

- A. 这个程序会产生多少输出行？
 B. 这些输出行的一种可能的顺序是什么？

8.4.4 让进程休眠

sleep 函数将一个进程挂起一段指定的时间。

```
#include <unistd.h>

unsigned int sleep(unsigned int secs);
```

返回：还要休眠的秒数。


如果请求的时间量已经到了，sleep 返回 0，否则返回还剩下的要休眠的秒数。后一种情况是可能的，如果因为 sleep 函数被一个信号中断而过早地返回。我们将在 8.5 节中详细讨论信号。

我们会发现另一个很有用的函数是 pause 函数，该函数让调用函数休眠，直到该进程收到一个信号。

```
#include <unistd.h>

int pause(void);
```

总是返回 -1。

 **练习题 8.5** 编写一个 sleep 的包装函数，叫做 snooze，带有下列的接口：

```
unsigned int snooze(unsigned int secs);
```

snooze 函数和 sleep 函数的行为完全一样，除了它会打印出一条消息来描述进程实际休眠了多长时间：

```
Slept for 4 of 5 secs.
```

8.4.5 加载并运行程序

execve 函数在当前进程的上下文中加载并运行一个新程序。

```
#include <unistd.h>

int execve(const char *filename, const char *argv[],
           const char *envp[]);
```

如果成功，则不返回，如果错误，则返回 -1。

execve 函数加载并运行可执行目标文件 filename，且带参数列表 argv 和环境变量列表 envp。只有当出现错误时，例如找不到 filename，execve 才会返回到调用程序。所以，与 fork 一次调用返回两次不同，execve 调用一次并从不返回。

参数列表是用图 8-20 中的数据结构表示的。argv 变量指向一个以 null 结尾的指针数组，其中每个指针都指向一个参数字符串。按照惯例，argv[0] 是可执行目标文件的名字。环境变量的列表是由一个类似的数据结构表示的，如图 8-21 所示。envp 变量指向一个以 null 结尾的指针数组，其中每个指针指向一个环境变量字符串，每个串都是形如“name=value”的名字-值对。