

УДК 004.432.42Haskell
ББК 32.973.28-018.1
Л61

А

Липовача, Миран.

Л61 Изучай Haskell во имя добра! / М. Липовача ; пер. с англ. Д. Леушина, А. Синицына, Я. Арсанукаева. — 2-е изд., эл. — 1 файл pdf : 492 с. — Москва : ДМК Пресс, 2023. — Систем. требования: Adobe Reader XI либо Adobe Digital Editions 4.5 ; экран 10". — Текст : электронный.

ISBN 978-5-89818-338-7

На взгляд автора, сущность программирования заключается в решении проблем. Программист всегда думает о проблеме и возможных решениях — либо пишет код для выражения этих решений.

Язык Haskell имеет множество впечатляющих возможностей, но главное его свойство в том, что меняется не только способ написания кода, но и сам способ размышления о проблемах и возможных решениях. Этим Haskell действительно отличается от большинства языков программирования. С его помощью мир можно представить и описать нестандартным образом. И поскольку Haskell предлагает совершенно новые способы размышления о проблемах, изучение этого языка может изменить и стиль программирования на всех прочих.

Еще одно необычное свойство Haskell состоит в том, что в этом языке придется особое значение рассуждениям о типах данных. Как следствие, вы помещаете больше внимания и меньше кода в ваши программы.

Вне зависимости от того, в каком направлении вы намерены двигаться, путешествуя в мире программирования, небольшой заход в страну Haskell себя оправдывает. А если вы решите там остаться, то наверняка найдете чем заняться и чему поучиться!

Эта книга поможет многим читателям найти свой путь к Haskell.

УДК 004.432.42Haskell
ББК 32.973.28-018.1

Электронное издание на основе печатного издания: Изучай Haskell во имя добра! / М. Липовача ; пер. с англ. Д. Леушина, А. Синицына, Я. Арсанукаева. — Москва : ДМК Пресс, 2014. — 492 с. — ISBN 978-5-97060-025-2. — Текст : непосредственный.

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

В соответствии со ст. 1299 и 1301 ГК РФ при устранении ограничений, установленных техническими средствами защиты авторских прав, правообладатель вправе требовать от нарушителя возмещения убытков или выплаты компенсации.

ISBN 978-5-89818-338-7

© Miran Lipovaca, No Starch Press, Inc.
© Оформление, перевод на русский язык,
ДМК Пресс, 2014

А

Оглавление

От издателя	12
Предисловие	13
Введение	16
1. На старт, внимание, марш!	21
Вызов функций.....	24
Функции: первые шаги	25
Списки	26
Конкатенация	29
Обращение к элементам списка	30
Списки списков	31
Сравнение списков.....	31
Другие операции над списками	32
Интервалы	35
Генераторы списков.....	37
Кортежи	41
Использование кортежей	42
Использование пар.....	43
В поисках прямоугольного треугольника	44
2. Типы и классы типов	46
Поверь в типы	46
Явное определение типов	47
Обычные типы в языке Haskell	48
Типовые переменные	50
Классы типов	51
Класс Eq	52
Класс Ord	53
Класс Show	54
Класс Read	54
Класс Enum	56
Класс Bounded	56
Класс Num	57

Класс Floating	58
Класс Integral	58
Несколько заключительных слов о классах типов	59
3. Синтаксис функций	60
Сопоставление с образцом	60
Сопоставление с парами	62
Сопоставление со списками и генераторы списков	63
Именованные образцы	66
Эй, стража!	67
Где же ты, where?!	69
Область видимости декларации where	71
Сопоставление с образцами в секции where	72
Функции в блоке where	72
Пусть будет let	73
Выражения let в генераторах списков	75
Выражения let в GHCi	75
Выражения для выбора из вариантов	76
4. Рекурсия	78
Привет, рекурсия!	78
Максимум удобства	79
Ещё немного рекурсивных функций	81
Функция replicate	81
Функция take	81
Функция reverse	82
Функция repeat	83
Функция zip	83
Функция elem	84
Сортируем, быстро!	84
Алгоритм	85
Определение	86
Думаем рекурсивно	87
5. Функции высшего порядка	89
Каррированные функции	89
Сечения	92
Печать функций	93
Немного о высоких материях	94
Реализация функции zipWith	95
Реализация функции flip	96
Инструментарий функционального программиста	98
Функция map	98
Функция filter	99
Ещё немного примеров использования map и filter	100
Функция map для функций нескольких переменных	103

