LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Summer Term 2021
Dr.-Ing. M. Mohr
Dr. J. Oeser

Scientific Programming
Assignment Sheet 2

Exercise 2 (*Sandbox*)

The slides, assignment sheets and sample solutions of the course on Scientific Programming are managed using the version control system Subversion. In this exercise you will checkout a working copy (a.k.a. sandbox) of this project.

Make sure the *host computer* you are running the Kubuntu virtual machine on is connected to the internet. Then open Konsole and execute the following command:

```
svn co svn+ssh://lxuksp@sshproxy-m.geophysik.uni-muenchen.de/SP2021
```

Exercise 3 (*Using Subversion*)

In this exercise we are going to play a little with the version control tool *subversion*.

a) In exercise 2 you already checked out a working copy (a.k.a. sandbox) of the SP2021 project. Change into the sandbox using the `cd` command and update it by executing the command `svn update`. What do you observe? Now run `svn up`. What happens?

b) In your home directory you will find the file `~/.subversion/config`. Edit this file:

- Comment in the line saying `editor-cmd`, i.e. remove the hash(es) and spaces in front of `editor-cmd`.
- Make `Emacs` your editor for commit messages by setting the value to `editor-cmd = /usr/bin/emacs -nw` (Or choose your favourite editor.) What is the default?

c) Now change to the sub-directory `04_Playground` of the root of the project directory. Generate a sub-directory named according to your family name. Add a file `myOrigin.txt` to this new sub-directory with a brief description of your home town. Commit these changes to the repository.

d) In the sub-directory `05_Working-Groups` you will find a sub-directory with the name of your team.

- Do you see the sub-directories of the other working groups?
- Check that you have write access to this part of the project by generating a file in your working group's directory and committing it.

e) Change to the root of the sandbox. There you will see a file called `PersistentFile.txt`.

- Try to remove the file from the project by running `svn remove PersistentFile.txt` and then committing this change. What happens?
- Check the status of your sandbox by running `svn status`. What is reported for `PersistentFile.txt`.

- Run `svn revert PersistentFile.txt` to avoid problems with future commits.

If you want to learn more about subversion you may e.g. visit the webpage http://svnbook.red-bean.com.

Exercise 4 (*Emacs Configuration*)

On the Moodle pages for this course you will find suggestions for configuring the Emacs editor. Emacs is the suggested editor for this course. If you plan on using it, please add at least the line associating the `F5` key to the `compile` action to your emacs configuration file. If the file `~/.emacs` does not exist, simple create it.

Exercise 5 (*Hello World*)

Write a *'Hello World!'* program in C and compile it with the `gcc` compiler. Instruct the compiler to name the executable `helloWorld`.

Exercise 6 (*Virtual Processor*)

A very simple model of a computer based on the von Neumann architecture is the *register machine*. In this exercise, which is based on material from *IT-Handbuch für Fachinformatiker — Der Ausbildungsbegleiter* by S. Kersken, we are going to examine programs for such a virtual processor.

Our virtual computer consists of two components:

- a virtual processor, which is composed of two registers (`A` and `B`) for computations and a status register `E`
- main memory, with addresses 0 – 199.

The processor makes no distinction between different types of numbers and can store arbitrarily large numbers in each register and memory cell. It uses assembler like commands as machine code. The available commands are the following:

`MOV dst, src`	copies the value stored in `src` to `dst`; here `dst` and `src` can either be one of the two registers `A` and `B` or `$addr`, with `addr` $\in \{0, \ldots, 199\}$ being an address in main memory
`MOV dst, val`	copies the constant number `val` to `dst`, e.g. `MOV A, 2` copies the value 2 into register `A`

