

# Оглавление

<b>Об авторе.....</b>	<b>24</b>
<b>О научном редакторе.....</b>	<b>25</b>
<b>Благодарности .....</b>	<b>26</b>
К первому изданию .....	26
Ко второму изданию.....	27
<b>Введение .....</b>	<b>28</b>
Зачем нужна командная строка?.....	29
О чем эта книга .....	29
Кому адресована эта книга.....	30
Что дается в этой книге.....	31
Как читать эту книгу.....	31
Предварительные условия .....	32
Что нового во втором издании.....	33
Ваши отзывы важны для нас! .....	33
От издательства .....	34
<b>Часть I. КОМАНДНАЯ ОБОЛОЧКА .....</b>	<b>35</b>
<b>Глава 1. Что такое командная оболочка.....</b>	<b>36</b>
Эмуляторы терминалов.....	36
Первые удары по клавишам .....	36
История команд.....	38
Управление курсором.....	38
Некоторые простые команды.....	38
Завершение сеанса работы с терминалом.....	39
Заключение .....	40

<b>Глава 2. Навигация .....</b>	<b>41</b>
Дерево каталогов файловой системы .....	41
Текущий рабочий каталог .....	42
Вывод содержимого каталога .....	43
Смена текущего рабочего каталога .....	43
Абсолютные пути .....	43
Относительные пути.....	44
Некоторые полезные сокращения .....	46
Заключение .....	46
<b>Глава 3. Исследование системы .....</b>	<b>47</b>
Любопытные возможности ls .....	47
Параметры и аргументы .....	48
Пристальный взгляд на длинный формат .....	49
Определение типов файлов командой file.....	50
Просмотр содержимого файлов командой less .....	51
Обзорное путешествие .....	53
Символические ссылки .....	57
Жесткие ссылки.....	58
Заключение .....	58
<b>Глава 4. Операции с файлами и каталогами.....</b>	<b>59</b>
Групповые символы.....	60
mkdir — создание каталогов .....	62
cp — копирование файлов и каталогов.....	63
Параметры команды cp и примеры ее использования.....	63
mv — перемещение и переименование файлов .....	64
Параметры команды mv и примеры ее использования.....	64
rm — удаление файлов и каталогов .....	65
Параметры команды rm и примеры ее использования .....	65
ln — создание ссылок .....	67
Жесткие ссылки .....	67
Символические ссылки .....	68

---

Постройка песочницы.....	68
Создание каталогов.....	68
Копирование файлов.....	69
Перемещение и переименование файлов.....	70
Создание жестких ссылок.....	71
Создание символических ссылок.....	72
Удаление файлов и каталогов.....	73
Заключение.....	75
<b>Глава 5. Работа с командами.....</b>	<b>76</b>
Что такое команды?.....	76
Идентификация команд.....	77
type — получение типа команды.....	77
which — определение местоположения выполняемого файла.....	77
Получение документации с описанием команд.....	78
help — получение справки для встроенных команд.....	78
--help — вывод инструкции по использованию.....	80
man — вывод страниц справочного руководства.....	80
apropos — вывод списка подходящих команд.....	82
whatis — вывод очень краткого описания команды.....	83
info — вывод записи из справочного руководства Info.....	83
README и другие файлы с описанием программ.....	85
Создание собственных команд с помощью alias.....	85
Заключение.....	87
<b>Глава 6. Перенаправление.....</b>	<b>88</b>
Стандартный ввод, вывод и вывод ошибок.....	88
Перенаправление стандартного вывода.....	89
Перенаправление стандартного вывода ошибок.....	91
Перенаправление стандартного вывода и стандартного вывода ошибок в один файл.....	91
Удаление нежелательного вывода.....	92
Перенаправление стандартного ввода.....	93
cat — объединение файлов.....	93