Лямбда-выражения	104
Я вас сверну!	106
Левая свёртка foldl	107
Правая свёртка foldr	108
Функции foldl1 и foldr1	110
Примеры свёрток	111
Иной взгляд на свёртки	112
Свёртка бесконечных списков	113
Сканирование	114
Применение функций с помощью оператора \$	115
Композиция функций	117
Композиция функций с несколькими параметрами	118
Бесточечная нотация	120
6. Модули	122
Импорт модулей	123
Решение задач средствами стандартных модулей	125
Подсчёт слов	125
Иголка в стоге сена	127
Салат из шифра Цезаря	129
О строгих левых свёртках	130
Поищем числа	132
Отображение ключей на значения	135
Почти хорошо: ассоциативные списки	135
Модуль Data.Map	137
Написание собственных модулей	142
Модуль Geometry	143
Иерархия модулей	145
7. Создание новых типов и классов типов	148
Введение в алгебраические типы данных	148
Отличная фигура за 15 минут	149
Верный способ улучшить фигуру	151
Фигуры на экспорт	153
Синтаксис записи с именованными полями	154
Параметры типа	157
Параметризовать ли машины?	160
Векторы судьбы	162
Производные экземпляры	163
Сравнение людей на равенство	164
Покажи мне, как читать	165
Порядок в суде!	167
Любой день недели	168
Синонимы типов	170
Улучшенная телефонная книга	170

Параметризация синонимов	172
Иди налево, потом направо	173
Рекурсивные структуры данных	176
Улучшение нашего списка	177
Вырастим-ка дерево	179
Классы типов, второй семестр	183
«Внутренности» класса Eq	183
Тип для представления светофора	184
Наследование классов	186
Создание экземпляров классов для параметризованных типов	187
Класс типов «да–нет»	190
Класс типов Functor	193
Экземпляр класса Functor для типа Maybe	195
Деревья тоже являются функторами	196
И тип Either является функтором	197
Сорта и немного тип-фу	198
8. Ввод-вывод	204
Разделение «чистого» и «нечистого»	204
Привет, мир!	205
Объединение действий ввода-вывода	207
Использование ключевого слова let внутри блока do	211
Обращение строк	212
Некоторые полезные функции для ввода-вывода	215
Функция putStr	215
Функция putChar	216
Функция print	217
Функция when	218
Функция sequence	218
Функция mapM	220
Функция forever	220
Функция forM	221
Обзор системы ввода-вывода	222
9. Больше ввода и вывода	223
Файлы и потоки	223
Перенаправление ввода	224
Получение строк из входного потока	225
Преобразование входного потока	228
Чтение и запись файлов	230
Использование функции withFile	233
Время заключать в скобки	234
Хватай дескрипторы!	235
Список дел	237
Удаление заданий	238

