Введение

Си — универсальный язык программирования. Он тесно связан с системой UNIX, так как был разработан в этой системе, которая как и большинство программ, работающих в ней, написаны на Си. Однако язык не привязан жестко к какой-то одной операционной системе или машине. Хотя он и назван "языком системного программирования", поскольку удобен для написания компиляторов и операционных систем, оказалось, что на нем столь же хорошо писать большие программы другого профиля.

Многие важные идеи Си взяты из языка BCPL, автором которого является Мартин Ричарде. Влияние BCPL на Си было косвенным — через язык B, разработанный Кеном Томпсоном в 1970 г. для первой системы UNIX, реализованной на PDP-7.

BCPL и B — "бестиповые" языки. В отличие от них Си обеспечивает разнообразие типов данных. Базовыми типами являются символы, а также целые и числа с плавающей точкой различных размеров. Кроме того, имеется возможность получать целую иерархию производных типов данных из указателей, массивов, структур и объединений. Выражения формируются из операторов и операндов. Любое выражение, включая присваивание и вызов функции, может быть инструкцией. Указатели обеспечивают машинно-независимую адресную арифметику.

В Си имеются основные управляющие конструкции, используемые в хорошо структурированных программах: составная инструкция (`{...}`), ветвление по условию (`if-else`), выбор одной альтернативы из многих (`switch`), циклы с проверкой наверху (`while`, `for`) и с проверкой внизу (`do`), а также средство прерывания цикла (`break`).

В качестве результата функции могут возвращать значения базовых типов, структур, объединений и указателей. Любая функция допускает рекурсивное обращение к себе. Как правило, локальные переменные функции — "автоматические", т. е. они создаются заново при каждом обращении к ней. Определения функций нельзя вкладывать друг в друга, но объявления переменных разрешается строить в блочно-структурной манере. Функции программы на Си могут храниться в отдельных исходных файлах и компилироваться независимо. Переменные по отношению к функции могут быть внутренними и внешними. Последние могут быть доступными в пределах одного исходного файла или всей программы.

На этапе препроцессирования выполняется макроподстановка в текст программы, включение других исходных файлов и у словная компиляция.

Си — язык сравнительно "низкого уровня". Однако это вовсе не умаляет его достоинств, просто Си имеет дело с теми же объектами, что и большинство компьютеров, т. е. с символами, числами и адресами. С ними, можно оперировать при помощи арифметических и логических операций, выполняемых реальными машинами.

В Си нет прямых операций над составными объектами, такими как строки символов, множества, списки и массивы. В нем нет операций, которые бы манипулировали с целыми массивами или строками символов, хотя структуры разрешается копировать целиком как единые объекты. В языке нет каких-либо средств распределения памяти, помимо возможности определения статических переменных и стекового механизма при выделении места для локальных переменных внутри функций. Нет в нем "кучи" и "сборщика мусора". Наконец, в самом Си нет средств ввода-вывода, инструкций `READ` (читать) и `WRITE` (писать) и каких-либо методов доступа к файлам. Все это — механизмы высокого уровня, которые в Си обеспечиваются исключительно с помощью явно вызываемых функций. Большинство реализованных Си-систем содержат в себе разумный стандартный набор этих функций.

В продолжение сказанного следует отметить, что Си предоставляет средства лишь последовательного управления ходом вычислений: механизм ветвления по условиям, циклы, составные инструкции,

подпрограммы — и не содержит средств мультипрограммирования, параллельных процессов, синхронизации и организации сопрограмм.

Отсутствие некоторых из перечисленных средств может показаться серьезным недостатком ("выходит, чтобы сравнить две строки символов, нужно обращаться к функции?"). Однако компактность языка имеет реальные выгоды. Поскольку Си относительно мал, то и описание его кратко, и овладеть им можно быстро. Программист может реально рассчитывать на то, что он будет знать, понимать и на практике регулярно пользоваться всеми возможностями языка.

