- shell 回收它所有的僵死子进程。如果任何作业因为收到一个未捕获的信号而终止，那么 shell 就输出一条消息到终端，消息中包含该作业的 PID 和对该信号的描述。

图 8-46 展示了一个 shell 会话示例。

```
linux> ./shell                                  Run your shell program
>bogus
bogus: Command not found.                       Execve can't find executable
>foo 10
Job 5035 terminated by signal: Interrupt        User types Ctrl+C
>foo 100 &
[1] 5036 foo 100 &
>foo 200 &
[2] 5037 foo 200 &
>jobs
[1] 5036 Running    foo 100 &
[2] 5037 Running    foo 200 &
>fg %1
Job [1] 5036 stopped by signal: Stopped         User types Ctrl+Z
>jobs
[1] 5036 Stopped    foo 100 &
[2] 5037 Running    foo 200 &
>bg 5035
5035: No such process
>bg 5036
[1] 5036 foo 100 &
>/bin/kill 5036
Job 5036 terminated by signal: Terminated
> fg %2                                         Wait for fg job to finish
>quit
linux>                                          Back to the Unix shell
```

图 8-46　家庭作业 8.26 的 shell 会话示例

练习题答案

8.1　进程 A 和 B 是互相并发的，就像 B 和 C 一样，因为它们各自的执行是重叠的，也就是一个进程在另一个进程结束前开始。进程 A 和 C 不是并发的，因为它们的执行没有重叠；A 在 C 开始之前就结束了。

8.2　在图 8-15 的示例程序中，父子进程执行无关的指令集合。然而，在这个程序中，父子进程执行的指令集合是相关的，这是有可能的，因为父子进程有相同的代码段。这会是一个概念上的障碍，所以请确认你理解了本题的答案。图 8-47 给出了进程图。

A. 这里的关键点是子进程执行了两个 printf 语句。在 fork 返回之后，它执行第 6 行的 printf。然后它从 if 语句中出来，执行第 7 行的 printf 语句。下面是子进程产生的输出：

```
p1: x=2
p2: x=1
```

B. 父进程只执行第 7 行的 printf：

```
p2: x=0
```

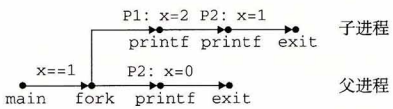

图 8-47　练习题 8.2 的进程图