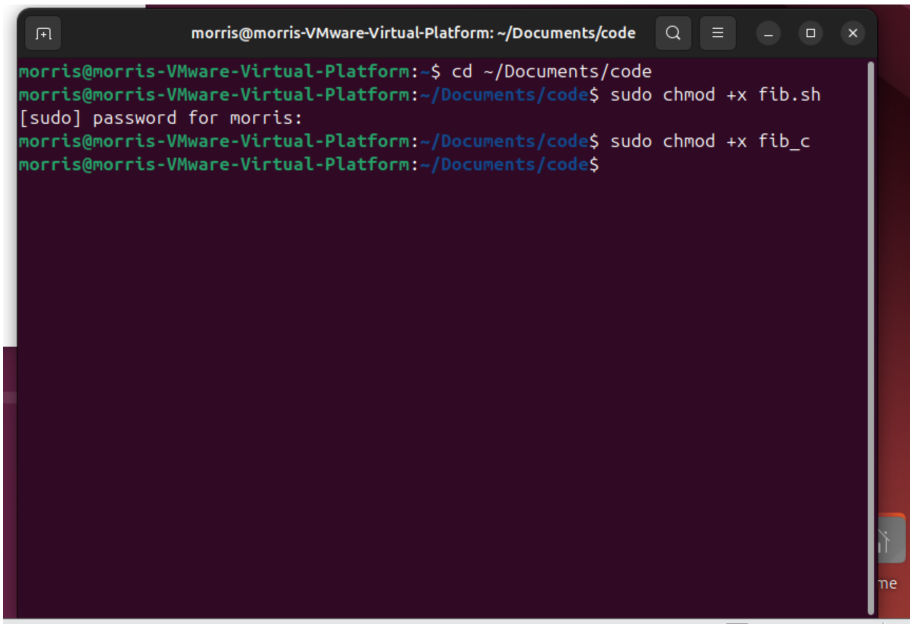
-Bij Java wordt er een nieuw .class-bestand aangemaakt wanneer je het .java-bestand compileert. Bij C wordt er een nieuw uitvoerbaar bestand aangemaakt wanneer je het .c-bestand compileert. Bij Python wordt er geen nieuw bestand aangemaakt omdat Python geïnterpreteerd wordt. Bij Bash wordt er geen nieuw bestand aangemaakt; alleen de permissies van het script veranderen.

Take relevant screenshots of the following commands:

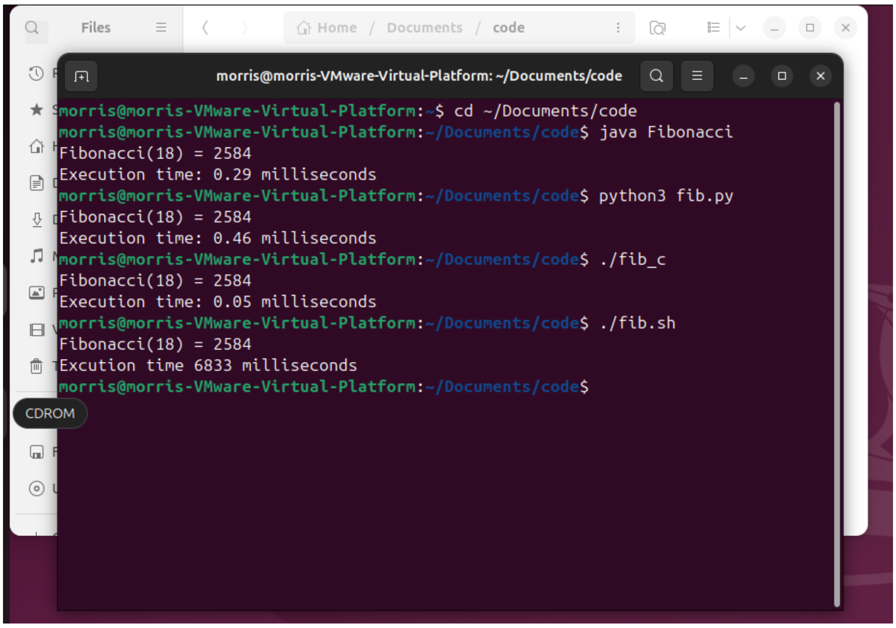
- Compile the source files where necessary



