Lecture 2C

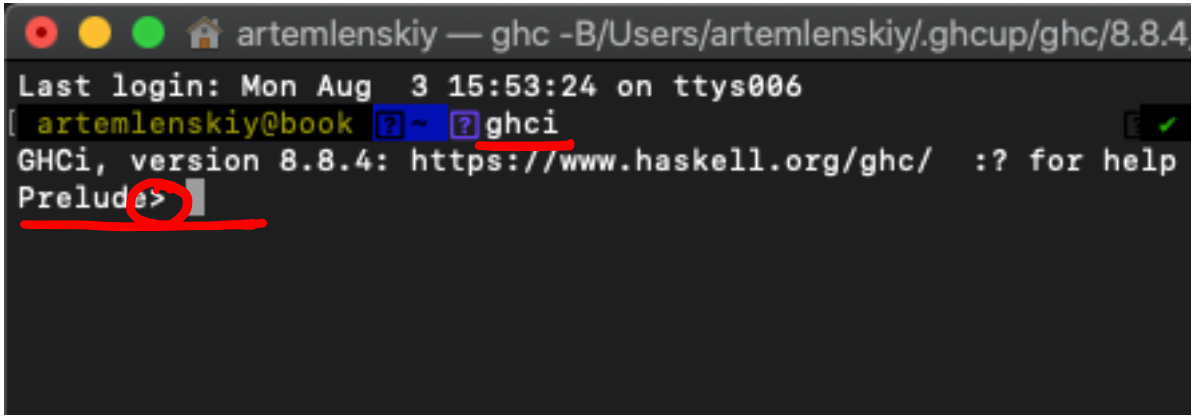
COMP 1100

Acknowledgement of Country

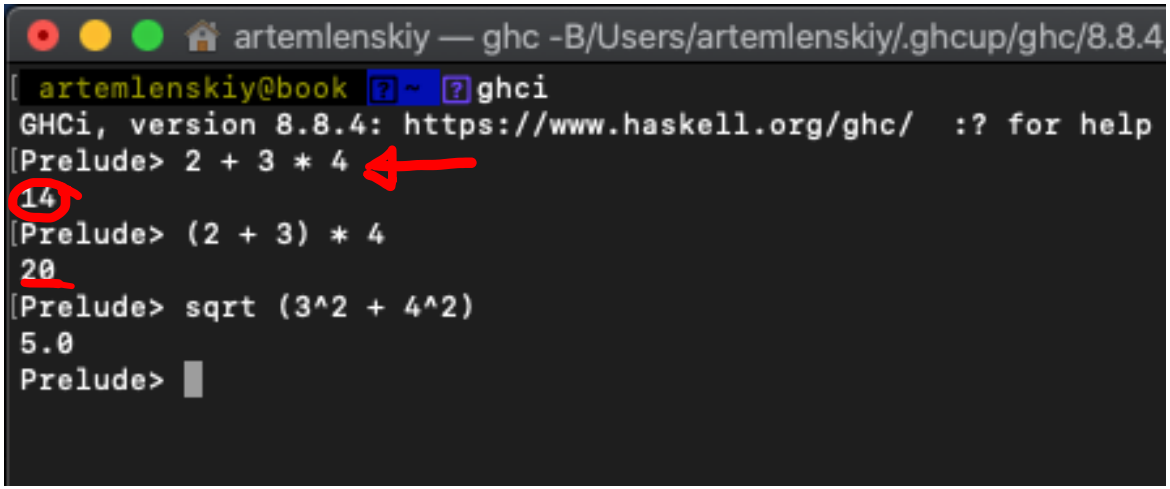
- ✓ *I wish to acknowledge the traditional custodians of the land we are meeting on, the Ngunnawal people. I wish to acknowledge and respect their continuing culture and the contribution they make to the life of this city and this region. I would also like to acknowledge and welcome any other Aboriginal and Torres Strait Islander people who are enrolled in our courses.*

