Introduction to Functional Programming using Haskell

Errata

April 7, 1999

Chapter 1

page 3, line 1 Replace $?\ square\ 14198724$ by $?\ square\ 14197824$.

page 9, line 14 Replace "hit the the interrupt key" by "hit the interrupt key".

page 22, line 8 Insufficient space between names in *square square* 3. [This unfortunates space compression occurs in various places throughout the text.]

page 23, line 14 "Look again the previous . . . " should read "Look again at the previous . . . "

page 25, line 22 Replace by "The link between the two is the requirement that the implementation satisfies . . . "

Chapter 2

page 33, lines 1 - 8 The text is confused. There are two solutions:

Either replace it by "We can declare *Bool* to be an instance of *Ord* by writing

$$
\begin{array}{rcl}
\textbf{instance } Ord\ Bool\ \textbf{where} & & \\
False \leq False & = & True \\
False \leq True & = & True \\
True \leq False & = & False \\
True \leq True & = & True
\end{array}
$$

The alternative definition, namely $x \leq y = not\ x \vee y$, doesn't quite work in the way expected (see Exercise 2.1.2). "

Or replace lines 7 and 8 by "As an alternative definition we can write $x < y = not\ x \wedge y$.

page 33, line 20 "Note that the two occurrences of . . . "

page 34 Delete Exercise 2.1.2.

page 35 In Exercise 2.1.9 replace the last sentence by "Show that these properties hold for the definition of (`==`) on the well-defined values of *Bool*.

page 35, line -2 "a different entity from the decimal number 7;..."

page 37, line -4 "but does not depend ..."

pages 39 − 41 Systematically interchange the names *toEnum* and *fromEnum* in the whole of Section 2.3.

page 43, line 12 Interchange *toEnum* and *fromEnum*.

page 46, line -2 The type of *plus* should be

$$plus :: (\alpha \rightarrow \beta, \gamma \rightarrow \delta) \rightarrow Either\,\alpha\,\gamma \rightarrow Either\,\beta\,\delta$$

page 47 Note that Haskell uses the name *either* rather than *case*.

Chapter 3

page 73, line 14 Replace $h\,(foldn\,h\,b\,(Succ\,n))$ by $h\,(foldn\,h\,b\,n)$.

page 79, line 16 Replace by

$$Rat\,x\,y \mathtt{\,==\,} Rat\,u\,v \quad = \quad (x \times v) \mathtt{\,==\,} (y \times u)$$

page 81, lines 4–5 Replace sentence beginning "Among possible representations" with "Among possible representations we can choose one in which $-5 \le z < 5$ and *abs y* is as small as possible.

page 81, line 25 "since programs that avoid case analyses are clearer and simpler than those that do not, ..."

page 83, line 23 The definition of *done* should read

$$done\,(m, n) \quad = \quad (m + 1 \mathtt{\,==\,} n)$$

page 85, line 8 In the definition of y_3 a division by 2 is missing:

$$y_3 = (1.4167 + 2/1.4167)/2 = 1.4142157$$

Chapter 4

page 103, line 14 First line in definition of *init* should read: $init\,[x] = [\,]$.

page 112, line 10 "This equation is valid provided p and q are strict functions."

page 114, line 4 Definition of *pyth* should read

$$pyth\,(x, y, z) \quad = \quad (x \times x + y \times y \mathtt{==} z \times z)$$

page 116, line 14 Last line in definition of *zip* should read:

$$zip\,(x : xs)\,(y : ys) \quad = \quad (x, y) : zip\,xs\,ys$$

page 116, line 18 "the scalar product of two vectors x and y of size n is defined by ..."

page 121, line 22 The function *zip* can be defined as an instance of *foldr*: we have $zip = foldr\,f\,e$ where

$$
\begin{aligned}
e\,ys \quad &= \quad [\,] \\
f\,x\,g\,[\,] \quad &= \quad [\,] \\
f\,x\,g\,(y : ys) \quad &= \quad (x, y) : g\,ys
\end{aligned}
$$

page 124, line 12 "the first is clearer, while the second is more efficient."

page 124, line -6 The type of *scanl* should read:

$$scanl :: (\beta \to \alpha \to \beta) \to \beta \to [\alpha] \to [\beta]$$

page 125, line -1 Replace a by e in equations involving *scanr*.

page 126, lines 9,10 Replace a by e in equations involving *scanr*.

