12.4.3　共享变量

我们说一个变量 v 是共享的，当且仅当它的一个实例被一个以上的线程引用。例如，示例程序中的变量 cnt 就是共享的，因为它只有一个运行时实例，并且这个实例被两个对等线程引用。在另一方面，myid 不是共享的，因为它的两个实例中每一个都只被一个线程引用。然而，认识到像 msgs 这样的本地自动变量也能被共享是很重要的。

练习题 12.6

A. 利用 12.4 节中的分析，为图 12-15 中的示例程序在下表的每个条目中填写“是”或者“否”。在第一列中，符号 $v.t$ 表示变量 v 的一个实例，它驻留在线程 t 的本地栈中，其中 t 要么是 m(主线程)，要么是 p0(对等线程 0)或者 p1(对等线程 1)。

变量实例	主线程引用的?	对等线程0引用的?	对等线程1引用的?
ptr			
cnt			
i.m			
msgs.m			
myid.po			
myid.p1			

B. 根据 A 部分的分析，变量 ptr、cnt、i、msgs 和 myid 哪些是共享的？

12.5　用信号量同步线程

共享变量是十分方便，但是它们也引入了同步错误(synchronization error)的可能性。考虑图 12-16 中的程序 badcnt.c，它创建了两个线程，每个线程都对共享计数变量 cnt 加 1。

code/conc/badcnt.c

```
1   /* WARNING: This code is buggy! */
2   #include "csapp.h"
3
4   void *thread(void *vargp);  /* Thread routine prototype */
5
6   /* Global shared variable */
7   volatile long cnt = 0; /* Counter */
8
9   int main(int argc, char **argv)
10  {
11      long niters;
12      pthread_t tid1, tid2;
13
14      /* Check input argument */
15      if (argc != 2) {
16          printf("usage: %s <niters>\n", argv[0]);
17          exit(0);
18      }
19      niters = atoi(argv[1]);
20
21      /* Create threads and wait for them to finish */
22      Pthread_create(&tid1, NULL, thread, &niters);
```

图 12-16　badcnt.c：一个同步不正确的计数器程序