Interactive GHCi system

- ✓ The interactive GHCI system can be started from the terminal

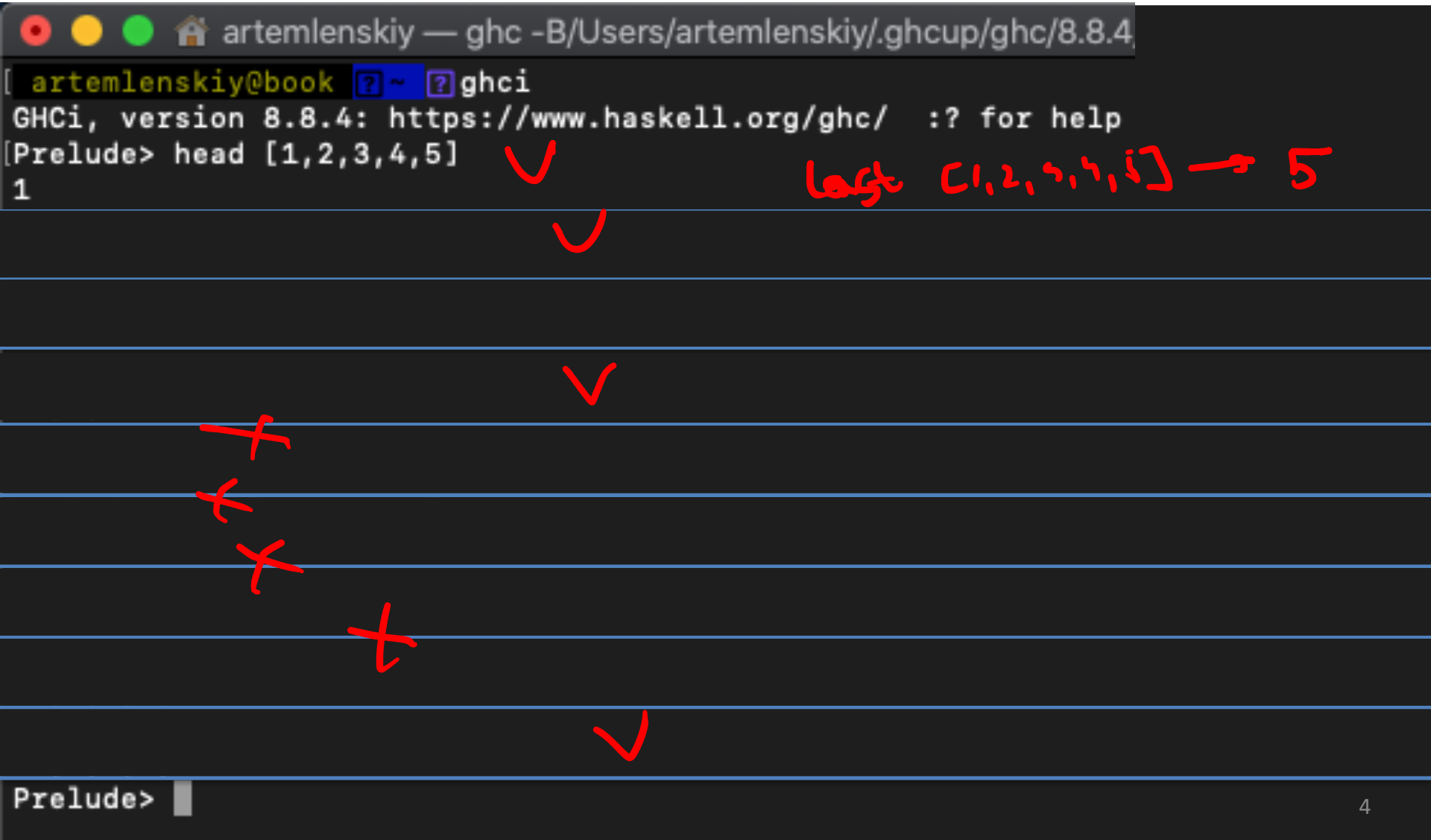


- ✓ The GHCI prompt > indicates that the system is now waiting for the user to enter an expression to be evaluated



Standard prelude

- ✓ Haskell comes with a large number of built-in functions



Function application

- ✓ Mathematical expressions vs Haskell expressions

$$f(a,b) + cd \iff \mathrm{f\ a\ b} + \mathrm{c} * \mathrm{d}$$

Function application in Haskell is denoted silently using spacing, while the multiplication is denoted using the operator *

