束。一个更好的方法是更细粒度的多路复用，服务器每次循环(至多)回送一个文本行。

code/conc/select.c

```
1   #include "csapp.h"
2   void echo(int connfd);
3   void command(void);
4
5   int main(int argc, char **argv)
6   {
7       int listenfd, connfd;
8       socklen_t clientlen;
9       struct sockaddr_storage clientaddr;
10      fd_set read_set, ready_set;
11
12      if (argc != 2) {
13          fprintf(stderr, "usage: %s <port>\n", argv[0]);
14          exit(0);
15      }
16      listenfd = Open_listenfd(argv[1]);
17
18      FD_ZERO(&read_set);              /* Clear read set */
19      FD_SET(STDIN_FILENO, &read_set); /* Add stdin to read set */
20      FD_SET(listenfd, &read_set);     /* Add listenfd to read set */
21
22      while (1) {
23          ready_set = read_set;
24          Select(listenfd+1, &ready_set, NULL, NULL, NULL);
25          if (FD_ISSET(STDIN_FILENO, &ready_set))
26              command(); /* Read command line from stdin */
27          if (FD_ISSET(listenfd, &ready_set)) {
28              clientlen = sizeof(struct sockaddr_storage);
29              connfd = Accept(listenfd, (SA *)&clientaddr, &clientlen);
30              echo(connfd); /* Echo client input until EOF */
31              Close(connfd);
32          }
33      }
34  }
35
36  void command(void) {
37      char buf[MAXLINE];
38      if (!Fgets(buf, MAXLINE, stdin))
39          exit(0); /* EOF */
40      printf("%s", buf); /* Process the input command */
41  }
```

code/conc/select.c

图 12-6　使用 I/O 多路复用的迭代 echo 服务器。服务器使用 select 等待监听描述符上的连接请求和标准输入上的命令

练习题 12.3　在 Linux 系统里，在标准输入上键入 Ctrl+D 表示 EOF。图 12-6 中的程序阻塞在对 select 的调用上时，如果你键入 Ctrl+D 会发生什么？

12.2.1　基于 I/O 多路复用的并发事件驱动服务器

I/O 多路复用可以用做并发事件驱动(event-driven)程序的基础，在事件驱动程序中，某些事件会导致流向前推进。一般的思路是将逻辑流模型化为状态机。不严格地说，一个