Problem 1: Thread analysis .

One of your colleagues is thinking of using signals to allow a parent process to count events that occur in a child process. The idea is to notify the parent each time an event occurs by sending it a signal, and letting the parent's signal handler increment a global counter variable, which the parent can then inspect after the child has terminated. However, when he runs the test program in Figure 8.41 on his system, he discovers that when the parent calls printf, counter always has a value of 2, even though the child has sent five signals to the parent. Perplexed, he comes to you for help. Can you explain the bug?

Problem 2: Signals and nonlocal jumps .

Write a version of the fgets function, called tfgets, that times out after 5 seconds. The tfgets function accepts the same inputs as fgets. If the user doesn't type an input line within 5 seconds, tfgets returns NULL. Otherwise, it returns a pointer to the input line.

Problem 3: UNIX I/O .

What is the output of the following program?

```
#include "csapp.h"

int main()
{
    int fd1, fd2;

    fd1 = Open("foo.txt", O_RDONLY, 0);
    fd2 = Open("bar.txt", O_RDONLY, 0);
    Close(fd2);
    fd2 = Open("baz.txt", O_RDONLY, 0);
    printf("fd2 = %d\n", fd2);
    exit(0);
}
```