- Make them executable



- Run them

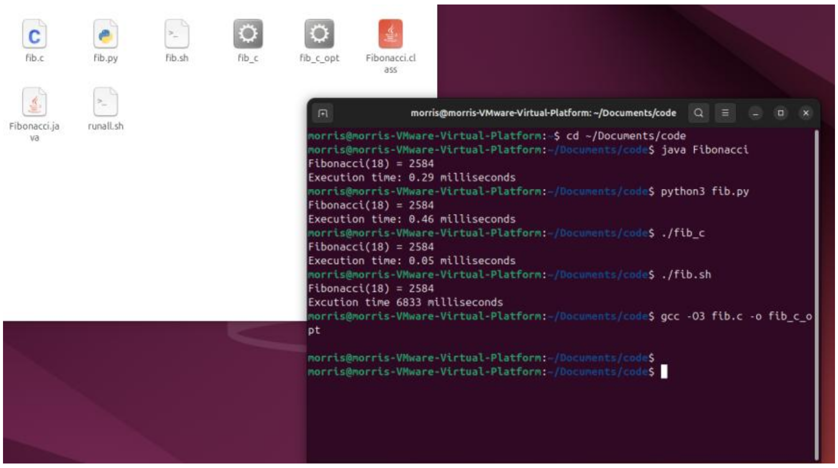


- Which (compiled) source code file performs the calculation the fastest?
  -C-programma is het snelst (0.05 ms)

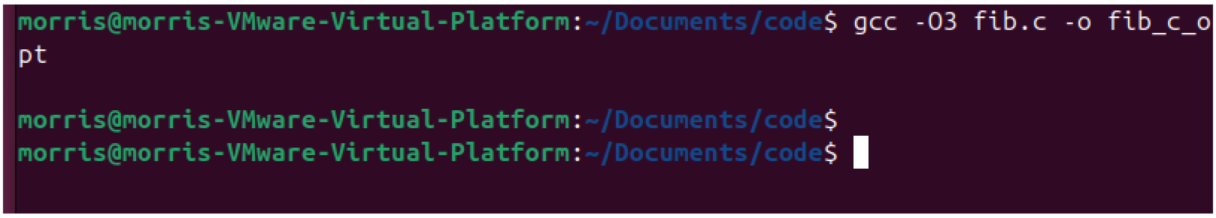
Assignment 4.4: Optimize

Take relevant screenshots of the following commands:

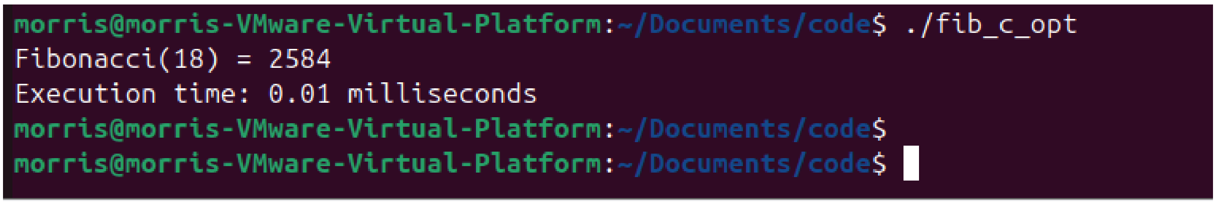
a) Figure out which parameters you need to pass to **the gcc** compiler so that the compiler performs a number of optimizations that will ensure that the compiled source code will run faster. **Tip!** The parameters are usually a letter followed by a number. Also read **page 191** of your book, but find a better optimization in the man pages. Please note that Linux is case sensitive.