page 127 Exercise 4.5.9 should read "What list does $scanl\,(/)\,1\,[1..n]$ produce?"

page 125 In Exercise 4.5.11 the type of *convert* should be $Liste\,\alpha \to [\alpha]$.

page 129, line -4 "Both sides simplify to $x \oplus y$."

page 129, line -1 Replace $z; xs$ by $z : xs$.

page 130, line -7 Missing period at end of paragraph.

page 137 In Exercise 4.6.10 the law should read

$$foldl1\,(\oplus) \cdot scanl\,(\otimes)\,e \quad = \quad fst \cdot foldl\,(\odot)\,(e, e)$$

Chapter 5

page 146, line 9 Replace *assign xs* by *assign*.

page 146, line 12 The definition of *mktriple* should read

$$mktriple\,(xn, xm)\,xr \;\;=\;\; (xn, xm, xr)$$

page 148, line 18 The last line of the definition of *sortby* should read:

$$sortby\,f\,(x : y : xs)$$
$$=\;\; mergeby\,f\,(cross\,(sortby\,f, sortby\,f)\,(divide\,(x : y : xs)))$$

page 163, line 12 The three = signs on the right-hand side of the definition of *leap* should be `==` signs.

page 165, lines 12,15 The types of *stackWith* and *spreadWith* should be

$$stackWith \quad :: \quad Height \rightarrow [Picture] \rightarrow Picture$$
$$spreadWith \quad :: \quad Width \rightarrow [Picture] \rightarrow Picture$$

page 167, line 16 Replace *entries* (d, s) with just *entries*.

page 167, line 23 In the definition of *dnames* the conversion to type *Picture* is omitted, so prefix the right-hand side with *row*.

Chapter 6

page 185, lines -6 − -1 Replace definition of *fork* by

$$fork \qquad :: \quad Atree\,\alpha \rightarrow Atree\,\alpha \rightarrow Atree\,\alpha$$
$$fork\,xt\,yt \;\;=\;\; Fork\,(lsize\,xt)\,xt\,yt$$

$$lsize \qquad\qquad :: \quad Atree\,\alpha \rightarrow Int$$
$$lsize\,(Leaf\;x) \qquad = \quad 1$$
$$lsize\,(Fork\,n\,xt\,yt) \quad = \quad n + lsizeyt$$

page 186, line 6 The two occurrences of *mkBtree* should be replaced by *mkAtree*.

page 187 In Exercise 6.1.3 the definition of *subtrees* should read

$$subtrees \qquad\qquad :: \quad Btree\,\alpha \rightarrow [Btree\,\alpha]$$
$$subtrees\,(Leaf\;x) \qquad = \quad [Leaf\;x]$$
$$subtrees\,(Fork\,xt\,yt) \quad = \quad [Fork\,xt\,yt] + subtrees\,xt + subtrees\,yt$$

page 188, line 9 The type of *member* should read

$$member :: Ord\,\alpha \Rightarrow \alpha \rightarrow Stree\,\alpha \rightarrow Bool$$

4

page 188, line 15 Space compressed in *member x xt*.

page 188, line 17 The type of *height* should read

$$height :: Ord\ \alpha \Rightarrow Stree\ \alpha \rightarrow Int$$

page 190, line -10 The identity should read:

$$xs + ys\ =\ xs + [head\ ys] + tail\ ys$$

page 191, line -1 Exercise 6.2.4 should read "Prove that *inordered* (*insert x xt*) = *True* for all finite binary search trees *xt*.

page 193, line -11 The type of *heapify* should read

$$heapify :: Ord\ \alpha \Rightarrow Htree\ \alpha \rightarrow Htree\ \alpha$$

page 193, line -6 The type of *sift* should read

$$sift :: Ord\ \alpha \Rightarrow \alpha \rightarrow Htree\ \alpha \rightarrow Htree\ \alpha \rightarrow Htree\ \alpha$$

page 196, lines 13 − 17 Omit the local definition of *maxlist*, and insert the following sentence in the text: "Recall that *maxlist* = *foldl1* (**max**).

page 197, line -5 The left-hand expression should read: $f(g(x, y), z, h(t))$.

page 200, line -6 The type of *combine* should read: *combine* :: $[[[\alpha]]] \rightarrow [[\alpha]]$.

page 201, line 8 Same correction as above.

page 205, line -3 The type declaration of *CodeTable* should read:

$$\textbf{type}\ CodeTable\ =\ [(Char, [Bit], Int)]$$

page 206, line 19 Replace local definition by **where** $(ys, zs) = span\ (== x)\ xs$.