Конвейеры.....	95
Фильтры.....	95
uniq — поиск или удаление повторяющихся строк.....	96
wc — вывод числа строк, слов и байтов .....	97
grep — поиск строк, соответствующих шаблону .....	97
head/tail — вывод первых/последних строк из файлов .....	98
tee — чтение со стандартного ввода и запись в стандартный вывод и в файлы.....	99
Заключение.....	101
<b>Глава 7. Взгляд на мир глазами командной оболочки.....</b>	<b>102</b>
Подстановка.....	102
Подстановка путей.....	103
Подстановка тильды.....	104
Подстановка результатов арифметических выражений .....	105
Подстановка фигурных скобок .....	106
Подстановка параметров.....	107
Подстановка команд.....	108
Экранирование .....	109
Двойные кавычки.....	109
Одиночные кавычки .....	111
Экранирование символов .....	111
Управляющие последовательности.....	112
Заключение.....	113
<b>Глава 8. Продвинутое приемы работы с клавиатурой.....</b>	<b>114</b>
Редактирование командной строки.....	114
Перемещение курсора.....	115
Изменение текста.....	115
Вырезание и вставка (удаление и возврат) текста.....	117
Дополнение.....	117
Использование истории .....	119
Поиск в истории .....	119
Подстановка записей истории .....	121
Заключение.....	122

---

<b>Глава 9. Привилегии.....</b>	<b>123</b>
Владельцы, члены группы и все остальные .....	124
Чтение, запись и выполнение.....	126
chmod — изменение режима доступа к файлу .....	128
Установка режима доступа к файлу с помощью графического интерфейса.....	131
umask — определение разрешений доступа к файлам по умолчанию.....	132
Некоторые специальные разрешения .....	134
Изменение идентичности.....	135
su — запуск командной оболочки с подстановкой идентификаторов пользователя и группы.....	136
sudo — выполнение команды от имени другого пользователя.....	137
chown — изменение владельца и группы файла.....	139
chgrp — изменение группы файла.....	140
Использование привилегий .....	140
Изменение своего пароля .....	142
Заключение.....	143
<b>Глава 10. Процессы .....</b>	<b>144</b>
Как действует процесс.....	144
Просмотр списка процессов.....	145
Просмотр состояния процессов в динамике с помощью top .....	148
Управление процессами .....	150
Прерывание процесса .....	151
Перевод процессов в фоновый режим.....	151
Возврат процесса на передний план.....	152
Приостановка процесса.....	152
Сигналы .....	153
Посылка сигналов процессам командой kill.....	154
Посылка сигналов нескольким процессам с помощью killall .....	156
Остановка системы.....	157
Другие команды управления процессами.....	158
Заключение.....	158

---

<b>Часть II. ОКРУЖЕНИЕ И НАСТРОЙКА .....</b>	<b>159</b>
<b>Глава 11. Окружение .....</b>	<b>160</b>
Что хранится в окружении? .....	160
Исследование окружения .....	161
Некоторые интересные переменные.....	162
Как устанавливается окружение? .....	163
Что находится в файлах запуска?.....	165
Изменение окружения .....	167
Какие файлы следует изменять? .....	167
Текстовые редакторы .....	167
Использование текстового редактора.....	168
Активация изменений.....	172
Заключение .....	172
<b>Глава 12. Плавное введение в vi .....</b>	<b>173</b>
Зачем осваивать vi .....	173
Немного предыстории .....	174
Запуск и завершение vi .....	174
Режимы редактирования .....	176
Переход в режим вставки.....	177
Сохранение изменений.....	177
Перемещение курсора .....	178
Основы редактирования .....	179
Добавление текста в конец .....	179
Вставка строки .....	180
Удаление текста.....	181
Вырезание, копирование и вставка текста.....	182
Объединение строк .....	183
Поиск и замена.....	184
Поиск в пределах строки.....	184
Поиск во всем файле.....	184
Глобальный поиск и замена .....	185