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

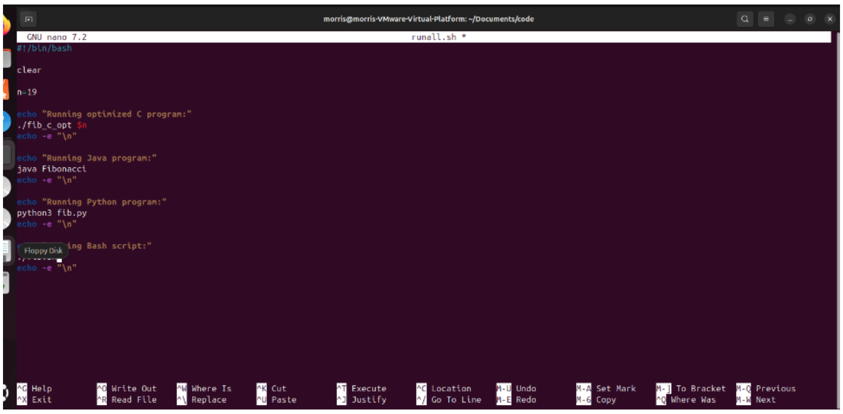


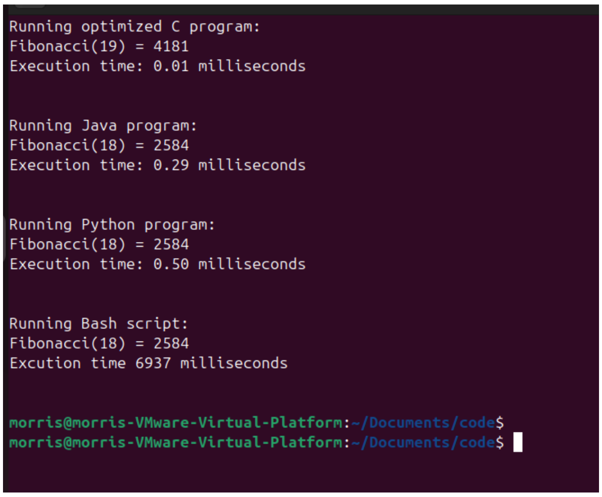
c) Run the newly compiled program. Is it true that it now performs the calculation faster?



Dus sneller dan de vorige: 0.04 milliseconden sneller

d) 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.

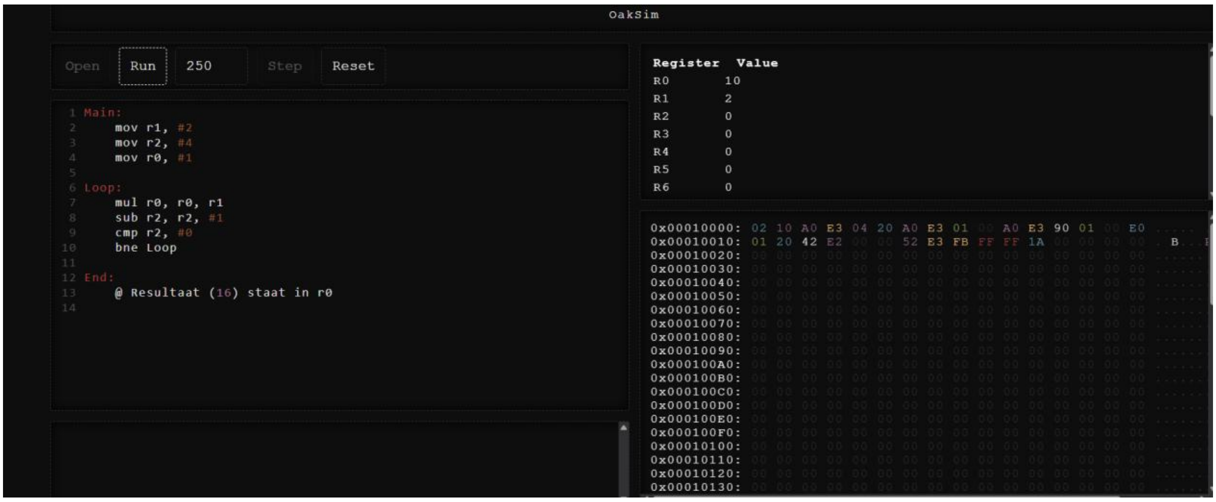
```
Main:
mov r1, #2
mov r2, #4


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**