某汽车轮渡口，过江渡船每次能载 10 辆车，每 10 分钟有一个渡轮到达。过江车辆分为客车和货车。上渡船有如下规定：客车先于货车上船，每上 4 辆客车允许上一辆货车；若等待的客车数不满 4 辆，则以货车代替。试编写一程序，模拟渡口的管理，统计客车和货车的平均等待时间。设车辆到达服从均匀分布，参数由用户指定。

【解】该模拟过程由三个阶段组成：生成客车到达事件，并保存在客车队列中；生成货车到达事件，保存在货车队列中；然后开始模拟。

每 10 分钟有一艘船到达。船上可以放 10 辆车。每上四辆客车可以上一辆货车。也就是说，一艘船上可以上 8 辆客车。于是先上 8 辆客车（如果有 8 辆以上的客车在等待），再上货车直到上满了 10 辆车货没有货车在等待。如果货车上完了，船上的车不满 10 辆，有可能是没有货车在等待，此时可以继续上客车直到上满 10 辆或没有客车在等待。每上一辆车，分别统计它的等待时间。当到达模拟要求的时间终点时，模拟结束，输出货车的等待时间、客车的等待时间、每艘船上平均的车数、以及模拟结束后尚在等待的车的数量。该过程见代码清单 4-4

**代码清单 4-4 车客渡的模拟**

```
1.  int main()
2.  {
3.      int busArrLow, busArrHigh, vanArrLow, vanArrHigh, busNum, vanNum;
4.      int totalTime, timer;      //totalTime:所需模拟的时间，timer：虚拟时钟
5.      int ship, shipTotal;  // ship：当前船上的车数，shipTotal：运走的车的总数
6.      double busWaitTime, vanWaitTime;          // 客车和货车的等待时间
7.      linkQueue<int> busQueue, vanQueue;          客车和货车的等待队列
8.
9.    // 输入模拟参数
10.     cout << "请输入所需模拟的时间（以分钟为单位）：";
11.     cin >> totalTime;
12.     cout << "请输入客车到达的间隔时间范围（下限 上限）：";
13.     cin >> busArrLow >> busArrHigh;
14.     cout << "请输入货车到达的间隔时间范围（下限 上限）：";
15.     cin >> vanArrLow >> vanArrHigh;
16.
17.     // 生成客车队列
18.   srand(time(NULL));
19.   timer = 0;
20.     busNum = 0;
21.     while (true) {
22.         timer += rand() % (busArrHigh - busArrLow + 1) + busArrLow;
23.         if (timer > totalTime) break;     // 到达时间超过了模拟的时间长度
24.         busQueue.enQueue(timer);
25.     }
26.
27.   // 生成货车队列
28.     timer = 0;
```

```
29.       vanNum = 0;
30.       while (true) {
31.            timer += rand() % (vanArrHigh - vanArrLow + 1) + vanArrLow;
32.            if (timer > totalTime) break;
33.            vanQueue.enQueue(timer);
34.       }
35.       cout << endl;
36.
37.    // 开始模拟
38.       busWaitTime = vanWaitTime = 0;      // 设置客车和货车的等待时间的初值
39.       busNum = vanNum = 0;                // 设置运走的客车数和货车数的初值
40.       shipTotal = 0;
41.       for (timer  = 10; timer <= totalTime; timer += 10) { // 模拟每艘船的到达
42.            for (ship = 0; ship < 8 && !busQueue.isEmpty()          // 上客车
43.           && busQueue.getHead() <= timer; ++ship) {
44.                 busWaitTime += timer - busQueue.deQueue(); // 统计客车等待时间
45.                 ++busNum;                          // 统计已过江的客车数
46.            }
47.            for (; ship < 10 &&  !vanQueue.isEmpty()              // 上货车
48.            && vanQueue.getHead() <= timer; ++ship) {
49.                 vanWaitTime += timer - vanQueue.deQueue();  // 统计货车等待时
    间
50.                 ++vanNum;                          // 统计已过江的货车数
51.            }
52.            for (; ship < 10 && !busQueue.isEmpty()        // 上完货车尚有余位
53.            && busQueue.getHead() <= timer; ++ship) {
54.                 busWaitTime += timer - busQueue.deQueue();
55.                 ++busNum;
56.            }
57.            shipTotal += ship;
58.       }
59.
60.       // 输出结果
61.       cout << "一共运走 " << shipTotal << "辆车，平均每艘船上有 "
62.        << shipTotal /((timer-10) / 10) << "辆车" << endl;
63.    cout << "客车的平均等待时间是 " << busWaitTime / busNum << endl;
64.    cout << "货车的平均等待时间是 " << vanWaitTime / vanNum << endl;
65.
66.       for (busNum = 0; !busQueue.isEmpty(); ++busNum) busQueue.deQueue();
67.       for (vanNum = 0; !vanQueue.isEmpty(); ++vanNum)
    vanQueue.deQueue();
68.
69.       cout << "还在等待的客车数和货车数分别为 " << busNum << " 和 "
70.        << vanNum << endl;
```

```
71.
72.         return 0;
73.  }
```