---

Редактирование нескольких файлов .....	186
Переключение между файлами .....	187
Открытие дополнительных файлов для редактирования .....	188
Копирование содержимого из одного файла в другой .....	188
Вставка целого файла в другой файл .....	189
Сохранение результатов работы.....	190
Заключение.....	191
<b>Глава 13. Настройка приглашения к вводу .....</b>	<b>192</b>
Устройство строки приглашения к вводу .....	192
Альтернативные варианты оформления приглашения .....	194
Добавление цвета .....	196
Перемещение курсора .....	198
Сохранение определения приглашения .....	200
Заключение.....	200
<b>Часть III. ТИПИЧНЫЕ ЗАДАЧИ И ОСНОВНЫЕ ИНСТРУМЕНТЫ .....</b>	<b>201</b>
<b>Глава 14. Управление пакетами .....</b>	<b>202</b>
Системы пакетов .....	203
Как действует система пакетов.....	203
Файлы пакетов.....	203
Репозитории.....	204
Зависимости.....	205
Высоко- и низкоуровневые инструменты управления пакетами .....	205
Типичные задачи управления пакетами.....	206
Поиск пакета в репозитории .....	206
Установка пакета из репозитория.....	206
Установка пакета из файла пакета .....	207
Удаление пакета .....	207
Обновление пакетов из репозитория.....	208
Обновление пакета из файла пакета.....	208
Список установленных пакетов .....	209
Определение, установлен ли пакет .....	209

---

Вывод информации об установленном пакете .....	209
Поиск пакета по установленному файлу .....	210
Заключение .....	210
<b>Глава 15. Устройства хранения .....</b>	<b>212</b>
Монтирование и размонтирование устройств хранения .....	213
Просмотр списка смонтированных файловых систем .....	214
Определение названий устройств .....	218
Создание новых файловых систем .....	221
Управление разделами с помощью fdisk .....	221
Создание новой файловой системы с помощью mkfs .....	224
Проверка и восстановление файловой системы .....	225
Непосредственное перемещение данных между устройствами .....	226
Создание образа компакт-диска .....	226
Создание образа-копии компакт-диска .....	227
Создание образа из коллекции файлов .....	227
Запись образа компакт-диска .....	228
Непосредственное монтирование файла ISO-образа .....	228
Очистка перезаписываемых компакт-дисков .....	228
Запись образа .....	228
Заключительное замечание .....	229
Дополнительные сведения .....	229
<b>Глава 16. Сети .....</b>	<b>231</b>
Исследование и мониторинг сети .....	232
ping .....	232
tracert .....	233
ip .....	234
netstat .....	235
Передача файлов по сети .....	237
ftp .....	237
lftp — более удачная версия ftp .....	239
wget .....	239



---

Безопасные взаимодействия с удаленными узлами.....	240
ssh .....	240
scp и sftp .....	245
Заключение .....	246
<b>Глава 17. Поиск файлов .....</b>	<b>247</b>
locate — простой способ поиска файлов .....	247
find — сложный способ поиска файлов .....	249
Проверки .....	250
Операторы .....	253
Предопределенные операции.....	255
Операции, определяемые пользователем.....	257
Увеличение эффективности .....	258
xargs .....	259
Возвращаемся в песочницу .....	260
Параметры .....	262
Заключение.....	263
<b>Глава 18. Архивация и резервное копирование .....</b>	<b>264</b>
Сжатие файлов.....	264
gzip .....	265
bzip2 .....	268
Архивирование файлов .....	268
tar .....	269
zip .....	274
Синхронизация файлов и каталогов .....	276
Использование rsync для копирования по сети.....	279
Заключение.....	280
<b>Глава 19. Регулярные выражения .....</b>	<b>281</b>
Что такое регулярные выражения?.....	281
grep.....	282
Метасимволы и литералы .....	284
Любой символ .....	284

Якоря .....	285
Выражения в квадратных скобках и классы символов.....	286
Отрицание .....	287
Традиционные диапазоны символов.....	287
Классы символов POSIX.....	288
Простые и расширенные регулярные выражения POSIX .....	291
Чередование .....	293
Квантификаторы .....	295
? — совпадение с элементом ноль или один раз.....	295
* — совпадение с элементом ноль или более раз.....	295
+ — совпадение с элементом один или более раз .....	296
{ } — совпадение с элементом определенное число раз.....	297
Практические примеры применения регулярных выражений .....	298
Проверка списка телефонов с помощью grep .....	298
Поиск необычных имен файлов с помощью find .....	299
Поиск файлов с помощью locate.....	299
Поиск текста в less и vim.....	300
Заключение.....	301
<b>Глава 20. Обработка текста .....</b>	<b>303</b>
Области применения текста .....	304
Документы .....	304
Веб-страницы.....	304
Электронная почта.....	304
Вывод на принтер .....	304
Исходный код программ .....	305
А вот и наши старые знакомые!.....	305
cat.....	305
sort.....	307
uniq — выявление или удаление повторяющихся строк.....	314
Нарезка и перетасовка текста .....	316
cut — удаление фрагментов из всех строк в файлах.....	316
paste — слияние строк из файлов .....	319
join — объединение строк из двух файлов по общему полю .....	321

