



# Содержание

<b>От авторов перевода</b> .....	10
О книге .....	10
О переводе .....	10
<b>Введение</b> .....	12
Предисловие .....	12
Благодарности .....	14
<b>Глава 1. Введение</b> .....	15
<b>Глава 2. Язык и синтаксис</b> .....	19
2.1. Упражнения .....	24
<b>Глава 3. Регулярные языки</b> .....	27
3.1. Упражнение .....	32
<b>Глава 4. Анализ контекстно-свободных языков</b> .....	33
4.1. Метод рекурсивного спуска .....	34
4.2. Таблично-управляемый нисходящий синтаксический анализ .....	38
4.3. Восходящий синтаксический анализ .....	40
4.4. Упражнения .....	42
<b>Глава 5. Атрибутные грамматики и семантики</b> ....	45
5.1. Правила типов .....	46
5.2. Правила вычислений .....	47
5.3. Правила трансляции .....	48
5.4. Упражнение .....	49
<b>Глава 6. Язык программирования Оберон-0</b> .....	51
6.1. Упражнение .....	54

---

<b>Глава 7. Синтаксический анализатор для Оберона-0</b> .....	55
7.1. Лексический анализатор .....	56
7.2. Синтаксический анализатор .....	57
7.3. Устранение синтаксических ошибок .....	59
7.4. Упражнения .....	64
<b>Глава 8. Учет контекста, заданного объявлениями</b> .....	65
8.1. Объявления .....	66
8.2. Записи о типах данных .....	68
8.3. Представление данных во время выполнения .....	69
8.4. Упражнения .....	73
<b>Глава 9. RISC-архитектура как цель</b> .....	75
9.1. Ресурсы и регистры .....	76
<b>Глава 10. Выражения и присваивания</b> .....	81
10.1. Прямая генерация кода по принципу стека .....	82
10.2. Отсроченная генерация кода .....	84
10.3. Индексированные переменные и поля записей .....	89
10.4. Упражнения .....	94
<b>Глава 11. Условные и циклические операторы и логические выражения</b> .....	95
11.1. Сравнения и переходы .....	96
11.2. Условные и циклические операторы .....	97
11.3. Логические операции .....	101
11.4. Присваивание логическим переменным .....	105
11.5. Упражнения .....	106
<b>Глава 12. Процедуры и концепция локализации</b> .....	109
12.1. Организация памяти во время выполнения .....	110
12.2. Адресация переменных .....	112
12.3. Параметры .....	114
12.4. Объявления и вызовы процедур .....	116

---

12.5. Стандартные процедуры .....	121
12.6. Процедуры-функции .....	122
12.7. Упражнения .....	123
<b>Глава 13. Элементарные типы данных .....</b>	<b>125</b>
13.1. Типы REAL и LONGREAL .....	126
13.2. Совместимость между числовыми типами данных .....	127
13.3. Тип данных SET .....	129
13.4. Упражнения .....	130
<b>Глава 14. Открытые массивы, указательный и процедурный типы .....</b>	<b>131</b>
14.1. Открытые массивы .....	132
14.2. Динамические структуры данных и указатели .....	133
14.3. Процедурные типы .....	136
14.4. Упражнения .....	138
<b>Глава 15. Модули и раздельная компиляция .....</b>	<b>141</b>
15.1. Принцип скрытия информации .....	142
15.2. Раздельная компиляция .....	143
15.3. Реализация символьных файлов .....	145
15.4. Адресация внешних объектов .....	149
15.5. Проверка конфигурационной совместимости .....	150
15.6. Упражнения .....	152
<b>Глава 16. Оптимизация и структура пре/постпроцессора .....</b>	<b>153</b>
16.1. Общие соображения .....	154
16.2. Простые оптимизации .....	155
16.3. Исключение повторных вычислений .....	156
16.4. Распределение регистров .....	157
16.5. Структура пре/постпроцессорного компилятора .....	158
16.6. Упражнения .....	162
<b>Приложение А. Синтаксис .....</b>	<b>164</b>
А1. Оберон-0 .....	164
А2. Оберон .....	164
А3. Символьные файлы .....	166

---

<b>Приложение В. Набор символов ASCII .....</b>	<b>167</b>
<b>Приложение С. Компилятор Оберон-0 .....</b>	<b>168</b>
С.1. Лексический анализатор .....	169
С.2. Синтаксический анализатор .....	172
С.3. Генератор кода .....	182
<b>Литература .....</b>	<b>191</b>

