0. Напутствие в качестве вступления.

Ум подобен желудку.
Важно не то, сколько ты в него вложишь,
а то, сколько он сможет переварить.

В этой книге вы найдете ряд задач, примеров, алгоритмов, советов и стилистических замечаний по использованию языка программирования "*C*" (*Си*) в среде операционной системы *UNIX*. Здесь собраны этюды разной сложности и "штрихи к портрету" языка *Си*. Также описаны различные "подводные камни" на которых нередко терпят крушение новички в *Си*. В этом смысле эту книгу можно местами назвать "Как **не надо** программировать на *Си*".

В большинстве случаев в качестве платформы используется персональный компьютер *IBM PC* с какой-либо системой *UNIX*, либо *SPARCstation 20* с системой *Solaris 2* (тоже *UNIX svr4*), но многие примеры без каких-либо изменений (либо с минимумом таковых) могут быть перенесены в среду *MS DOS*[*], либо на другой тип машины с системой *UNIX*.

Это ваша ВТОРАЯ книга по Си. Эта книга не учебник, а хрестоматия к учебнику. Она не является ни систематическим курсом по *Си*, ни справочником по нему, и предназначена не для одноразового последовательного прочтения, а для чтения в несколько проходов на разных этапах вашей "зрелости". Поэтому читать ее следует **вместе** с "настоящим" учебником по *Си*, среди которых наиболее известна книга Кернигана и Ритчи.

Эта книга - *не ПОСЛЕДНЯЯ ваша книга по Си*. Во-первых потому, что кое-что в языке все же меняется со временем, хотя и настал час, когда стандарт на язык Си наконец принят... Но появился язык C++, который развивается довольно динамично. Еще есть Objective-C. Во-вторых потому, что есть **библиотеки** и **системные вызовы**, которые развиваются вслед за развитием *UNIX* и других операционных систем. Следующими вашими (настольными) книгами должны стать "Справочное руководство": *man2* (по системным вызовам), *man3* (по библиотечным функциям).

Мощь языка Си - в существующем многообразии библиотек.

Прошу вас с первых же шагов следить за стилем оформления своих программ. Делайте отступы, пишите комментарии, используйте осмысленные имена переменных и функций, отделяйте логические части программы друг от друга пустыми строками. Помните, что "лишние" пробелы и пустые строки в *Си* допустимы везде, кроме изображений констант и имен. Программы на *Си*, набитые в одну колонку (как на *FORTRAN*-е) очень тяжело читать и понимать. Из-за этого бывает трудно находить потерянные скобки { и }, потерянные символы `;' и другие ошибки.

Существует несколько "школ" оформления программ - приглядитесь к примерам в этой книге и в других источниках - и выберите любую! Ничего страшного, если вы будете смешивать эти стили. Но - ПОДАЛЬШЕ ОТ *FORTRAN*-а !!!

Программу можно автоматически сформатировать к "каноническому" виду при помощи, например, программы *cb*.

```
cb < НашФайл.c > /tmp/$$
mv /tmp/$$ НашФайл.c
```

но лучше сразу оформлять программу правильно.

Выделяйте логически самостоятельные ("замкнутые") части программы в функции (даже если они будут вызываться единственный раз). Функции - не просто средство избежать повторения одних и тех же операторов в тексте программы, но и средство структурирования процесса программирования, делающее программу более понятной. Во-первых, вы можете в другой программе использовать текст уже написанной вами ранее функции вместо того, чтобы писать ее заново. Во-вторых, операцию, оформленную в виде функции, можно рассматривать

как неделимый примитив (от довольно простого по смыслу, вроде *strcmp*, *strcpy*, до довольно сложного – *qsort*, *malloc*, *gets*) и забыть о его внутреннем устройстве (это хорошо – надо меньше помнить).

Не гонитесь за краткостью в ущерб ясности. *Си* позволяет порой писать такие выражения, над которыми можно полчаса ломать голову. Если же их записать менее мудрено, но чуть длиннее – они самоочевидны (и этим более защищены от ошибок).

В системе *UNIX* вы можете посмотреть описание любой команды системы или функции *Си*, набрав команду

man **названиеФункции**

(*man* – от слова **manual**, "руководство").

Еще одно напутствие: учите английский язык! Практически все языки программирования используют английские слова (в качестве ключевых слов, терминов, имен переменных и функций). Поэтому лучше понимать значение этих слов (хотя и восприятие их как просто неких символов тоже имеет определенные достоинства). Обратно – программирование на *Си* поможет вам выучить английский.

По различным причинам на территории России сейчас используется много разных восьмибитных русских кодировок. Среди них:

КОИ-8

Исторически принятая на русских *UNIX* системах – самая ранняя из появившихся.

Отличается тем свойством, что если у нее обрезан восьмой бит: **c** & 0177 – то она все же читаема с терминала как транслитерация латинских букв. Именно этой кодировкой пользуется автор этой книги (как и большинство *UNIX*-sites сети RelCom).

ISO 8859/5

Это американский стандарт на русскую кодировку. А русские программисты к ее разработке не имеют никакого отношения. Ею пользуется большинство коммерческих баз данных.

Microsoft 1251

Это та кодировка, которой пользуется *Microsoft Windows*. Возможно, что именно к

этой кодировке придут и *UNIX* системы (гипотеза 1994 года).

Альтернативная кодировка для **MS DOS**

Русская кодировка с псевдографикой, использовавшаяся в *MS DOS*.

Кодировка для Macintosh Это великое "разнообразие" причиняет массу неудобств. Но, господа, это Россия – что значит – широта души и абсолютный бардак. Relax and enjoy.

Многие примеры в данной книге даны вместе с ответами – как образцами для подражания. Однако мы надеемся, что Вы удержитесь от искушения и сначала проверите свои силы, а лишь потом посмотрите в ответ! Итак, читая примеры – делайте по аналогии.

* *MS DOS* – торговый знак фирмы *Microsoft Corporation*. (читается "Майкрософт");
DOS – дисковая операционная система.

© Copyright А. Богатырев, 1992-95
Си в UNIX

Содержание | Вперед

[Главная] [Гостевая]

Рейт ТОП Разработка игр