---

Сравнение текста .....	323
comm — построчное сравнение двух сортированных файлов .....	323
diff — построчное сравнение файлов .....	324
patch — применение diff-файла к оригиналу.....	327
Редактирование на лету .....	328
tr — перекодирование или удаление символов.....	328
sed — потоковый редактор для фильтрации и преобразования текста .....	331
aspell — интерактивная проверка орфографии.....	339
Заключение.....	343
Дополнительное задание.....	343
<b>Глава 21. Форматирование вывода.....</b>	<b>344</b>
Инструменты простого форматирования.....	344
nl — нумерация строк.....	345
fold — перенос строк после указанной длины.....	348
fmt — простое форматирование текста .....	349
pr — форматирование текста для печати .....	352
printf — форматирование и вывод данных.....	353
Системы форматирования документов.....	357
groff .....	358
Заключение .....	364
<b>Глава 22. Печать .....</b>	<b>365</b>
Краткая история поддержки печати.....	365
Печать в ночное время.....	366
Символьные принтеры .....	366
Графические принтеры .....	367
Печать в Linux .....	369
Подготовка файлов к печати .....	369
pr — преобразование текстовых файлов для печати .....	369
Отправка задания печати на принтер.....	371
lpr — печать файлов (в стиле Berkeley) .....	371
lp — печать файлов (в стиле System V) .....	372
Еще одна возможность: a2ps.....	373



---

Наблюдение за заданиями печати и управление ими .....	376
lprstat — вывод информации о состоянии принтера.....	376
lprm — вывод информации о состоянии очереди печати .....	377
lprm и cancel — отмена заданий печати.....	378
Заключение.....	378
<b>Глава 23. Компиляция программ.....</b>	<b>379</b>
Что такое компиляция? .....	380
Все ли программы компилируются?.....	381
Компиляция программ на C .....	382
Получение исходного кода .....	382
Исследование дерева исходных текстов .....	384
Сборка программ.....	386
Установка программ.....	390
Заключение.....	390
<b>Часть IV. СЦЕНАРИИ КОМАНДНОЙ ОБОЛОЧКИ .....</b>	<b>391</b>
<b>Глава 24. Создание первого сценария командной оболочки.....</b>	<b>392</b>
Что такое сценарии командной оболочки .....	392
Как написать сценарий командной оболочки .....	393
Формат файла сценария.....	393
Разрешения на выполнение .....	394
Местоположение файла сценария .....	394
Выбор местоположения для сценариев .....	396
Дополнительные хитрости по оформлению .....	397
Длинные имена параметров .....	397
Отступы и продолжения строк .....	397
Заключение.....	398
<b>Глава 25. Начало проекта .....</b>	<b>399</b>
Этап первый: минимальный документ .....	399
Этап второй: добавление некоторых данных .....	401
Переменные и константы .....	402
Присваивание значений переменным и константам.....	405

---

Встроенные документы.....	407
Заключение.....	409
<b>Глава 26. Проектирование сверху вниз.....</b>	<b>410</b>
Функции командной оболочки.....	411
Локальные переменные.....	414
Постоянное опробование сценария.....	416
Заключение.....	419
<b>Глава 27. Управление потоком выполнения: ветвление при помощи if.....</b>	<b>420</b>
Инструкция if.....	420
Код завершения.....	421
Команда test.....	423
Выражения для проверки файлов.....	423
Выражения для проверки строк.....	426
Выражения для проверки целых чисел.....	428
Более современная версия команды test.....	429
(( )) — для проверки целых чисел.....	430
Объединение выражений.....	431
Операторы управления: еще один способ ветвления.....	434
Заключение.....	435
<b>Глава 28. Чтение ввода с клавиатуры.....</b>	<b>437</b>
read — чтение значений со стандартного ввода.....	438
Параметры.....	440
Выделение полей в строке ввода с помощью IFS.....	442
Проверка ввода.....	444
Меню.....	446
Заключение.....	448
Дополнительные сведения.....	448
<b>Глава 29. Управление потоком выполнения: циклы while и until.....</b>	<b>449</b>
Циклы.....	449
while.....	450