Chapter 7

page 231, line 8 The type of *dfcat* should read:

$$dfcat\ ::\ [Rose\ \alpha] \rightarrow [\alpha] \rightarrow [\alpha]$$

page 236, line 8 The second line in the definition of *fills* should read:

$$fills\ (w : ws)\ =\ [us : vss \mid (us, vs) \leftarrow splits\ (w : ws); vss \leftarrow fills\ vs]$$

page 236, line -1 Space compression in *fill vs*.

Chapter 8

page 256, line 16 "The second implementation therefore has a different efficiency from the first,..."

page 256, line 21 The right-hand side of the second axiom for *back* should be

$$join\ x\ (back\ (join\ y\ xq))$$

page 272, lines 13,15 Remove closing parenthesis from right-hand expressions.

page 278, line 12 The type of *fork* should be

$$fork\quad ::\quad \alpha \rightarrow Htree\ \alpha \rightarrow Htree\ \alpha \rightarrow Htree\ \alpha$$

page 279, line 1 The type of *delMin* should be

$$delMin\quad ::\quad Ord\ \alpha \Rightarrow Htree\ \alpha \rightarrow Htree\ \alpha$$

page 279, line 3 The type of *union* should be

$$union\quad ::\quad Ord\ \alpha \Rightarrow Htree\ \alpha \rightarrow Htree\ \alpha \rightarrow Htree\ \alpha$$

page 280, line 6 The type of *mkBag* should be

$$mkBag\quad ::\quad Ord\ \alpha \Rightarrow [\alpha] \rightarrow Htree\ \alpha$$

page 280, line 8 The type of *mkTwo* should be

$$mkTwo\quad ::\quad Ord\ \alpha \Rightarrow Int \rightarrow [\alpha] \rightarrow (Htree\ \alpha, [\alpha])$$

page 284, line 9 Replace right-hand side of *otherwise* branch by

$$Fork\ n\ xt\ (update\ yt\ (k - m)\ x)$$

page 288, line 4 Replace *nullys* by *null ys*.

page 288, line 22 It should be pointed out that the definition of *abstr* is exactly the same as in the implementation of Section 8.1 since

$$ys + reverse\ xs \quad = \quad reverse\ (xs + reverse\ ys)$$

page 289, line 14 Replace *reverseys* by *reverse ys*.

page 290, line -1 Replace last line by

$$(3, 0, rot\ (rot\ [\,]\ [1]\ [\,])\ [3, 2]\ [\,],\ [\,])$$

page 291, line 2 Replace by

$$(3, 3, rot\ (rot\ [\,]\ [1]\ [\,])\ [3, 2]\ [\,],\ [6, 5, 4])$$

page 291, line 4 Replace by

$$(7, 0, rot\ (rot\ (rot\ [\,]\ [1]\ [\,])\ [3, 2]\ [\,])\ [7, 6, 5, 4]\ [\,],\ [\,])$$

page 291, line 6 Replace by

$$(7, 7, rot\ (rot\ (rot\ [\,]\ [1]\ [\,])\ [3, 2]\ [\,])\ [7, 6, 5, 4]\ [\,],\ [14, 13..8])$$

Chapter 9

page 296, line 17 "the computer determines the first four elements ..."

Chapter 10

page 330, line 13 Should add "where C may be empty".

page 342 In Exercise 10.2.2 the type definition should read

$$\textbf{newtype}\ Count\ \alpha\ =\ CNT\ (\alpha, Counter)$$

Chapter 11

page 365, line -10 The definition should read

$$p\ \textbf{orelse}\ q\ =\ MkP\ f$$
$$\textbf{where}\ \ f\ s\ =\ \textbf{if}\ null\ ps\ \textbf{then}\ apply\ q\ s\ \textbf{else}\ ps$$
$$\textbf{where}\ ps = apply\ p\ s$$

page 365, line -6 The operator **orelse** does not satisfy the distributive law of **plus**.

page 368, line -1 Replace $\triangleright$ by $\gg$.

page 373 In Exercise 11.4.1 add "for deterministic parsers p and q".

Chapter 12

page 384, line -11 Type of *notuple* should read

$$notuple :: Parser\ [Expr]$$

Appendix

page 411, line -1 Replace by $head\ (x : xs) = x$.