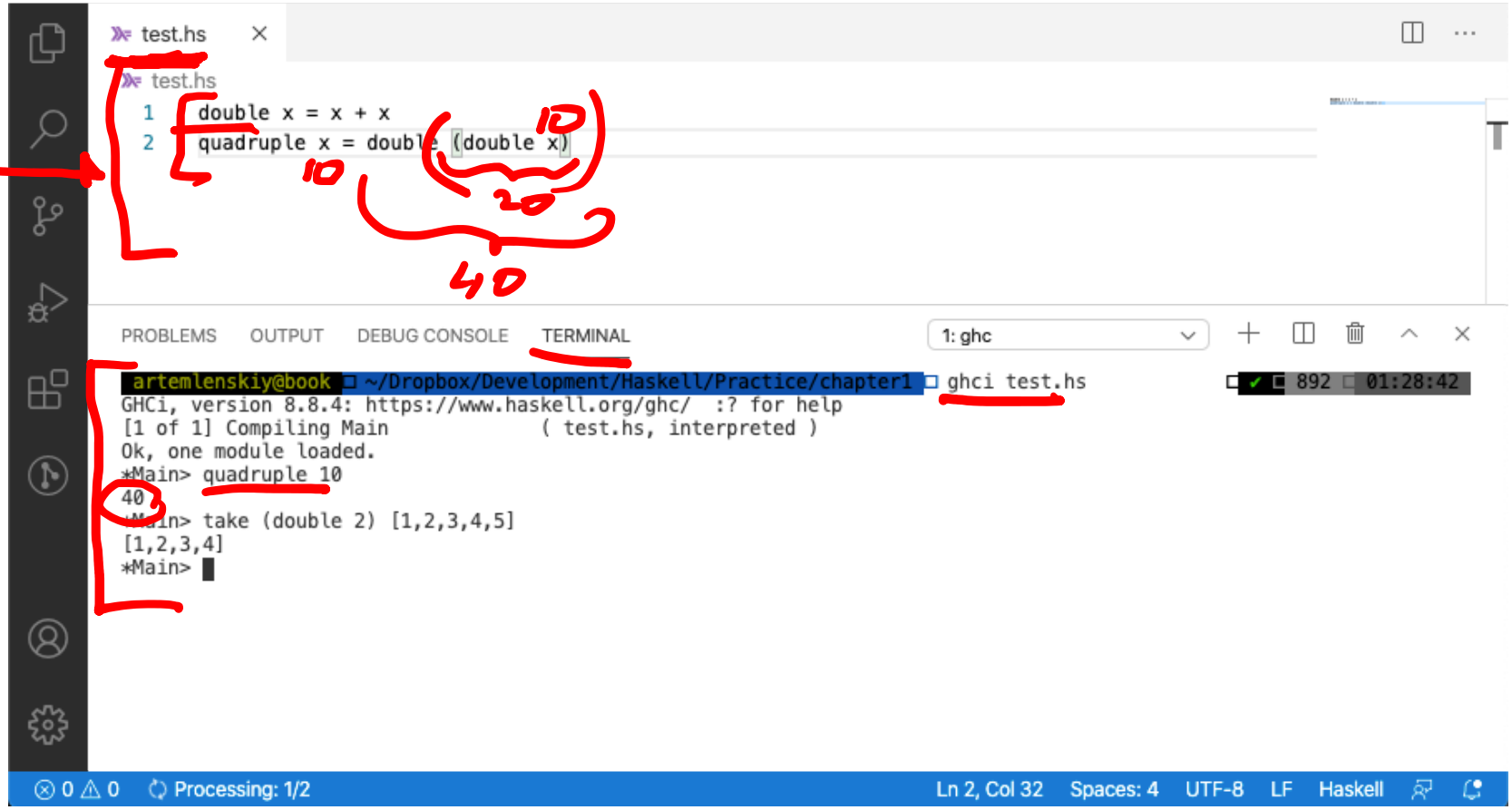
- ✓ More example

Mathematics	Haskell
$f(x)$	f x
$f(x,y)$	f x y
$f(g(x))$	f (g x)
$f(x, g(y))$	f x (g y)
$f(x)g(y)$	f x * g y