Note: For the following commands `dst` must be one of the registers `A` or `B`

`ADD dst, src`	adds the content stored in `src` to that stored in `dst`, storing the result in `dst`
`SUB dst, src`	subtracts the content stored in `src` from that stored in `dst`, storing the result in `dst`
`MUL dst, src`	multiplies the content stored in `src` with that stored in `dst`, storing the result in `dst`
`DIV dst, src`	divides the content stored in `dst` by that stored in `src`, storing the result in `dst`; if the content of `src` is 0, the content of `dst` remains unchanged and the error register `E` is set to 1

`HLT`	program gets stopped

Example:

```
MOV A, $0   ⟶ content of memory address 0 is copied into register A
ADD A, $1   ⟶ content of memory address 1 is added to content of register A
MOV $2, A   ⟶ content of register A is copied to memory address 2
HLT         ⟶ program is stopped
```

We now assume that the memory addresses 0 and 1 have been initialised to hold the values 3 and 4. The following table shows the contents of the registers and memory cells involved, when we run our program (after the command on the left was executed):

command	`A`	**0**	**1**	**2**
—	-	3	4	-
`MOV A, $0`	3	3	4	-
`ADD A, $1`	7	3	4	-
`MOV $2, A`	7	3	4	7

Task:

Assume that the main memory of our virtual computer is initialised as follows:

address	0	1	2	3	4
value	200	5	10	3	2

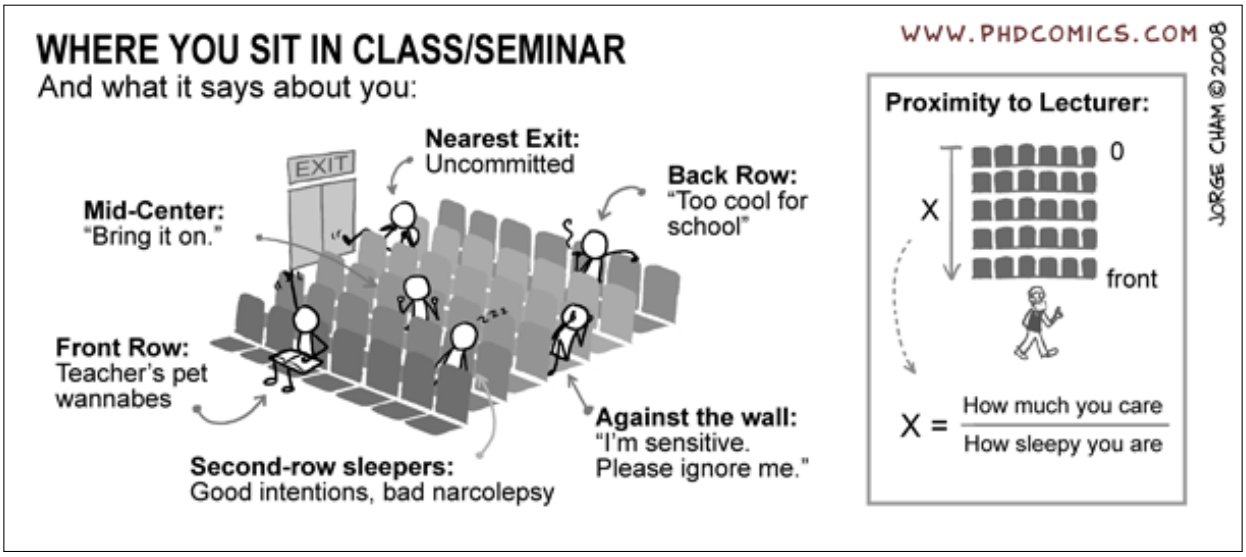
Now write a program that evaluates the formula

$$\Big[(2+3)\cdot 10 + 200/5\Big]/10$$

and stores the result in address 0. As in the example combine your program with a table that shows the contents of the registers and memory cells involved.

Organisational Info:

- Please submit one solution for exercise #6 per group in Moodle before Monday, 03 May 2021, 12:00.



„Piled Higher and Deeper“ by Jorge Cham: www.phdcomics.com