Tutorial 1A: Getting Started with Haskell

In this tutorial we will see how to work with Haskell on campus and at home. We only need the bare basics:

- **GHCi**, the Glasgow Haskell Compiler (interactive), an interpreter that reads your Haskell files and executes Haskell code at the prompt.
- A **standard text editor** of your choice to write your code. Haskell files are standard text files with the extension `.hs`

If you prefer a fancier installation at home with a full IDE, that's fine, but it's not necessary.

On campus

Haskell is installed on campus machines. You can run it as follows:

- Open the start menu
- In the search bar, search for `command prompt` and open the app
- If you want to run Haskell on a file, navigate to that directory
- Type `ghci` , or `ghci <filename>` to open a file

For a text editor, I will use:

- `Start menu` $\rightarrow$ `Notepad++`

You may use any text editor of your choice.

Online

You can work online if you make an account here:

- replit.com

Warning: everything you do on this site is public. You should **not** use this for the coursework.

At home

Installing Haskell at home starts here:

- www.haskell.org/get-started/

This will give you two options:

- A direct install with GHCup: www.haskell.org/ghcup/
- An installation of VSCode, which includes Haskell: code.visualstudio.com/

Running things

Once you have GHCi running, you can do the obligatory "Hello world":

```
*Prelude> putStrLn "Hello world!"
Hello world!
```

The interpreter can evaluate any Haskell expression, which is quite a lot. Let's start with some arithmetic. Simply type in the following lines, to get a feel for things.

```
*Prelude> 2 + 3 * 4 ^ 5
3074
*Prelude> ((2 + 3) * 4) ^ 5
3200000
*Prelude> 2 - 3 / log 4
-0.1640425613334453
*Prelude> sin (0.5 * pi)
1.0
```

You can set variables:

```
*Prelude> x = 6*7
*Prelude> x
42
*Prelude> 2 * x
84
```

And those variables can be functions:

```
*Prelude> square x = x * x
*Prelude> square 123
15129
*Prelude> hyp x y = sqrt (square x + square y)
*Prelude> hyp 3 4
5.0
```

Note how in Haskell a function `fun` with two arguments is written `fun x y`.

Loading files

Let's put those functions in a file. Create a standard text file as `new.hs`, and make sure that GHCi was invoked in the same directory as your file (restart it if necessary). Then in the interpreter, use `:l` to load the file.

```
*Prelude> :l new.hs
Ok, one module loaded.
*Main>
```

The file is empty, but Haskell is happy. It has changed the prompt, which turns out to be the name of the **module** that you've loaded, and if your module has no name, for instance because your file is empty, it settles on `Main`. Open the file in your favourite editor to add the following lines, and don't forget to save.

```
square x = x * x
hyp x y = sqrt (square x + square y)
```

Use `:r` to reload the edited file. If you didn't forget to save, then you can use the functions `square` and `hyp` at the prompt:

```
*Main> :r
Ok, one module loaded.
*Main> square 2345702938
5502322273341831844
*Main> hyp 6 8
10.0
```

And that's it! This will be our workflow for the coming weeks.