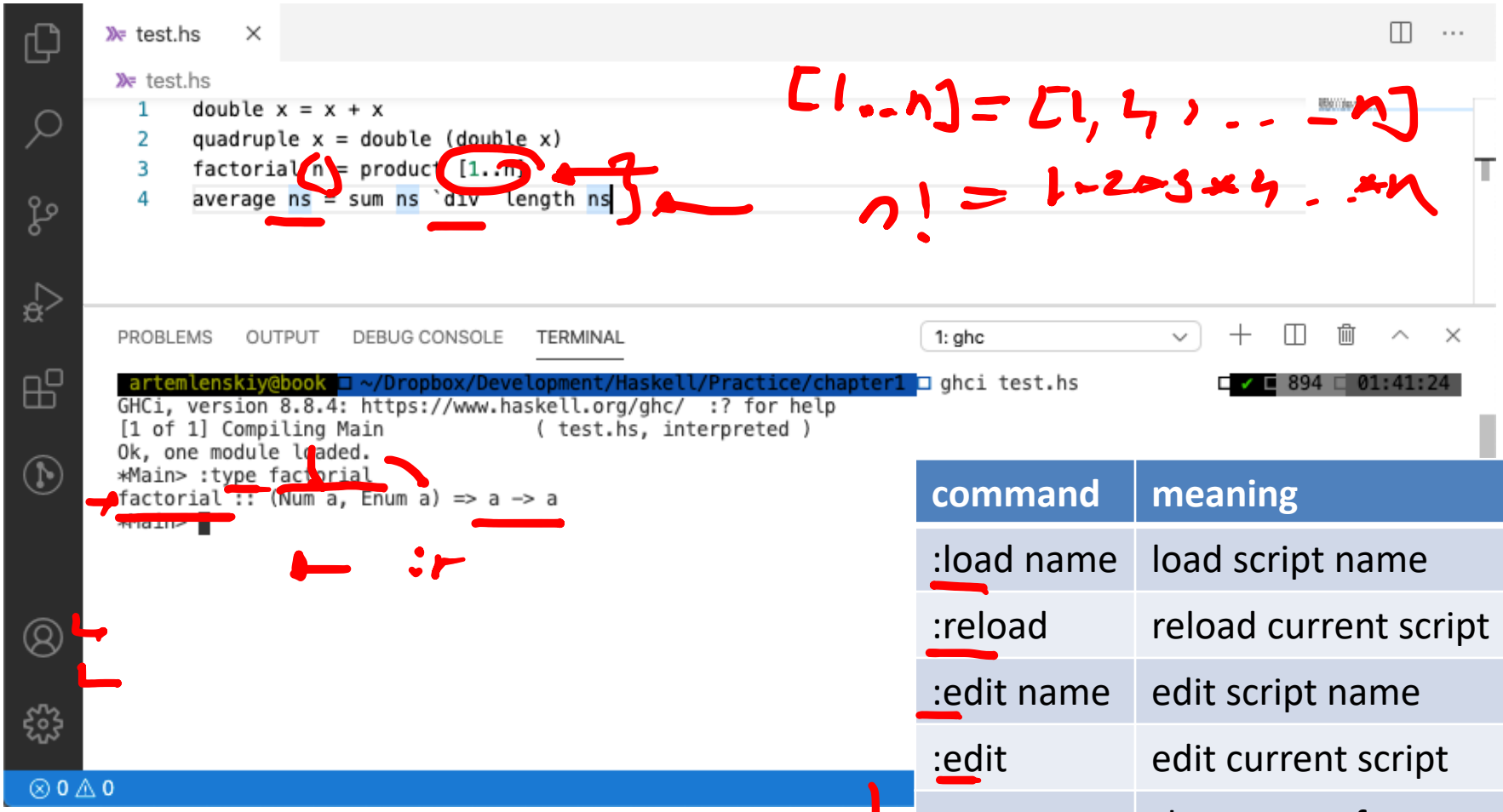
- ✓ Note that parentheses are still required e.g. f x (g y)

Haskell scripts: setup

- ✓ When running Haskell, keep two windows open:
  - ✓ An editor (e.g. VSCode)
  - ✓ GHCi



Haskell scripts: commands



command	meaning
:load name	load script name
:reload	reload current script
:edit name	edit script name
:edit	edit current script
:type expr	show type of expr
:?	Show all commands
:quit	quit GHCi

Haskell scripts: Naming requirements

- The names of the function and its arguments must *begin with a lower-case letter*, but can be followed by zero, more letters, digits, underscores and forward single quotes.
- **Examples**:

```
myFun      fun1      arg_2      x'
```

- The following list of keywords have a special meaning in the language

case	class	data	default	deriving
do	else	foreign	if	import
in	infix	infixl	infixr	instance
let	module	newtype	of	then
type	where			

- By convention, list arguments in Haskell usually have the suffix `s` on their name to indicate that they may contain multiple value.

Haskell scripts: the layout rule

- ✓ The layout rule makes it possible to determine the grouping of definitions from their indentation.
- ✓ **Example**:

```
a = b + c
    where
      b = 1
      c = 2
d = a * 2
```

It is clear that b and c are local definitions for use within the body of a.

- ✓ The grouping can be made explicitly using curly parentheses and semi-colon.

```
a = b + c
where{
b = 1;
c = 2;
d = a * 2
```

- ✓ Don't use tabs

Haskell scripts: comments

- Ordinary comments begin with the symbol – –

```
-- Factorial of a positive integer:
factorial n = product [1..n]
-- Average of a list of integers:
average ns = sum ns `div` length ns
```

- Nested comments begin and with the symbols {– and –} and may span multiple lines

```
{-
double x = x + x
quadruple x = double (double x)
-}
```

Exercise

- The script below contains three syntactic errors. Correct these errors and then check that your script works properly using GHCi

```
N = a “div” length xs
    where
        a = 10
       xs = [1, 2, 3, 4, 5]
```

- The library function last selects the last element of a non-empty list;
  - For example: `last [1, 2, 3, 4, 5] = 5`

  Show how the function last could be defined in terms of the other library functions.