# От авторов перевода

## О книге

Давно известно, что лучший способ постичь секреты мастерства – это наблюдать за работой мастера. Эта небольшая, но насыщенная информацией книжка, по сути дела, представляет собой отчет о такой работе. Ну а то, что ее автор – настоящий мастер своего дела, сомнению не подлежит, потому что имя профессора Никлауса Вирта ни в каких дополнительных рекомендациях не нуждается. Эта книга – своего рода мастер-класс, который дает своим ученикам всемирно известный маэстро. Она не является ни «тяжелой» теоретической монографией, ни сборником наставлений и поучений увенчанного лаврами мэтра. Эта книжка – *практическое* пособие для всех тех любознательных людей, кто желает разобраться и понять, что такое компилятор и как он устроен. По мнению автора, без этого ни один программист не может называть себя квалифицированным специалистом.

В отличие от многочисленных книг, которые исчерпывающе описывают и теорию, и разнообразные методы синтаксического анализа, перевода и компиляции, эта книжка посвящена реализации одного-единственного компилятора *современного* языка программирования для *конкретного* компьютера. Но это нисколько не умаляет ее достоинства. Если обычные книги после прочтения почти всегда оставляют читателя наедине с вопросом «А что же дальше? Где же результат?» или с загадочными, полными опечаток текстами готовых программ, то эта небольшая книжка расставляет практически все точки над *i*, проводя читателя от *самого начала* до *самого конца* процесса разработки компилятора, попутно предупреждая его о неверных шагах и давая ему в руки богатый практический материал. Автор придерживается принципа «Делай со мной. Делай, как я. Делай лучше меня».

Таким образом, книга Н. Вирта – безусловно, не только прекрасное дополнение к многочисленным и столь же прекрасным фундаментальным трудам по этой теме, но может и должна использоваться в качестве практического пособия по изучению компиляторов. Кроме того, простота и доступность преподнесения довольно сложного материала снимает с него покров таинственности и делает его доступным практически каждому любителю программирования. Остается только сожалеть о том, что эта книга не была своевременно переведена и издана у нас.

Для практического использования текст компилятора Оберон-0, о котором идет речь в книге, адаптирован к системе БлэкБокс (BlackBox Component Builder – вариант системы Оберон). Оригинальные и адаптированные исходные тексты компилятора можно найти на сайте [www.oberoncore.ru](http://www.oberoncore.ru).

## О переводе

Несколько слов о переводе.

В силу того, что мы имеем дело не с развернутой монографией, а с конспектом лекций, каждая фраза, часто облекаемая в форму тезиса, до предела насыщена

информацией. Поэтому наша основная задача при переводе состояла в том, чтобы сохранить лаконичность и информационную насыщенность авторского текста и при этом максимально точно довести его суть до читателя, не поддаваясь искушению сдобрить его отсебятиной.

Несмотря на царящие до сих пор «разброд и шатания» в терминологии по этой теме, мы при переводе, следуя за автором, отдавали предпочтение наиболее устоявшимся, хотя и не всегда правильным и точным, терминам.

В связи с этим нельзя не упомянуть о терминах «front-end» и «back-end». Они уже давно употребляются в разнообразной англоязычной технической литературе, но тем не менее до сих пор не находят адекватных русскоязычных эквивалентов. Чаще всего их перевод зависит от контекста. Применительно к компиляторам наиболее точными их русскими аналогами являются, пожалуй, «машинно-независимая часть» и «машинно-зависимая часть» соответственно. Однако мы, теперь уже следуя авторской лаконичности, предпочли им более абстрактные и менее точные, но более короткие термины – «препроцессор» и «постпроцессор» соответственно.

Кроме того, список литературы пронумерован, и именные ссылки на него в тексте заменены номерными. К списку литературы добавлено несколько более поздних публикаций.

Авторы перевода выражают благодарность В. Н. Лукину за прочтение перевода и сделанные замечания.

# Введение

## Предисловие

Эта книга появилась из моих конспектов лекций по вводному курсу проектирования компиляторов в ЕТН (Федеральном технологическом институте) в Цюрихе. Несколько раз меня просили объяснить необходимость этого курса, так как проектирование компиляторов рассматривается как некий эзотерический предмет, применяемый только в нескольких узкоспециализированных программистских фирмах. Поскольку в наши дни все, что не приносит немедленной выгоды, должно быть оправдано, я должен попробовать объяснить, почему я вообще считаю этот предмет важным и уместным для студентов, изучающих информатику.

