

## Week 2 Practical Class Exercises

### Basic Unix and Haskell

---

#### Objectives

This session is an introduction to the Unix operating system and the interactive mode of the Glasgow Haskell Compiler. You will:

- further your use of the Konqueror utility, in interacting with the Unix file system,
- have some hands-on experience with ghci,
- further your experience in creating and editing text files,
- gain some familiarity with the Unix command line.

Once again, you should bring along your copy of *Student Computing Environment: User Guide*.

#### Exercise 1 (Creating Directories)

See Ch 4 of the *Student Computing Environment: User Guide*.

You should create a new directory for yourself whenever it is convenient, in order to organize your work, just as you organize your paperwork into folders and then organize the folders into into binders. If you do not already have a comp1100 directory in your home directory, use Konqueror to create it. Use Konqueror to display its contents (it will be empty). (Later, you should create some subdirectories to organise your comp1100 files as they accumulate.)

Copy the file NumWords.hs from the labs/week02/ sub-directory of /dept/dcs/comp1100/public/ to the comp1100 directory you have just created. Re-read the section from last week's familiarisation exercises if necessary. You own these copies and you are entitled to change them or delete them if you wish. You do not own the public copies and are not able to modify, move or delete them.

#### Exercise 2 (Using Unix)

Launch a terminal window by clicking on the K-menu icon on the KDE front panel ⇒ System Tools ⇒ Konsole. This gives you access to the operating system. You can type Unix commands in a terminal window.

**Dates:** Type `date` at the terminal prompt and hit Enter. You have just run the unix `date` command, to display the current date.

- Type `Date` at the terminal prompt.
- Type `DATE` at the terminal prompt.

What does this tell you about the names of Unix commands?

- Enter the command `cal`.
- Enter the command `cal 2006`.
- Enter the command `cal 7 2006`.
- Use the `cal` command to find out the day of the week on which you were born.
- I recall September 1752 was an interesting month.

**Directory listing:** Enter the command `ls` in a terminal window. The `ls` command is used to list the files and sub-directories in the current directory. (In window interfaces like KDE, directories are often called folders.) When you first log in, the current directory is your own *home directory*. You should see the `comp1100` directory you created earlier.

**Changing directory:** You have already moved around the directory structure using Konqueror. You can also move around using Unix commands.

- Enter the command `cd comp1100` to change into your `comp1100` directory.
- Type `pwd` (path of working directory) to find out what the current directory is.
- Enter the command `cd ..`  
Use the `pwd` command to find out where you've gone.

### Exercise 3 (Interacting with ghci)

We will be using the the interactive interface to the Glasgow Haskell Compiler very frequently in this course. Start up `ghci` by typing `ghci` at a terminal prompt. Ghci behaves rather like a calculator. Evaluate a few expressions like:

```
2 + 3
2 ^ 5
sqrt 2.0
sqrt 4*100    (Check the answer of this one carefully.)
6 < 4
sqrt 15.0 >= 3.5
sin (pi/2.0)
[1..20]
length "This sentence no verb."
words "The quick brown fox jumped over the lazy dog."
reverse "Backwards"
take 11 "Fresh fried fish for free."
map (*2) [10,9..1]
zip [1..10] ['a'..'z']
lookup 5 (zip [1..10] ['a'..'z'])
concat (replicate 10 "Wibble")
```

Notice that at both the shell and `ghci` prompts you can repeat an expression you typed previously by pressing the Up key.

Leave `ghci` by typing `:quit` at the `Prelude>` prompt.

#### Exercise 4 (Load a Haskell script into ghci)

In a terminal window, change to the directory where you earlier put your copy of `NumWords.hs`. Open the script in an editor, but don't change anything. We don't expect you to understand very much of it at this stage. Read the comment at the beginning of the file to get an idea of what it does. To try out the functions defined in this script, it must be loaded into ghci. There are two ways to do this:

- start up ghci, then type `:load NumWords.hs` at the ghci prompt, *or*
- type `ghci NumWords.hs` at the terminal prompt.

Now you can use any of the definitions in `NumWords.hs` in this ghci session. The main function is `convert` so evaluate some expressions like `convert 54321` and `convert 100005`

#### Exercise 5 (Create a Haskell script)

Now we want you to create your own script from scratch, using the editor. Open kate and type in the following script:

---

```
-- Comp1100. Semester 2, 2006.
-- Week 2 Practical.
-- <your name here>

-- Here are a few simple mensuration definitions:
cube :: Int -> Int
cube x = x * x * x

edge, volume :: Int
edge = 3
volume = cube edge

-- If I remember correctly, this is the formula for the surface area
-- of a sphere, in terms of its radius:
surfaceArea :: Float -> Float
surfaceArea r = 4.0 * pi * r^2
```

---

Save the file to your `comp1100` directory with an appropriate name, like `mensuration.hs`. Don't forget that the suffix must be `.hs` or ghci won't recognise it as a Haskell script.

To use the script you have just created, load it into ghci as in the previous exercise. If you haven't made any typing mistakes, ghci should successfully load your script, and you will be able to try out the definitions.

If you have made a typing mistake, ghci will give you an error message while attempting to load the script. The error message itself may seem difficult to understand at this stage, but it will tell you the line number (and character position) at which ghci could not continue. Note that the error will often *precede* that line number. Correct the error in the editor, save the file and attempt to reload it into ghci, by typing `:reload` at the ghci prompt.

When you have successfully loaded your script, you should experiment by getting ghci to evaluate a few expressions using the definitions in your script.

**Exercise 6 (Example code from lectures)**

Open up Konquerer and go to the comp1100 web-site at <http://cs.anu.edu.au/student/comp1100>. Click on the lectures link at the top of the page to get the list of lecture notes and example script files.

Download the `RestRate.hs` and `Length.hs` scripts from last Friday's lecture by right clicking on them and selecting *Save link as*. Start up GHCi and use the appropriate function in `RestRate.hs` to work out your resting metabolic rate.

**Exercise 7 (Prelude documentation)**

Go back to the comp1100 website and select the Haskell link at the top of the page. On this page are a number of documents that you may find useful over the course of the semester.

Select the link to the page which describes the functions in the *Standard Prelude*.

Find the description of the `length` function. Is this definition of `length` the same as the one in the `Length.hs` script from the previous exercise. If not, then what is different?

Show your tutor your completed work.

***Make sure you terminate your session by clicking the logout button on the front panel.***