|  |
| --- |
| /\*②player.c文件的完整内容\*/ |
|  | #include "player.h" |
|  | #include <stdio.h> |
|  |  |
|  | int readPla(Player \*pla , int n) /\*读入球员记录值，球员号码为0或读满规定条数记录时停止\*/ |
|  | { |
|  | int i,j; |
|  | for (i=0;i<n;i++) |
|  | { |
|  | printf("Input one player\'s information\n"); |
|  | printf("num: "); |
|  | scanf("%ld", &pla[i].num); |
|  | if (pla[i].num==0) break; |
|  | printf("name: "); |
|  | scanf("%s",pla[i].name); |
|  | printf("nation: "); |
|  | scanf("%s",pla[i].nation); |
|  | pla[i].total=0; /\*总分需要计算求得，初值置为0\*/ |
|  | printf("Input three courses of the player:\n"); |
|  | for (j=0;j<3;j++) |
|  | { |
|  | scanf("%d",&pla[i].score[j]); |
|  | } |
|  |  |
|  | } |
|  | return i; /\*返回实际读入的记录条数\*/ |
|  | } |
|  |  |
|  | void printPla ( Player \*pla , int n) /\*输出所有球员记录的值\*/ |
|  | { |
|  | int i,j; |
|  | for (i=0;i<n;i++) |
|  | { |
|  | printf("%8ld ", pla[i].num); |
|  | printf("%8s", pla[i].name); |
|  | printf("%8s", pla[i].nation); |
|  | for (j=0;j<3;j++) |
|  | printf("%6d",pla[i].score[j]); |
|  |  |
|  | printf("%5d\n",pla[i].rank); |
|  | } |
|  | } |
|  |  |
|  | int equal(Player s1,Player s2,int condition) /\*如何判断两个Player记录相等\*/ |
|  | { |
|  | if (condition==1) /\*如果参数condition的值为1，则比较球员号码\*/ |
|  | return s1.num==s2.num; |
|  | else if (condition==2) /\*如果参数condition的值为2，则比较姓名\*/ |
|  | { |
|  | if (strcmp(s1.name,s2.name)==0) return 1; |
|  | else return 0; |
|  | } |
|  | else if (condition==3) /\*如果参数condition的值为3，则比较名次\*/ |
|  | return s1.rank==s2.rank; |
|  |  |
|  |  |
|  | else return 1; /\*其余情况返回1\*/ |
|  | } |
|  |  |
|  | int larger(Player s1,Player s2,int condition) /\*根据condition条件比较两个Player记录的大小\*/ |
|  | { |
|  | if (condition==1) /\*如果参数condition的值为1，则比较球员号码\*/ |
|  | return s1.num>s2.num; |
|  |  |
|  |  |
|  | else return 1; /\*其余情况返回1\*/ |
|  | } |
|  |  |
|  | void reverse(Player pla[],int n) /\*数组元素逆置\*/ |
|  | { |
|  | int i; |
|  | Player temp; |
|  | for (i=0;i<n/2;i++) /\*循环次数为元素数量的一半\*/ |
|  | { |
|  | temp=pla[i]; |
|  | pla[i]=pla[n-1-i]; |
|  | pla[n-1-i]=temp; |
|  | } |
|  | } |
|  |  |
|  | void calcuTotal(Player pla[],int n) /\*计算所有球员的总分\*/ |
|  | { |
|  | int i,j; |
|  | for (i=0;i<n;i++) /\*外层循环控制所有球员记录\*/ |
|  | { |
|  | pla[i].total =0; |
|  |  |
|  | pla[i].total +=pla[i].score[1]; |
|  | } |