---

Прерывание цикла.....	452
until.....	454
Чтение файлов в циклах.....	454
Заключение.....	455
<b>Глава 30. Поиск и устранение ошибок.....</b>	<b>456</b>
Синтаксические ошибки.....	456
Отсутствующие кавычки.....	457
Отсутствующие или неожиданные лексемы.....	458
Непредвиденная подстановка.....	458
Логические ошибки.....	460
Защитное программирование.....	460
Будьте внимательны к именам файлов.....	462
Проверка ввода.....	463
Тестирование.....	464
Комплекты тестов.....	465
Отладка.....	465
Поиск проблемной области.....	465
Трассировка.....	466
Исследование значений в процессе выполнения.....	468
Заключение.....	469
<b>Глава 31. Управление потоком выполнения: ветвление с помощью case.....</b>	<b>470</b>
Команда case.....	470
Шаблоны.....	472
Выполнение нескольких вариантов.....	474
Заключение.....	476
<b>Глава 32. Позиционные параметры.....</b>	<b>477</b>
Доступ к командной строке.....	477
Определение числа аргументов.....	478
shift — доступ к множеству аргументов.....	479
Простые приложения.....	480



---

Использование позиционных параметров в функциях .....	481
Обработка позиционных параметров скопом .....	482
Более сложное приложение .....	484
Заключение .....	487
<b>Глава 33. Управление потоком выполнения: цикл for .....</b>	<b>491</b>
for: традиционная форма .....	491
for: форма в стиле языка C .....	494
Заключение .....	496
<b>Глава 34. Строки и числа .....</b>	<b>498</b>
Подстановка параметров .....	498
Простые параметры .....	499
Подстановка пустых переменных .....	499
Получение имен переменных .....	501
Операции со строками .....	502
Преобразование регистра символов .....	505
Вычисление и подстановка арифметических выражений .....	507
Основание системы счисления .....	508
Унарные операторы .....	508
Простая арифметика .....	508
Присваивание .....	510
Битовые операции .....	512
Логические операторы .....	513
bc — язык калькулятора для вычислений с произвольной точностью .....	516
Применение bc .....	517
Пример сценария .....	518
Заключение .....	519
Дополнительные сведения .....	519
<b>35. Массивы .....</b>	<b>520</b>
Что такое массивы? .....	520
Создание массива .....	521
Присваивание значений массиву .....	521

Доступ к элементам массива.....	522
Операции с массивами .....	524
Вывод содержимого всего массива .....	524
Определение числа элементов в массиве .....	525
Поиск используемых индексов.....	526
Добавление элементов в конец массива.....	526
Сортировка массива .....	526
Удаление массива .....	527
Ассоциативные массивы .....	528
Заключение .....	529
<b>Глава 36. Экзотика.....</b>	<b>530</b>
Группы команд и подболочки .....	530
Подстановка процессов .....	534
Ловушки.....	536
Асинхронное выполнение с командой wait.....	540
Именованные каналы .....	541
Создание именованного канала.....	542
Использование именованных каналов .....	542
Заключение .....	543

# 31

## Управление потоком выполнения: ветвление с помощью case

В этой главе мы продолжим знакомство с инструментами управления потоком выполнения. В главе 28 мы сконструировали простое меню и реализовали логику обработки выбора его пунктов пользователем. Для этого использовалась серия команд `if`, выясняющих, какой из возможных вариантов выбран. Такие конструкции часто можно увидеть в программах, причем так часто, что в некоторых языках программирования (включая командную оболочку) был реализован механизм управления потоком выполнения для случаев с множеством альтернативных вариантов.

