CS Basics - Exercises
Compilation of C code

C. Grothoff and E. Benoist

Fall Term 2017-18

1 Addapt the makefile of an existing project

Pull the solution for the exercises about libraries and arrays. Then:

- Add the C preprocessor instructions (`#ifndef` `#define` and `#endif`) needed inside the `.h` file for preventing multiple insertions. You may need to have a look at the course of last week.
- Change the Makefile to make proper use of the various features discussed in the course.

2 Back to Square 1

Download and compile GNU Hello.

Install mingw-w64 (Debian package, or from `https://sourceforge.net/projects/mingw-w64/files/`).

3 Cross compile "Hello World"

Cross compile the canonical "Hello world":

```
$ i686-w64-mingw32-gcc hello.c -o hello.exe
```

Copy the binary to a W32 system (or wine) and run it.

4 Cross compile GNU Hello

```
$ ./configure --host=i686-w64-mingw32
$ make
```

Copy the binary to a W32 system and run it.

5 Cross compile `sort`

Use `autoscan` and the necessary manual adjustments to create an autotools build system for your `sort` project (unless you already did).

If you use `getline`, use `gnulib-tool --import getline` to install a getline implementation.

Finally, cross-compile `sort` for W32 and test the resulting binary on W32.

6 Bonus: Cross compile to ARM

Install `gcc-arm-linux-gnueabi` and `binutils-arm-linux-gnueabi` or `gnueabihf` for ARM systems implementing "hardfloat".

Your target platform is then `arm-linux-gnueabi(hf)`.

- Cross-compile your `sort` program and run it on the target platform.
- Cross-compile your DNS resolver program. What additional difficulties does this create?

7 Bonus: Cross compile to JavaScript

Install Emscripten. Cross-compile your C code for "Hello World" to JavaScript. Embed it into a Web site. Good luck!