

《计算机程序设计》作业 No-11 及第10次上机

作业内容要点： 结构体和链表

【姓名_____学号_____】

(一)在计算机上编程程序，加上必要的注释。

(二)上机实验，经助教检查通过后，复制源码并记录实验结果，完成报告。

(三)实验报告：记录调试及改错过程；知识点或方法技巧的收获心得。



1、用结构体数组实现学生成绩表。

说明：

1) 结构体类型定义为：

```
struct student{  
    int stunum;           //学号  
    char name[20];        //姓名  
    float examscore;      //考试成绩  
    float labscore;       //实验成绩  
    float totalmark;      //总评成绩  
};
```

2) 在主函数中定义结构体数组，`struct student stutable[10];`

输入如下十个学生的成绩数据，每个学生信息包括 学号、姓名、考试成绩，实验成绩。同时计算每个学生的总评成绩（=考试成绩*60% + 实验成绩*40%）并保存至每个结构体的totalmark。输入格式如下：

71250	李霞	95	82
69753	李友友	88	86
12254	东方亮	87	88
61256	张男	73	85
30258	孙杰	25	88
11260	柯以乐	82	76
33262	谢涛	91	85
29263	叶林	80	75
22483	陈翔	80	76
71525	王子	71	88

- 3) 在主函数中定义一个结构体指针数组，`struct student *parray[10]`；使其每一个指针指向上述结构体数组中的一个元素；按总评成绩从高到低的顺序，对指针数组

输入输出样例：略

(一) 【源码】

```
1. //
2. //  main.c
3. //  121101
4. //
5. //  Created by 李佩哲 on 2021/12/7.
6. //
7.
8. #include <stdio.h>
9. #include <stdlib.h>
10. #include <string.h>
11.
12.
13. typedef struct student{
14.     int stunum;        //学号
15.     char name[20];     //姓名
16.     float examscore;   //考试成绩
17.     float labscore;    //实验成绩
18.     float totalmark;   //总评成绩
19. }STU;
20.
21. int main(int argc, const char * argv[]) {
22.     STU stutable[10];
23.     for (int i=0; i<10; i++) {
24.         scanf("%d%s%f%f",&stutable[i].stunum,stutable[i].name,
&stutable[i].examscore,&stutable[i].labscore);
25.         stutable[i].totalmark=stutable[i].examscore*0.6+stutable[i].labscore*0.4;
26.     }
27.     STU *parray[10];
28.     for(int i=0;i<10;i++)
29.         parray[i]=&stutable[i];
30.     for(int j=0;j<9;j++)
31.         for(int i=0;i<9-j;i++)
32.             if(parray[i]->totalmark<=parray[i+1]->totalmark)
{
```

```

33.             STU *temp=parray[i];
34.             parray[i]=parray[i+1];
35.             parray[i+1]=temp;
36.         }
37.         for(int i=0;i<10;i++)
38.             printf("%d %s:%f\n",parray[i]->stunum,parray[i]-
>name,parray[i]->totalmark);
39.         return 0;
40.     }
41. /*
42. 71250李霞      95      82
43. 69753李友友    88      86
44. 12254东方亮    87      88
45. 61256张男      73      85
46. 30258孙杰      25      88
47. 11260柯以乐    82      76
48. 33262谢涛      91      85
49. 29263叶林      80      75
50. 22483陈翔      80      76
51. 71525王子      71      88
52. */

```

(二) 【运行结果】

```

1. 71250李霞      95      82
2. 69753李友友    88      86
3. 12254东方亮    87      88
4. 61256张男      73      85
5. 30258孙杰      25      88
6. 11260柯以乐    82      76
7. 33262谢涛      91      85
8. 29263叶林      80      75
9. 22483陈翔      80      76
10. 71525王子      71      88
11. 71250 李霞:89.800003
12. 33262 谢涛:88.599998
13. 12254 东方亮:87.400002
14. 69753 李友友:87.199997
15. 11260 柯以乐:79.599998
16. 22483 陈翔:78.400002
17. 29263 叶林:78.000000
18. 71525 王子:77.800003
19. 61256 张男:77.800003
20. 30258 孙杰:50.200001
21. Program ended with exit code: 0

```

(三) 【实验报告】

使用指针数组，指向10个结构体，然后对指针进行操作

2. 用链表实现学生成绩表管理。

接上题。

1) 结构体类型定义修改为：

```
struct student{  
    int stunum;           //学号  
  
    char name[20];       //姓名  
  
    float examscore;     //考试成绩  
  
    float labscore;      //实验成绩  
  
    float totalmark;     //总评成绩  
  
    struct student * next; //下一个结点  
  
};
```

- 2) 编写函数实现建立链表：`struct student * create(int n)`, `n`是学生人数。函数中输入`n`个学生的信息，同时计算总评成绩，按照总评成绩从高到低的方式形成有序链表。返回链表头指针。
- 3) 编写函数 `struct student * delete(struct student * head, int stunum)`, 将学号为`stunum`的结点删除；返回链表头指针。
- 4) 编写函数`struct student * insert(struct student * head)`, 插入一个新的结点到链表中，并保持按总评成绩从高到低有序。返回链表头指针。
- 5) 在主函数中分别调用上述函数，建立链表的10个学生数据同第一题。删除结点时的测试数据可以是现有的学号、也可以是不存在的学号—函数应输出提示未找到并返回原有头指针。新增结点时数据为任意与现有结点不同的值。在主函数中输出每次函数调用后的链表内容。