### Команда case

Командная оболочка `bash` поддерживает составную команду выбора из нескольких вариантов, которая называется `case`. Она имеет следующий синтаксис:

```
case слово in
    [шаблон [| шаблон]...] команды ;;]...
esac
```

Взгляните еще раз, как программа `read-menu` из главы 28 обрабатывает выбор пользователя:

```
#!/bin/bash

# read-menu: программа вывода системной информации,
#             управляемая с помощью меню
```



```
clear
echo "
Please Select:

1. Display System Information
2. Display Disk Space
3. Display Home Space Utilization
0. Quit
"
read -p "Enter selection [0-3] > "

if [[ "$REPLY" =~ ^[0-3]$ ]]; then
    if [[ "$REPLY" == 0 ]]; then
        echo "Program terminated."
        exit
    fi
    if [[ "$REPLY" == 1 ]]; then
        echo "Hostname: $HOSTNAME"
        uptime
        exit
    fi
    if [[ "$REPLY" == 2 ]]; then
        df -h
        exit
    fi
    if [[ "$REPLY" == 3 ]]; then
        if [[ "$(id -u)" -eq 0 ]]; then
            echo "Home Space Utilization (All Users)"
            du -sh /home/*
        else
            echo "Home Space Utilization ($USER)"
            du -sh "$HOME"
        fi
        exit
    fi
else
    echo "Invalid entry." >&2
    exit 1
fi
```

С помощью case можно сделать логику выбора немного проще:

```
#!/bin/bash
```

```
# case-menu: программа вывода системной информации,
```

```
#           управляемая с помощью меню

clear
echo "
Please Select:

1. Display System Information
2. Display Disk Space
3. Display Home Space Utilization
0. Quit
"
read -p "Enter selection [0-3] > "

case "$REPLY" in
    0)    echo "Program terminated."
          exit
          ;;
    1)    echo "Hostname: $HOSTNAME"
          uptime
          ;;
    2)    df -h
          ;;
    3)    if [[ "$(id -u)" -eq 0 ]]; then
          echo "Home Space Utilization (All Users)"
          du -sh /home/*
          else
          echo "Home Space Utilization ($USER)"
          du -sh "$HOME"
          fi
          ;;
    *)    echo "Invalid entry" >&2
          exit 1
          ;;
esac
```

Команда `case` берет значение *слова* — в данном примере значение переменной `REPLY` — и затем сопоставляет его с указанными *шаблонами*. Найдя соответствие, она выполняет команды, связанные с найденным шаблоном. После нахождения соответствия сопоставление с нижележащими шаблонами уже не производится.

## Шаблоны

Шаблоны обрабатываются командой `case` точно так же, как пути механизмом подстановки. Шаблоны завершаются символом `)`. В табл. 31.1 перечислены некоторые допустимые шаблоны.

**Таблица 31.1.** Примеры шаблонов в команде case

Шаблон	Описание
a)	Соответствует, если <i>слово</i> содержит a
[[[:alpha:]]])	Соответствует, если <i>слово</i> содержит единственный алфавитный символ
???)	Соответствует, если <i>слово</i> содержит ровно три символа
*.txt)	Соответствует, если <i>слово</i> заканчивается символами .txt
*)	Соответствует любому значению <i>слова</i> . Считается хорошей практикой включать этот шаблон в команду case последним, чтобы перехватывать любые значения <i>слова</i> , не соответствующие ни одному из предыдущих шаблонов, то есть чтобы перехватывать любые недопустимые значения

Следующий пример демонстрирует работу шаблонов:

```
#!/bin/bash

read -p "enter word > "

case "$REPLY" in
  [[[:alpha:]]]) echo "is a single alphabetic character." ;;
  [ABC][0-9])   echo "is A, B, or C followed by a digit." ;;
  ???)         echo "is three characters long." ;;
  *.txt)       echo "is a word ending in '.txt'" ;;
  *)           echo "is something else." ;;
esac
```

