**LYAH – Book Outline With Website Links**

[**1. INTRODUCTION**](http://learnyouahaskell.com/introduction)

So, What’s Haskell?

What You Need to Dive In

Acknowledgments

[**2. STARTING OUT**](http://learnyouahaskell.com/starting-out)

Calling Functions

Baby’s First Functions

An Intro to Lists

Concatenation

Accessing List Elements

Lists Inside Lists

Comparing Lists

More List Operations

Texas Ranges

I’m a List Comprehension.

Tuples

Using Tuples.

Using Pairs

Finding the Right Triangle

[**3. TYPES AND TYPECLASSES – BELIEVE THE TYPE**](http://learnyouahaskell.com/types-and-typeclasses)

Explicit Type Declaration

Common Haskell Types

Type Variables

Type Classes 101

The Eq Type Class

The Ord Type Class

The Show Type Class

The Read Type Class

The Enum Type Class

The Bounded Type Class.

The Num Type Class

The Floating Type Class

The Integral Type Class

Some Final Notes on Type Classes

[**4. SYNTAX IN FUNCTIONS**](http://learnyouahaskell.com/syntax-in-functions)

Pattern Matching

Pattern Matching with Tuples

Pattern Matching with Lists and List Comprehensions.

As-patterns

Guards, Guards!

where?!

where’s Scope

Pattern Matching with where

Functions in where Blocks

let It Be

let in List Comprehensions

let in GHCi

case Expressions

[**5. HELLO RECURSION!**](http://learnyouahaskell.com/recursion)

Maximum Awesome

A Few More Recursive Functions

replicate

take

reverse.

repeat

zip

elem

Quick, Sort!

The Algorithm

The Code

Thinking Recursively

[**6. HIGHER-ORDER FUNCTIONS**](http://learnyouahaskell.com/higher-order-functions)

Curried Functions

Sections

Printing Functions

Some Higher-Orderism Is in Order

Implementing zipWith

Implementing flip

The Functional Programmer’s Toolbox

The map Function

The filter Function

More Examples of map and filter

Mapping Functions with Multiple Parameters

Lambdas

I Fold You So

Left Folds with foldl

Right Folds with foldr

The foldl and foldr1 Functions

Some Fold Examples

Another Way to Look at Folds

Folding Infinite Lists

Scans

Function Application with $

Function Composition

Function Composition with Multiple Parameters

Point-Free Style

[**7. MODULES**](http://learnyouahaskell.com/modules)

Importing Modules

Solving Problems with Module Functions

Counting Words

Needle in the Haystack

Caesar Cipher Salad

On Strict Left Folds

Let’s Find Some Cool Numbers

Mapping Keys to Values

Almost As Good: Association Lists

Enter Data.Map

Making Our Own Modules

A Geometry Module

Hierarchical Modules

[**8. MAKING OUR OWN TYPES AND TYPE CLASSES**](http://learnyouahaskell.com/making-our-own-types-and-typeclasses)

Defining a New Data Type

Shaping Up.

Improving Shape with the Point Data Type

Exporting Our Shapes in a Module

Record Syntax

Type Parameters

Should We Parameterize Our Car?

Vector von Doom

Derived Instances.

Equating People

Show Me How to Read

Order in the Court!

Any Day of the Week

Type Synonyms

Making Our Phonebook Prettier

Parameterizing Type Synonyms

Go Left, Then Right

Recursive Data Structures

Improving Our List

Let’s Plant a Tree

Type Classes 102

Inside the Eq Type Class

A Traffic Light Data Type

Subclassing

Parameterized Types As Instances of Type Classes

A Yes-No Type Class

The Functor Type Class.

Maybe As a Functor

Trees Are Functors, Too

Either a As a Functor

Kinds and Some Type-Foo

[**9. INPUT AND OUTPUT**](http://learnyouahaskell.com/input-and-output)

Separating the Pure from the Impure.

Hello, World!

Gluing I/O Actions Together

Using let Inside I/O Actions

Putting It in Reverse

Some Useful I/O Functions

putStr

putChar

print

when

sequence

mapM

forever

forM

I/O Action Review

**(Book 9. MORE INPUT AND MORE OUTPUT)**

Files and Streams.

Input Redirection

Getting Strings from Input Streams

Transforming Input

Reading and Writing Files.

Using the withFile Function

It’s Bracket Time

Grab the Handles!

To-Do Lists

Deleting Items

Cleaning Up

Command-Line Arguments

More Fun with To-Do Lists

A Multitasking Task List

Dealing with Bad Input

Randomness

Tossing a Coin

More Random Functions

Randomness and I/O

Bytestrings

Strict and Lazy Bytestrings

Copying Files with Bytestrings

[**10. FUNCTIONALLY SOLVING PROBLEMS**](http://learnyouahaskell.com/functionally-solving-problems)

Reverse Polish Notation Calculator

Calculating RPN Expressions

Writing an RPN Function

Adding More Operators

Heathrow to London

Calculating the Quickest Path

Representing the Road System in Haskell

Writing the Optimal Path Function

Getting a Road System from the Input

[**11. APPLICATIVE FUNCTORS**](http://learnyouahaskell.com/functors-applicative-functors-and-monoids)

Functors Redux

I/O Actions As Functors

Functions As Functors

Functor Laws

Law 1

Law 2

Breaking the Law

Using Applicative Functors

Say Hello to Applicative

Maybe the Applicative Functor

The Applicative Style

Lists

IO Is An Applicative Functor, Too

Functions As Applicatives

Zip Lists

Applicative Laws

Useful Functions for Applicatives

**(Book 12. MONOIDS)**

Wrapping an Existing Type into a New Type

Using newtype to Make Type Class Instances

On newtype Laziness

type vs. newtype vs. data

About Those Monoids.

The Monoid Type Class

The Monoid Laws

Meet Some Monoids.

Lists Are Monoids

Product and Sum.

Any and All

The Ordering Monoid

Maybe the Monoid

Folding with Monoids

[**12. A FISTFUL OF MONADS**](http://learnyouahaskell.com/a-fistful-of-monads)

Upgrading Our Applicative Functors

Getting Your Feet Wet with Maybe

The Monad Type Class

Walk the Line

Code, Code, Code

I’ll Fly Away

Banana on a Wire

do Notation

Do As I Do

Pierre Returns

Pattern Matching and Failure

The List Monad

do Notation and List Comprehensions.

MonadPlus and the guard Function

A Knight’s Quest

Monad Laws

Left Identity

Right Identity

Associativity

[**13. FOR A FEW MONADS MORE**](http://learnyouahaskell.com/for-a-few-monads-more)

Writer? I Hardly Knew Her!

Monoids to the Rescue

The Writer Type

Using do Notation with Writer

Adding Logging to Programs

Inefficient List Construction

Using Difference Lists

Comparing Performance

Reader? Ugh, Not This Joke Again

Functions As Monads.

The Reader Monad

Tasteful Stateful Computations

Stateful Computations

Stacks and Stones.

The State Monad

Getting and Setting State

Randomness and the State Monad

Error Error on the Wall.

Some Useful Monadic Functions

liftM and Friends

The join Function

filterM

foldM

Making a Safe RPN Calculator

Composing Monadic Functions

Making Monads

[**14. ZIPPERS**](http://learnyouahaskell.com/zippers)

Taking a Walk

A Trail of Breadcrumbs

Going Back Up

Manipulating Trees Under Focus

Going Straight to the Top, Where the Air Is Fresh and Clean!

Focusing on Lists

A Very Simple Filesystem

Making a Zipper for Our Filesystem

Manipulating a Filesystem

Watch Your Step