Основой любого академического образования является то, что передается не только знание и, в случае инженерного образования, «ноу-хау», но и понимание сути явления и способность проникнуть в его суть. В частности, в информатике мало поверхностного знания системы, необходимо еще и понимание ее содержания. Каждый образованный программист должен знать возможности компьютера, понимать способы и методы представления и интерпретации программ. Компилятор преобразует текст программы во внутренний код, это мост, соединяющий программное обеспечение и аппаратные средства.

Однако кому-то может показаться, что нет необходимости знать методы трансляции для понимания связи между исполняемой программой и кодом и еще менее важно знать, как на самом деле пишется компилятор. Личный опыт преподавателя подсказывает мне, что глубокое понимание предмета лучше всего приходит при всестороннем проникновении как в общую идею системы, так и в детали ее реализации. В нашем случае таким проникновением будет написание реального компилятора.

Конечно, мы должны сосредоточиться на основах. В конце концов, эта книга – вводный курс, а не справочник для специалистов. Наше первое ограничение касается входного языка. Было бы неуместным рассматривать проектирование компиляторов для больших языков. Язык должен быть небольшим, но тем не менее должен содержать все поистине фундаментальные элементы языков программирования. Для наших целей мы выбрали подмножество языка Оберон. Второе ограничение касается целевого компьютера. Он должен иметь обычную структуру и простой набор команд. Наиболее важна практичность обучающих понятий. Оберон – это общецелевой гибкий и мощный язык, а наш целевой компьютер идеальным образом отражает удачную RISC-архитектуру. И наконец, третье ограничение состоит в отказе от изощренных методов оптимизации кода. При таких условиях можно объяснить весь компилятор в деталях и даже создать его в ограниченные рамки курса сроки.

В главах 2 и 3 рассматриваются основы языка и синтаксиса. Глава 4 посвящена синтаксическому анализу, то есть методу разбора предложений и программ. Мы

сосредоточили внимание на простом, но удивительно мощном методе рекурсивного спуска, который используется в нашем иллюстративном компиляторе. Мы рассматриваем синтаксический анализ как средство для достижения цели, но не как самоцель. Глава 5 готовит нас к переходу от синтаксического анализатора к компилятору, а выбранный метод ставится в зависимость от атрибутов синтаксических конструкций.

После знакомства в главе 6 с языком Оберон-0 в главе 7 приводится разработка его синтаксического анализатора методом рекурсивного спуска. Из практических соображений обсуждается также обработка синтаксических ошибок. В главе 8 мы объясняем, почему языки, содержащие объявления и, следовательно, зависимость от контекста, могут тем не менее обрабатываться как контекстно-свободные.

До этого момента не было необходимости в рассмотрении целевого компьютера и набора его команд. Но поскольку последующие главы посвящены теме генерации кода, то становится неизбежной спецификация целевого компьютера (глава 9). Это RISC-архитектура с небольшим набором команд и набором регистров. В связи с этим центральная тема разработки компилятора – генерация последовательностей команд – разнесена по трем главам: код для выражений и присваиваний (глава 10), код для условных операторов и операторов цикла (глава 11), код для объявлений процедур и обращений к ним (глава 12). Вместе они покрывают все конструкции языка Оберон-0.

Последующие главы посвящены нескольким дополнительным важным конструкциям языков программирования общего назначения. Их трактовка поверхностна и не затрагивает деталей, но подкреплена несколькими упражнениями в конце соответствующих глав. Рассматриваются следующие темы: элементарные типы данных (глава 13), открытые массивы, динамические структуры данных и процедурные типы, называемые методами в объектно-ориентированной терминологии (глава 14).

Глава 15 касается модульного конструирования и принципов скрытия информации. Это приводит к теме разработки программного обеспечения в команде, основанной на определении интерфейсов и последующей независимой реализации частей (модулей). Методика основана на раздельной компиляции модулей с полным контролем совместимости типов всех компонентов интерфейса. Такая методика имеет первостепенное значение для разработки программного обеспечения в целом и для современных языков программирования в частности.

Наконец, глава 16 дает краткий обзор проблем оптимизации кода. Она необходима, с одной стороны, из-за семантической пропасти между исходными языками и архитектурами компьютеров, а с другой – из-за нашего желания как можно полнее использовать все доступные ресурсы компьютеров.