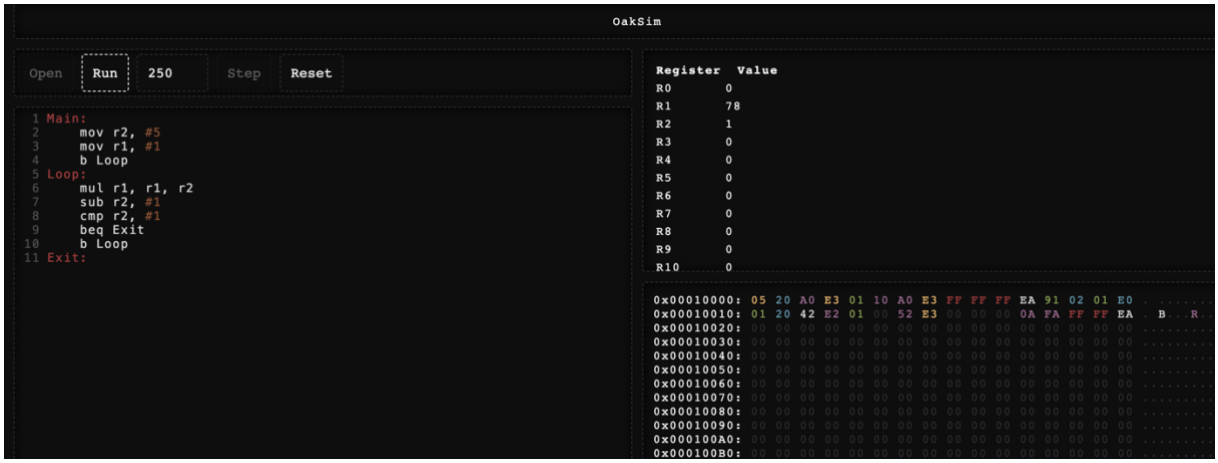
Template Week 4 – Software

Student number: 580606

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:

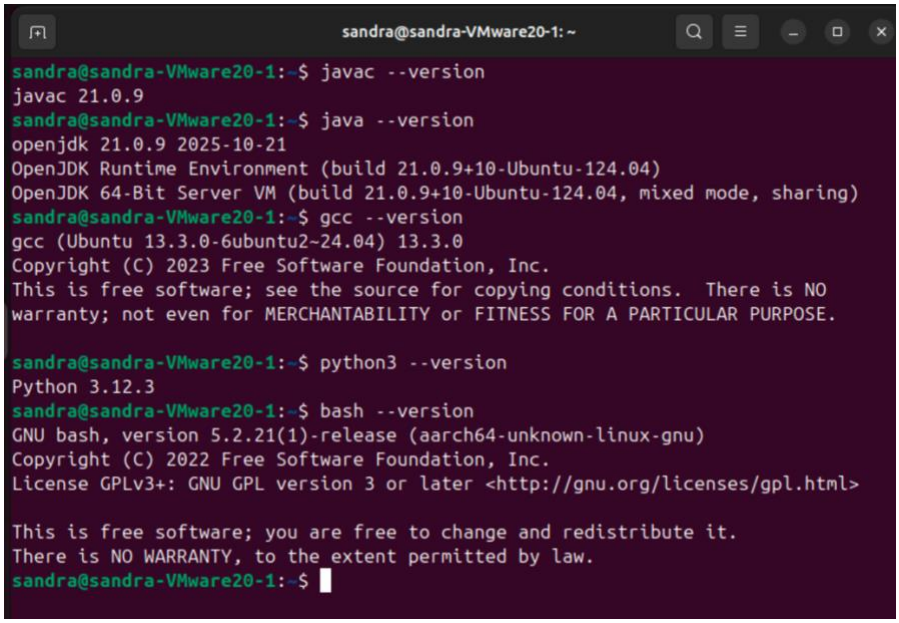
javac --version

java --version

gcc --version

python3 --version

bash --version



Assignment 4.3: Compile

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

Files fib.c and Fibonacci.java have to be compiled first, because C and Java require a compiler before execution.

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

File fib.c is compiled into machine code and then is executed by the computer's processor.

Which source code files are compiled to byte code?

File Fibonacci.java is converted into bytecode before being executed by a VM.

Which source code files are interpreted by an interpreter?

File fib.py is executed by an interpreter.

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

The fastest of the codes should be the fib.c file, because C is a compiled language and the code is translated into machine code, which can then be directly executed by the processor.

How do I run a Java program?

First, you have to make sure you have jdk installed and then run the following commands in Terminal:

javac file.java

java file

The first line compiles the code and the second one executes it.

How do I run a Python program?

First, you have to make sure you have python installed and then run the following command in Terminal:

python3 file.py

How do I run a C program?

First, you have to make sure you have gcc installed and then run the following commands in Terminal:

gcc file.c -o run_file

./run_file

The first line compiles the code and the second one executes it.

How do I run a Bash script?

