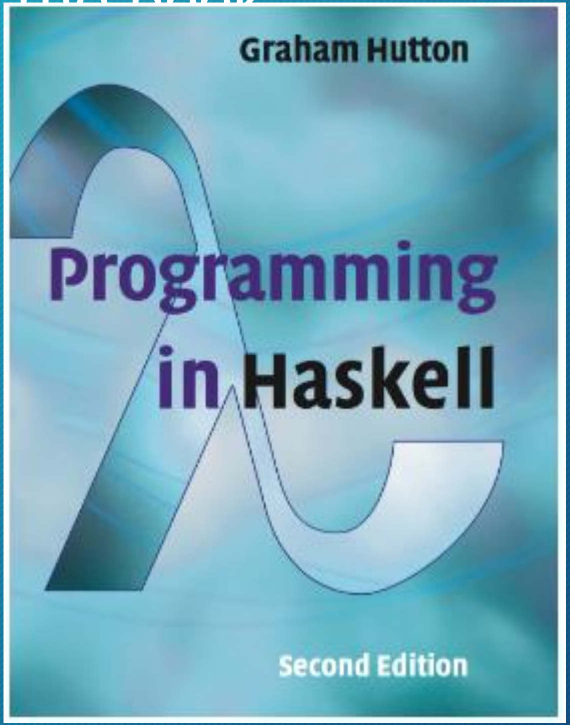
PROGRAMMING IN HASKELL

Chapter 1 - Introduction

Book Title

This course(including slides) is largely based on the book:

Programming in Haskell,
Graham Hutton, 2nd Ed,
Cambridge University Press,
ISBN 978-1-316-62622-1.



What is a Functional Programming?

- A programming style using functions to compute results.

- Emphasizes immutability, pure functions, and recursion.

Comparison

Summing the integers 1 to 10:

Java

```
int total = 0;
for (int i = 1; i ≤ 10; i++)
   total = total + i;
```

The computation method is variable assignment.

Haskell

```
sum [1..10]
```

The computation method is function application.

Key Characteristics of Functional Programming

- - **Immutability**: Variables don't change once set.
- - **Higher-order functions**: Functions as arguments or return values.
- - **Pure functions**: No side effects.
- - **Lazy evaluation**: Values computed only when needed.
- - Recursion: Functions calling themselves

Historical Evolution of Functional Programming

- 1930’s : Lambda calculus by Alonzo Church.
- 1950’s: Lisp by John McCarthy.
- 1960’s : ISWIM by Peter Landin.
- 1970’s : FP (Backus) and ML (Milner).
- 1980’s : Miranda (Turner).
- 1987 : Haskell development begins.
- 2003 : Haskell Report published.
- 2010+ : Modern Haskell advancements.

Why Haskell?

- **Strong type system** with type inference.
- **Lazy evaluation** for efficiency.
- **Expressive syntax** for clean code.
- Extensive library and tooling support.
- Promotes reasoning about programs.

Basic Syntax in Haskell

```
f []     = []
f (x:xs) = f ys ++ [x] ++ f zs
           where
              ys = [a | a ← xs, a ≤ x]
              zs = [b | b ← xs, b > x]
```

- Pattern matching
- List comprehensions
- Recursive definitions

Real world applications of Haskell

- Data analysis and processing.
- Blockchain (e.g., Cardano).
- Compilers and language tooling.
- Web development frameworks.