**问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201609-2 |
| 试题名称： | 火车购票 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　请实现一个铁路购票系统的简单座位分配算法，来处理一节车厢的座位分配。 　　假设一节车厢有20排、每一排5个座位。为方便起见，我们用1到100来给所有的座位编号，第一排是1到5号，第二排是6到10号，依次类推，第20排是96到100号。 　　购票时，一个人可能购一张或多张票，最多不超过5张。如果这几张票可以安排在同一排编号相邻的座位，则应该安排在编号最小的相邻座位。否则应该安排在编号最小的几个空座位中（不考虑是否相邻）。 　　假设初始时车票全部未被购买，现在给了一些购票指令，请你处理这些指令。  **输入格式**  　　输入的第一行包含一个整数*n*，表示购票指令的数量。 　　第二行包含*n*个整数，每个整数*p*在1到5之间，表示要购入的票数，相邻的两个数之间使用一个空格分隔。  **输出格式**  　　输出*n*行，每行对应一条指令的处理结果。 　　对于购票指令*p*，输出*p*张车票的编号，按从小到大排序。  **样例输入**  4 2 5 4 2  **样例输出**  1 2 6 7 8 9 10 11 12 13 14 3 4  **样例说明**  　　1) 购2张票，得到座位1、2。 　　2) 购5张票，得到座位6至10。 　　3) 购4张票，得到座位11至14。 　　4) 购2张票，得到座位3、4。  **评测用例规模与约定**  　　对于所有评测用例，1 ≤ *n* ≤ 100，所有购票数量之和不超过100。 |
| 代码程序： | #include<stdio.h>  #include<string.h>  int deal(int MPcount[25], int p);  int main()  {  bool Seat[105];  for(int i = 1; i<=100;i++)  {  Seat[i] = true;  }  int MPcount[25];  for(int i = 1; i<=20;i++)  {  MPcount[i] = 5;  }  int n;  scanf("%d",&n);  int p;  for(int i = 0; i<n;i++)  {  scanf("%d",&p);  int ret = deal(MPcount, p);  int count = 0;  if(ret>0)  {  for(int j = ret\*5-4;j<=ret\*5;j++)  {  if(Seat[j])  {  printf("%d",j);  count++;  Seat[j] = false;  if(count < p)  {  printf(" ");  }else{  break;  }  }  }  printf("\n");  MPcount[ret] -= p;  }else{// 没有找到整行  for(int i = 1; i<=100;i++)  {  if(Seat[i])  {  printf("%d",i);  count++;  Seat[i] = false;  MPcount[(i-1)/5+1] --;  if(count < p)  {  printf(" ");  }else{  break;  }  }  }  printf("\n");  }  }  }  int deal(int MPcount[25], int p)  {  for(int i = 1; i<=20;i++)  {  if(MPcount[i]>=p)  {  return i;;  }  }  return 0;  } |