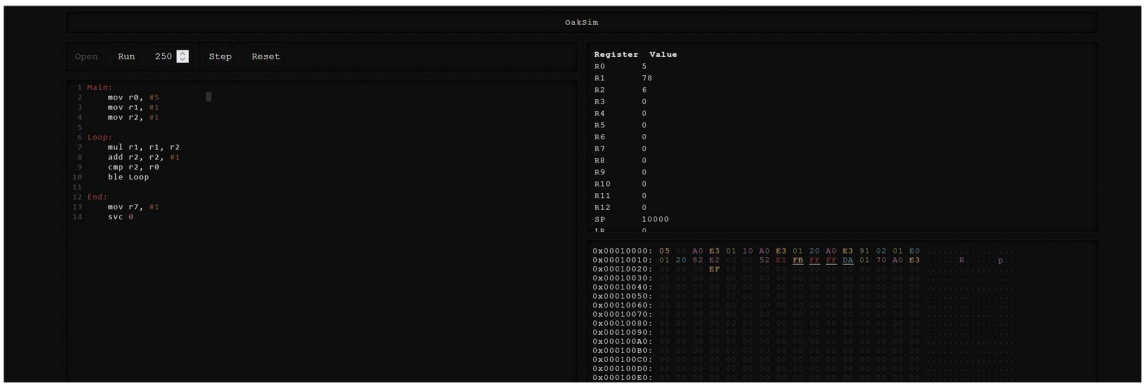
Template Week 4 – Software

Student number: 569681

Assignment 4.1: ARM assembly

Screenshot of working assembly code of factorial calculation:

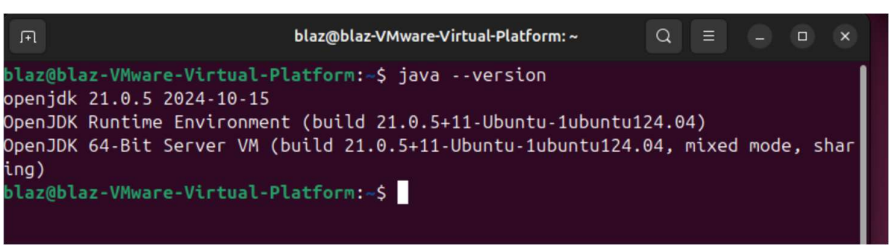


Assignment 4.2: Programming languages

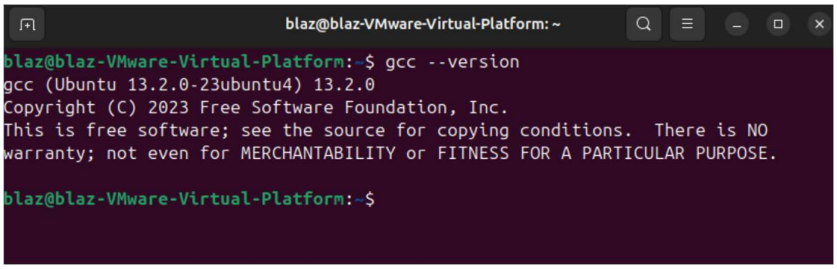
Take screenshots that the following commands work:

javac –version

java –version

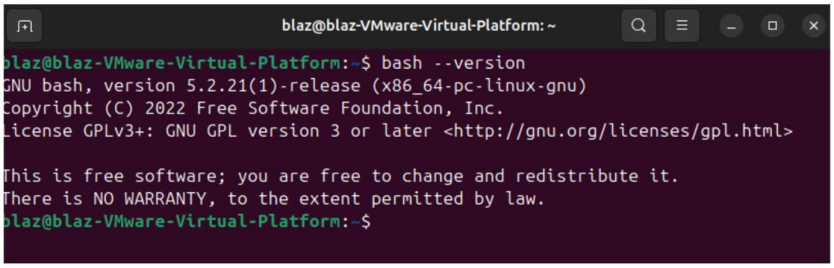


gcc –version



python3 –version

bash --version



Assignment 4.3: Compile

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

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 and 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

How do I run a Java program?

Compile with javac Fibonacci.java and run with java Fibonacci

How do I run a Python program?

Run with python3 fib.py.

How do I run a C program?