Уборка.....	240
Аргументы командной строки.....	241
Ещё больше шалостей со списком дел	243
Многозадачный список задач	244
Работаем с некорректным вводом.....	248
Случайность.....	249
Подбрасывание монет	252
Ещё немного функций, работающих со случайностью	254
Случайность и ввод-вывод.....	255
Bytestring: тот же String, но быстрее	259
Строгие и ленивые	261
Копирование файлов при помощи Bytestring	263
Исключения.....	265
Обработка исключений, возникших в чистом коде	267
Обработка исключений ввода-вывода	273
Вспомогательные функции для работы с исключениями.....	279
10. Решение задач в функциональном стиле	281
Вычисление выражений в обратной польской записи.....	281
Вычисление выражений в ОПЗ	282
Реализация функции вычисления выражений в ОПЗ	283
Добавление новых операторов	286
Из аэропорта в центр	287
Вычисление кратчайшего пути.....	289
Представление пути на языке Haskell.....	291
Реализация функции поиска оптимального пути.....	293
Получение описания дорожной системы из внешнего источника..	296
11. Аппликативные функторы	299
Функторы возвращаются.....	300
Действия ввода-вывода в качестве функторов	301
Функции в качестве функторов	304
Законы функторов	308
Закон 1	308
Закон 2	309
Нарушение закона	310
Использование аппликативных функторов	313
Поприветствуйте аппликативные функторы.....	315
Аппликативный функтор Maybe	316
Аппликативный стиль	318
Списки.....	320
Тип IO – тоже аппликативный функтор	323
Функции в качестве аппликативных функторов.....	325
Застёгиваемые списки	327
Аппликативные законы	329
Полезные функции для работы с аппликативными функторами...	329

12. Моноиды.....	336
Оборачивание существующего типа в новый тип.....	336
Использование ключевого слова <code>newtype</code> для создания экземпляров классов типов.....	339
О ленивости <code>newtype</code>	341
Ключевое слово <code>type</code> против <code>newtype</code> и <code>data</code>	344
В общих чертах о моноидах.....	346
Класс типов <code>Monoid</code>	348
Законы моноидов	349
Познакомьтесь с некоторыми моноидами.....	350
Списки являются моноидами.....	350
Типы <code>Product</code> и <code>Sum</code>	352
Типы <code>Any</code> и <code>All</code>	354
Моноид <code>Ordering</code>	355
Моноид <code>Maybe</code>	359
Свёртка на моноидах.....	361
13. Пригоршня монад.....	367
Совершенствуем наши аппликативные функторы	367
Приступаем к типу <code>Maybe</code>	369
Класс типов <code>Monad</code>	373
Прогулка по канату.....	376
Код, код, код.....	377
Я улечу	379
Банан на канате	382
Нотация <code>do</code>	385
Делай как я	387
Пьер возвращается	388
Сопоставление с образцом и неудача в вычислениях	390
Списковая монада.....	392
Нотация <code>do</code> и генераторы списков	396
Класс <code>MonadPlus</code> и функция <code>guard</code>	396
Ход конём.....	399
Законы монад	402
Левая единица.....	402
Правая единица.....	404
Ассоциативность.....	405
14. Ещё немного монад	408
Writer? Я о ней почти не знаю!	409
Моноиды приходят на помощь.....	412
Тип <code>Writer</code>	414
Использование нотации <code>do</code> с типом <code>Writer</code>	416
Добавление в программы функции журналирования	417
Добавление журналирования в программы	418

Неэффективное создание списков	420
Разностные списки	422
Сравнение производительности	424
Монада Reader? Тьфу, опять эти шуточки!	425
Функции в качестве монад	426
Монада Reader	427
Вкусные вычисления с состоянием	429
Вычисления с состоянием	430
Стеки и чебуреки	431
Монада State	433
Получение и установка состояния	437
Случайность и монада State	438
Свет мой, Error, скажи, да всю правду доложи	439
Некоторые полезные монадические функции	442
liftM и компания	442
Функция join	446
Функция filterM	449
Функция foldM	452
Создание безопасного калькулятора выражений в обратной польской записи	454
Композиция монадических функций	457
Создание монад	459
15. Застёжки	467
Прогулка	468
Тропинка из хлебных крошек	471
Движемся обратно вверх	473
Манипулируем деревьями в фокусе	476
Идем прямо на вершину, где воздух чист и свеж!	478
Фокусируемся на списках	478
Очень простая файловая система	480
Создаём застёжку для нашей файловой системы	482
Манипулируем файловой системой	485
Осторожнее – смотрите под ноги!	486
Благодарю за то, что прочитали!	489