- **C 语言小而简单**。C 语言的设计是由一个人而非一个协会掌控的，因此这是一个简洁明了、没有什么冗赘的设计。K&R 这本书用大量的例子和练习描述了完整的 C 语言及其标准库，而全书不过 261 页。C 语言的简单使它相对而言易于学习，也易于移植到不同的计算机上。
- **C 语言是为实践目的设计的**。C 语言是设计用来实现 Unix 操作系统的。后来，其他人发现能够用这门语言无障碍地编写他们想要的程序。

C 语言是系统级编程的首选，同时它也非常适用于应用级程序的编写。然而，它也并非适用于所有的程序员和所有的情况。C 语言的指针是造成程序员困惑和程序错误的一个常见原因。同时，C 语言还缺乏对非常有用的抽象的显式支持，例如类、对象和异常。像 C++ 和 Java 这样针对应用级程序的新程序语言解决了这些问题。

1.2　程序被其他程序翻译成不同的格式

hello 程序的生命周期是从一个高级 C 语言程序开始的，因为这种形式能够被人读懂。然而，为了在系统上运行 hello.c 程序，每条 C 语句都必须被其他程序转化为一系列的低级机器语言指令。然后这些指令按照一种称为可执行目标程序的格式打好包，并以二进制磁盘文件的形式存放起来。目标程序也称为可执行目标文件。

在 Unix 系统上，从源文件到目标文件的转化是由编译器驱动程序完成的：

```
linux> gcc -o hello hello.c
```

在这里，GCC 编译器驱动程序读取源程序文件 hello.c，并把它翻译成一个可执行目标文件 hello。这个翻译过程可分为四个阶段完成，如图 1-3 所示。执行这四个阶段的程序（预处理器、编译器、汇编器和链接器）一起构成了编译系统（compilation system）。

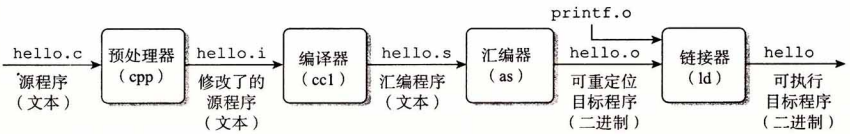

图 1-3　编译系统

- 预处理阶段。预处理器（cpp）根据以字符#开头的命令，修改原始的 C 程序。比如 hello.c 中第 1 行的#include <stdio.h> 命令告诉预处理器读取系统头文件 stdio.h 的内容，并把它直接插入程序文本中。结果就得到了另一个 C 程序，通常是以 .i 作为文件扩展名。
- 编译阶段。编译器（cc1）将文本文件 hello.i 翻译成文本文件 hello.s，它包含一个汇编语言程序。该程序包含函数 main 的定义，如下所示：

```
1   main:
2     subq    $8, %rsp
3     movl    $.LC0, %edi
4     call    puts
5     movl    $0, %eax
6     addq    $8, %rsp
7     ret
```

定义中 2～7 行的每条语句都以一种文本格式描述了一条低级机器语言指令。汇编语言是非常有用的，因为它为不同高级语言的不同编译器提供了通用的输出语