Compile with gcc fib.c -o fib and run with ./fib.

How do I run a Bash script?

Make it executable with chmod +x fib.sh and run with ./fib.sh (or run directly with bash fib.sh).

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

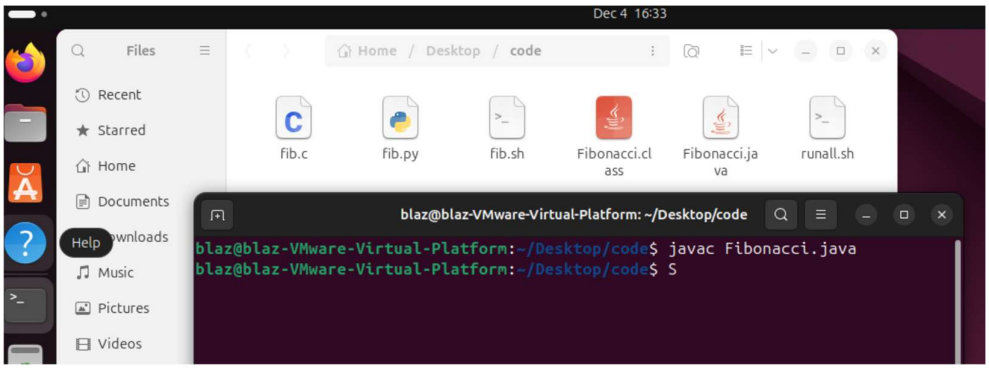
Java: Compiling creates Fibonacci.class.

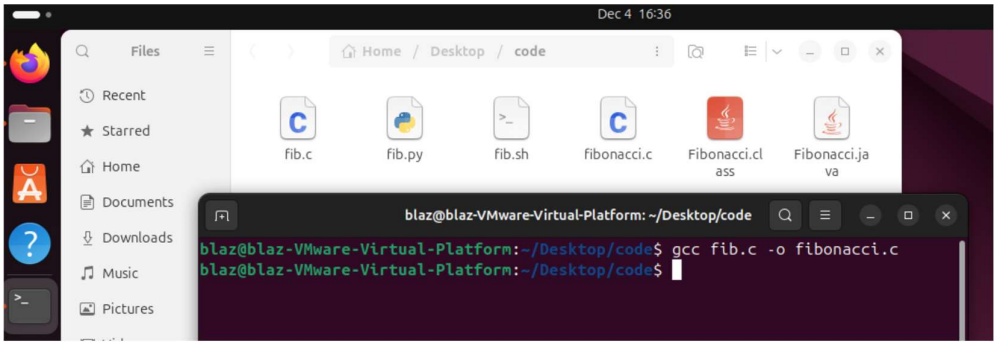
C: Compiling creates an executable file, typically fib.

Python and Bash: No new file is created since they are interpreted.

Take relevant screenshots of the following commands:

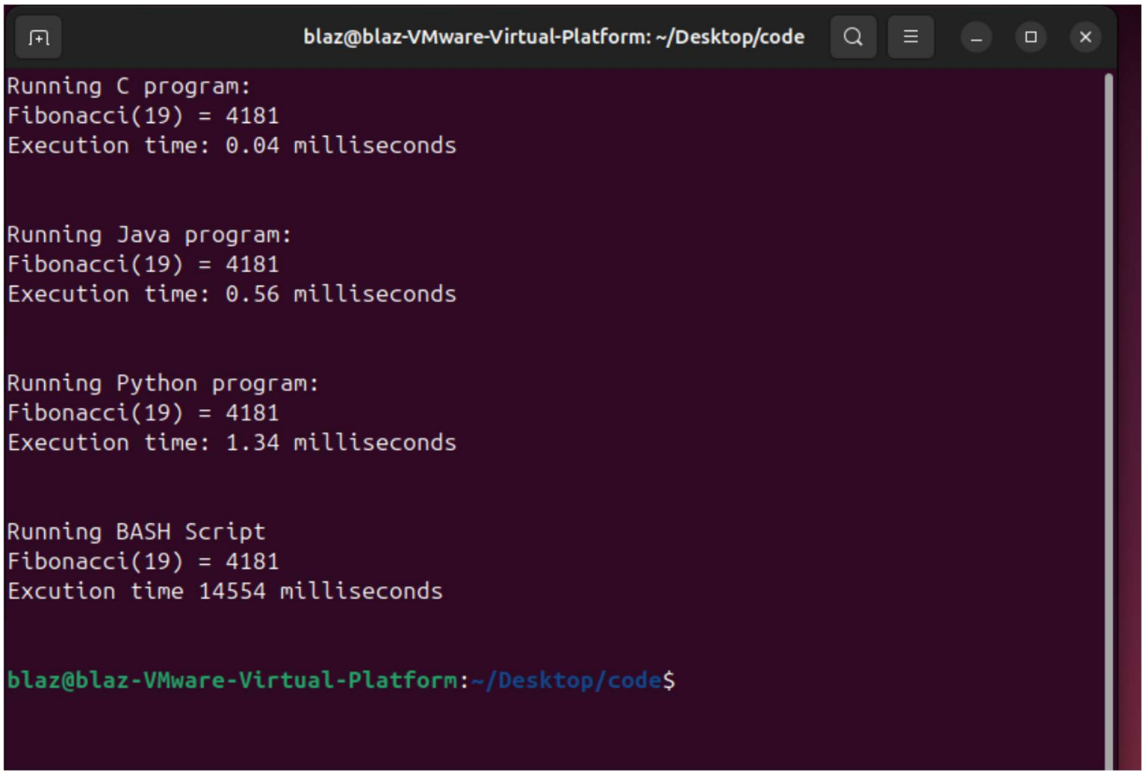
- Compile the source files where necessary





- Make them executable
- Run them

- Which (compiled) source code file performs the calculation the fastest?



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.

   -o turns on the optimization flags.

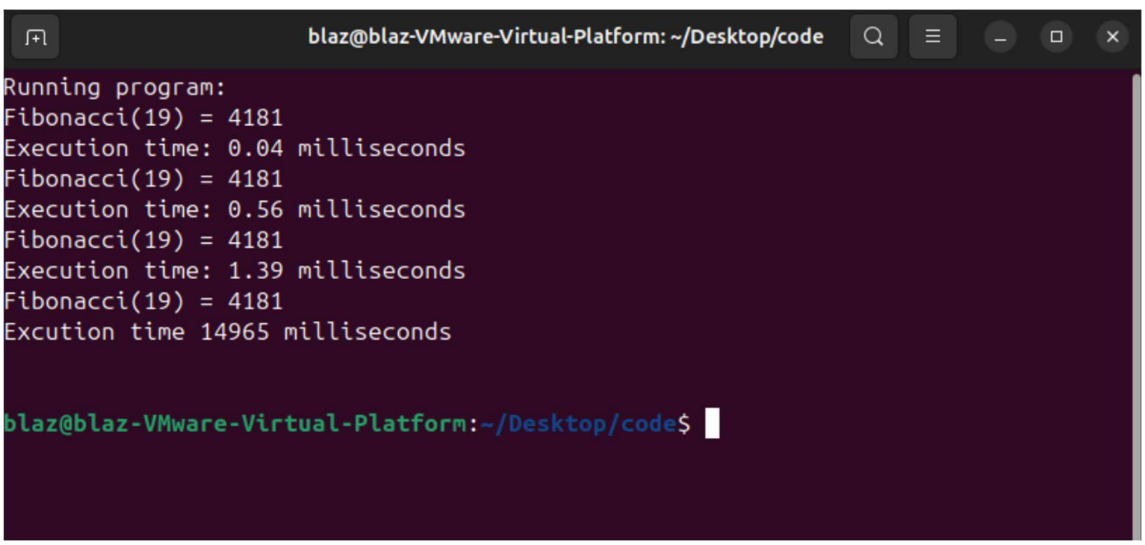
b) Compile **fib.c** again with the optimization parameters

   Already done, see task 4.3

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

   The program runs a lot faster than the normally compiled c file.

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.



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 r0, #2
    mov r1, #2

Loop:
    mul r0, r0, r1

    cmp r2, #16

    beq End
    bne Loop

End:
    svc 0
```

Complete the code. See the PowerPoint slides of week 4.

Screenshot of the completed code here.