|  | } |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | |  | | --- | | void calcuMark(double m[3][3],Player pla[],int n) /\*求三项数据的最高、最低数\*/ | |  | /\*其中形式参数二维数组m的第一维代表三门课，第二维代表最高、最低\*/ | |  | { | |  | int i,j; | |  | for (i=0;i<3;i++) /\*求三项数据的最高分\*/ | |  | { | |  | m[i][0]=pla[0].score[i]; | |  | for (j=1;j<n;j++) | |  | if (m[i][0]<pla[j].score[i]) | |  | m[i][0]=pla[j].score[i]; | |  | } | |  | for (i=0;i<3;i++) /\*求三项数据的最低分\*/ | |  | { | |  | m[i][1]=pla[0].score[i]; | |  | for (j=1;j<n;j++) | |  | if (m[i][1]>pla[j].score[i]) | |  | m[i][1]=pla[j].score[i]; | |  | } | |  |  | |  |  | |  | ｝ | |  |  | |  |  | |  |  | |  |  | |  |  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | void sortPla(Player pla[],int n,int condition) /\*选择法排序，按condition条件由小到大排序\*/ |
|  | { |
|  | int i,j,minpos; /\*minpos用来存储本趟最小元素所在的下标\*/ |
|  | Player t; |
|  | for (i=0;i<n-1;i++) /\*控制循环的n-1趟\*/ |
|  | { |
|  | minpos=i; |
|  | for (j=i+1;j<n;j++) /\*寻找本趟最小元素所在的下标\*/ |
|  | if (larger(pla[minpos],pla[j],condition)) |
|  | minpos=j; |
|  | if (i!=minpos) /\*保证本趟最小元素到达下标为i的位置\*/ |
|  | { |
|  | t=pla [i]; |
|  | pla[i]=stu[minpos]; |
|  | pla[minpos]=t; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | int searchPla(Player pla[],int n,Player s,int condition,int f[ ]) /\*在pla数组中依condition条件查找\*/ |
|  | /\*与s相同的元素，由于不止一条记录符合条件，因此将这些元素的下标置于 f数组中\*/ |
|  | { |
|  | int i,j=0,find=0; |
|  | for (i=0;i<n;i++) /\*待查找的元素\*/ |
|  | if (equal(pla[i],s,condition)) |
|  | { |
|  | f[j++]=i; /\*找到了相等的元素，将其下标放到f数组中\*/ |
|  | find++; /\*统计找到的元素个数\*/ |
|  | } |
|  | return find; /\*返回find，其值为0则表示没找到\*/ |
|  | } |
|  |  |
|  | int insertPla(Player pla[],int n,Player s) /\*向pla数组中依球员号码递增插入一个元素s\*/ |
|  | { |
|  | int i; |
|  | sortPla(pla,n,1); /\*先按球员号码排序\*/ |
|  | for (i=0;i<n;i++) |
|  | { |
|  | if (equal(pla[i],s,1)) /\*球员号码相同不允许插入，保证球员号码的唯一性\*/ |
|  | { |
|  | printf("this record exist,can not insert again!\n"); |
|  | return n; |
|  | } |
|  | } |
|  | for (i=n-1;i>=0;i--) /\*按球员号码从小到大有序\*/ |
|  | { |
|  | if (!larger(pla[i],s,1)) /\*如果s大于当前元素pla[i]，则退出循环\*/ |
|  | break; |
|  | pla[i+1]=pla[i]; /\*否则元素pla[i]后移一个位置\*/ |
|  | } |
|  | pla[i+1]=s; /\*在下标i+1处插入元素s\*/ |
|  | n++; /\*元素个数增加1\*/ |
|  | return n; /\*返回现有元素个数\*/ |
|  | } |
|  |  |
|  | int deleteStu(Player stu[],int n,Player s) /\*从数组中删除指定球员号码的一个元素\*/ |
|  | { |
|  | int i,j; |
|  | for (i=0;i<n;i++) /\*寻找待删除的元素\*/ |
|  | if (equal(pla[i],s,1)) break; /\*如果找到相等元素则退出循环\*/ |
|  | if (i==n) /\*如果找不到待删除的元素\*/ |
|  | { |
|  | printf("This record does not exist!\n"); /\*给出提示信息然后返回\*/ |
|  | return n; |
|  | } |
|  | for (j=i; j<n-1; j++) /\*此处隐含条件为i<n且equal(pla[i],s,1)成立\*/ |
|  | pla[j]=pla[j+1]; /\*通过移动覆盖删除下标为i的元素\*/ |
|  |  |
|  | n--; /\*元素个数减少加1\*/ |
|  | return n; /\*返回现有个数\*/ |
|  | } |