В течение многих лет единственным определением языка Си было первое издание книги "Язык программирования Си". В 1983 г. Институтом американских национальных стандартов (ANSI) учреждается комитет для выработки современного исчерпывающего определения языка Си. Результатом его работы явился стандарт для Си ("ANSI-C"), выпущенный в 1988 г. Большинство положений этого стандарта уже учтено в современных компиляторах.

Стандарт базируется на первоначальном справочном руководстве. По сравнению с последним язык изменился относительно мало. Одной из целей стандарта было обеспечить, чтобы в большинстве случаев существующие программы оставались правильными или вызывали предупреждающие сообщения компиляторов об изменении поведения.

Для большинства программистов самое важное изменение — это новый синтаксис объявления и определения функций. Объявление функции может теперь включать и описание ее аргументов. В соответствии с этим изменился и синтаксис определения функции. Дополнительная информация значительно облегчает компилятору выявление ошибок, связанных с несогласованностью аргументов; по нашему мнению, это очень полезное добавление к языку.

Следует также отметить ряд небольших изменений. В языке узаконены присваивание структур и перечисления, которые уже некоторое время широко используются. Вычисления с плавающей точкой теперь допускаются и с одинарной точностью. Уточнены свойства арифметики, особенно для беззнаковых типов. Усовершенствован препроцессор. Большинство программистов эти изменения затронут очень слабо.

Второй значительный вклад стандарта — это определение библиотеки, поставляемой вместе с Си-компилятором, в которой специфицируются функции доступа к возможностям операционной системы (например чтения-записи файлов), форматного ввода-вывода, динамического выделения памяти, манипуляций со строками символов и т. д. Набор стандартных заголовочных файлов обеспечивает единообразный доступ к объявлениям функций и типов данных. Гарантируется, что программы, использующие эту библиотеку при взаимодействии с операционной системой, будут работать также и на других машинах. Большинство программ, составляющих библиотеку, созданы по образу и подобию "стандартной библиотеки ввода-вывода" системы UNIX. Эта библиотека описана в первом издании книги и широко используется в других системах. И здесь программисты не заметят существенных различий.

Так как типы данных и управляющих структур языка Си поддерживаются командами большинства существующих машин, исполнительная система (run-time library), обеспечивающая независимый запуск и выполнение программ, очень мала. Обращения к библиотечным функциям пишет сам программист (не компилятор), поэтому при желании их можно легко заменить на другие. Почти все программы, написанные на Си, если они не касаются каких-либо скрытых в операционной системе деталей, переносимы на другие машины.

Си соответствует аппаратным возможностям многих машин, однако он не привязан к архитектуре какой-либо конкретной машины. Проявляя некоторую дисциплину, можно легко писать переносимые программы, т. е. программы, которые без каких-либо изменений могут работать на разных машинах. Стандарт предоставляет

возможность для явного описания переносимости с помощью набора констант, отражающих характеристики машины, на которой программа будет работать.

Си не является "строго типизированным" языком, но в процессе его развития контроль за типами был усилен. В первой версии Си хоть не одобрялся, но разрешался бесконтрольный обмен указателей и целых, что вызывало большие нарекания, но это уже давным-давно запрещено. Согласно стандарту теперь требуется явное объявление или явное указание преобразования, что уже и реализовано в хороших компиляторах. Новый вид объявления функций — еще один шаг в этом направлении. Компилятор теперь предупреждает о большей части ошибок в типах и автоматически не выполняет преобразования данных несовместимых типов. Однако основной философией Си остается то, что программисты сами знают, что делают; язык лишь требует явного указания об их намерениях.

Си, как и любой другой язык программирования, не свободен от недостатков. Уровень старшинства некоторых операторов не является общепринятым, некоторые синтаксические конструкции могли бы быть лучше. Тем не менее, как оказалось, Си — чрезвычайно эффективный и выразительный язык, пригодный для широкого класса задач.