You have to run the following commands in Terminal:

chmod +x ./file.sh

./file.sh

The first line gives executable permission to the bash file and the second line runs it.

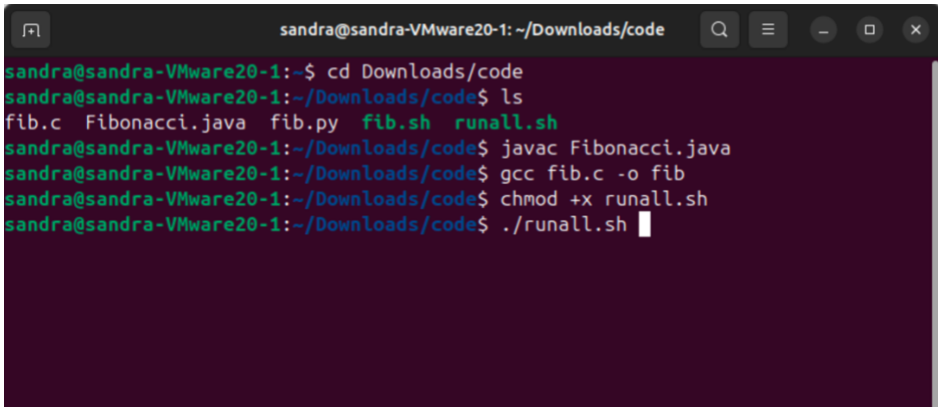
If I compile the above source code, will a new file be created? If so, which file?

Yes, an executable file will be created from the C source code file and a .class file will be created from the .java file.

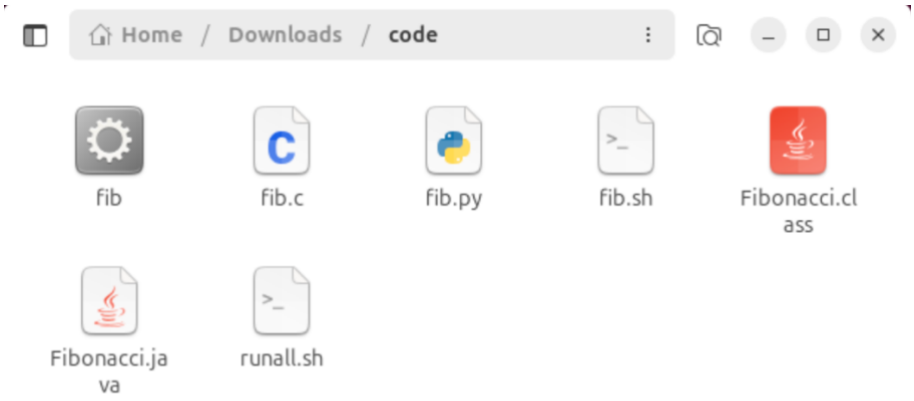
Take relevant screenshots of the following commands:

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

Compiling the C and java code, giving executable permission to the runall bash script:



Two new files fib and Fibonacci.class were created:



The result of executing the runall bash script:

```
sandra@sandra-VMware20-1:~/Downloads/code$ javac Fibonacci.java
sandra@sandra-VMware20-1:~/Downloads/code$ java Fibonacci
Fibonacci(18) = 2584
Execution time: 0.22 milliseconds
sandra@sandra-VMware20-1:~/Downloads/code$
```

```
sandra@sandra-VMware20-1:~/Downloads/code$ python3 fib.py
Fibonacci(18) = 2584
Execution time: 0.46 milliseconds
```

```
sandra@sandra-VMware20-1:~/Downloads/code$ gcc -o fib fib.c
sandra@sandra-VMware20-1:~/Downloads/code$ ./fib
Fibonacci(18) = 2584
Execution time: 0.03 milliseconds
sandra@sandra-VMware20-1:~/Downloads/code$
```

```
sandra@sandra-VMware20-1:~/Downloads/code$ chmod +x ./fib.sh
sandra@sandra-VMware20-1:~/Downloads/code$ ./fib.sh
Fibonacci(18) = 2584
Excution time 1907 milliseconds
```

The fastest calculation was performed by the C source 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.

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

```
sandra@sandra-VMware20-1:~$ cd Downloads/code
sandra@sandra-VMware20-1:~/Downloads/code$ gcc -O2 -o fib fib.c
sandra@sandra-VMware20-1:~/Downloads/code$ ./fib
Fibonacci(18) = 2584
Execution time: 0.01 milliseconds
sandra@sandra-VMware20-1:~/Downloads/code$
```

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

Yes, without the optimization parameters the code runs in 0.03 milliseconds, and with the optimization it compiles in 0.01 millisecond.

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.

```
Running C program:
Fibonacci(19) = 4181
Execution time: 0.05 milliseconds


Running Java program:
Fibonacci(19) = 4181
Execution time: 0.23 milliseconds


Running Python program:
Fibonacci(19) = 4181
Execution time: 0.44 milliseconds


Running BASH Script
Fibonacci(19) = 4181
Excution time 3209 milliseconds
```

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.