Мы можем объединить несколько шаблонов, перечислив их через символ вертикальной черты. В результате получается комбинированный условный шаблон, объединенный по «ИЛИ». Эта возможность может пригодиться, например, для обработки символов верхнего и нижнего регистров:

```
#!/bin/bash

# case-menu: программа вывода системной информации,
#             управляемая с помощью меню

clear
echo "
Please Select:
A. Display System Information
```



```

B. Display Disk Space
C. Display Home Space Utilization
Q. Quit
"
read -p "Enter selection [A, B, C or Q] > "

case "$REPLY" in
    q|Q)    echo "Program terminated."
            exit
            ;;
    a|A)    echo "Hostname: $HOSTNAME"
            uptime
            ;;
    b|B)    df -h
            ;;
    c|C)    if [[ "$(id -u)" -eq 0 ]]; then
            echo "Home Space Utilization (All Users)"
            du -sh /home/*
            else
            echo "Home Space Utilization ($USER)"
            du -sh "$HOME"
            fi
            ;;
    *)      echo "Invalid entry" >&2
            exit 1
            ;;
esac

```

Здесь мы изменили программу `case-menu`, предложив пользователю выбирать пункты меню вводом букв, а не цифр. Обратите внимание, что новые шаблоны позволяют вводить буквы обоих регистров — верхнего и нижнего.

## Выполнение нескольких вариантов

В версиях `bash` до 4.0 команда `case` могла выполнить только один вариант, соответствующий совпавшему шаблону. После этого команда завершалась. Рассмотрим сценарий, проверяющий введенный символ:

```

#!/bin/bash

# case4-1: проверка символа

read -n 1 -p "Type a character > "
echo
case "$REPLY" in

```

```
[[:upper:]] echo "'$REPLY' is upper case." ;;
[[:lower:]] echo "'$REPLY' is lower case." ;;
[[:alpha:]] echo "'$REPLY' is alphabetic." ;;
[[:digit:]] echo "'$REPLY' is a digit." ;;
[[:graph:]] echo "'$REPLY' is a visible character." ;;
[[:punct:]] echo "'$REPLY' is a punctuation symbol." ;;
[[:space:]] echo "'$REPLY' is a whitespace character." ;;
[[:xdigit:]] echo "'$REPLY' is a hexadecimal digit." ;;
esac
```

Вот как выглядит результат выполнения этого сценария:

```
[me@linuxbox ~]$ case4-1
Type a character > a
'a' is lower case.
```

В большинстве случаев сценарий прекрасно справляется со своей задачей, но терпит неудачу, если символ соответствует нескольким символьным классам POSIX. Например, символ `a` соответствует классам алфавитных символов и символов нижнего регистра, а также шестнадцатеричных цифр. В `bash` до версии 4.0 не было никакой возможности заставить `case` выполнить больше одной успешной проверки. Современные версии `bash` поддерживают дополнительную нотацию `;&` в конце каждого варианта, которая используется, как показано ниже:

```
#!/bin/bash

# case4-2: проверка символа

read -n 1 -p "Type a character > "
echo
case "$REPLY" in
  [[:upper:]] echo "'$REPLY' is upper case." ;;&
  [[:lower:]] echo "'$REPLY' is lower case." ;;&
  [[:alpha:]] echo "'$REPLY' is alphabetic." ;;&
  [[:digit:]] echo "'$REPLY' is a digit." ;;&
  [[:graph:]] echo "'$REPLY' is a visible character." ;;&
  [[:punct:]] echo "'$REPLY' is a punctuation symbol." ;;&
  [[:space:]] echo "'$REPLY' is a whitespace character." ;;&
  [[:xdigit:]] echo "'$REPLY' is a hexadecimal digit." ;;&
esac
```

Запустив этот сценарий, мы получим:

```
[me@linuxbox ~]$ case4-2
Type a character > a
'a' is lower case.
```

```
'a' is alphabetic.  
'a' is a visible character.  
'a' is a hexadecimal digit.
```

Дополнительный синтаксис `;;&` позволяет команде `case` продолжить проверку вместо простого завершения после первого найденного совпадения.

## Заключение

Команда `case` является удобным дополнением к нашей коллекции приемов программирования. Как будет показано в следующей главе, она отлично подходит для решения некоторых видов задач.