输入输出样例：略

```
1.  //  
2.  //  main.c  
3.  //  121102  
4.  //  
5.  //  Created by 李佩哲 on 2021/12/8.  
6.  //  
7.
```

```

8. #include <stdio.h>
9. #include <stdlib.h>
10.
11. typedef struct student{
12.     int stunum;           //学号
13.     char name[20];        //姓名
14.     float examscore;      //考试成绩
15.     float labscore;       //实验成绩
16.     float totalmark;      //总评成绩
17.     struct student * next; //下一个结点
18. } STU;
19.
20. STU *create(int n){
21.     STU *head, *node, *end;
22.     head=(STU*)malloc(sizeof(STU));
23.     end=head;
24.     for (int i=0; i<n; i++) {
25.         node=(STU*)malloc(sizeof(STU));
26.         scanf("%d%s%f%f", &node->stunum, node->name, &node->
examscore, &node->labscore);
27.         node->totalmark=0.6*node->examscore+0.4*node->
labscore;
28.         end->next=node;
29.         end=node;
30.     }
31.     end->next=NULL;
32.     for(int i=0; i<n; i++)
33.         for (STU *q=head; (q!=NULL) && (q->next!=NULL) && (q->
next->next!=NULL); q=q->next) {
34.             if ((q==head) && (q->totalmark<=q->next->
totalmark)) {
35.                 STU *p=q->next->next;
36.                 head=q->next;
37.                 q->next->next=q;
38.                 q->next=p;
39.             }
40.             else if (q->next->totalmark<=q->next->next->
totalmark) {
41.                 if (q->next->next==end) {
42.                     end=q->next;
43.                     q->next=q->next->next;
44.                     q->next->next=end;
45.                     end->next=NULL;
46.                 }
47.                 else{
48.                     STU *p1=q->next, *p2=q->next->next->
next;
49.                     q->next=q->next->next;
50.                     q->next->next=p1;
51.                     q->next->next->next=p2;
52.                 }
53.             }

```

```

54.     }
55.     return head;
56. }
57.
58. STU *delete(STU *head,int stunum) {
59.     STU *p=head;
60.     STU *q=p;
61.     for (; (p->stunum!=stunum) && (p!=NULL) && (p->next!=NULL) ; p=p-
        >next)
62.         q=p;
63.     if (p->stunum==stunum) {
64.         q->next=p->next;
65.         free(p);
66.     }
67.     else printf("NULL Person\n");
68.     return head;
69. }
70.
71. STU *insert(STU *head) {
72.     STU *node, *end=head;
73.     for (; end->next!=NULL; end=end->next);
74.     node=(STU*)malloc(sizeof(STU));
75.     scanf("%d%s%f%f",&node->stunum,node->name,&node-
        >examscore,&node->labscore);
76.     node->totalmark=0.6*node->examscore+0.4*node->labscore;
77.     end->next=node;
78.     end=node;
79.     end->next=NULL;
80.     for(int i=0;i<10;i++)
81.         for (STU *q=head; (q!=NULL) && (q->next!=NULL) && (q-
            >next->next!=NULL); q=q->next) {
82.             if ((q==head) && (q->totalmark<=q->next-
                >totalmark)) {
83.                 STU *p=q->next->next;
84.                 head=q->next;
85.                 q->next->next=q;
86.                 q->next=p;
87.             }
88.             else if (q->next->totalmark<=q->next->next-
                >totalmark) {
89.                 if (q->next->next==end) {
90.                     end=q->next;
91.                     q->next=q->next->next;
92.                     q->next->next=end;
93.                     end->next=NULL;
94.                 }
95.                 else{
96.                     STU *p1=q->next,*p2=q->next->next-
                        >next;
97.                     q->next=q->next->next;
98.                     q->next->next=p1;
99.                     q->next->next->next=p2;
100.                }

```

```

101.         }
102.     }
103.     return head;
104. }
105.
106. int main(int argc, const char * argv[]) {
107.     int stunum;
108.     STU *p=create(10);
109.     printf("Sorted:\n");
110.     for (STU *q=p; q!=NULL; q=q->next)
111.         if(q->stunum!=0)
112.             printf("%d %s:%f\n",q->stunum,q->name,q->
>totalmark);
113.     printf("Delete:");
114.     scanf("%d",&stunum);
115.     p=delete(p, stunum);
116.     printf("Deleted:\n");
117.     for (STU *q=p->next; q!=NULL; q=q->next)
118.         if(q->stunum!=0)
119.             printf("%d %s:%f\n",q->stunum,q->name,q->
>totalmark);
120.     printf("Insert:");
121.     p=insert(p);
122.     printf("Inserted:\n");
123.     for (STU *q=p->next; q!=NULL; q=q->next)
124.         if(q->stunum!=0)
125.             printf("%d %s:%f\n",q->stunum,q->name,q->
>totalmark);
126.     return 0;
127. }
128. /*
129. 71250李霞      95      82
130. 69753李友友    88      86
131. 12254东方亮    87      88
132. 61256张男      73      85
133. 30258孙杰      25      88
134. 11260柯以乐    82      76
135. 33262谢涛      91      85
136. 29263叶林      80      75
137. 22483陈翔      80      76
138. 71525王子      71      88
139. */

```

```

1. 71250李霞      95      82
2. 69753李友友    88      86
3. 12254东方亮    87      88
4. 61256张男      73      85
5. 30258孙杰      25      88
6. 11260柯以乐    82      76

```

```
7. 33262谢涛      91      85
8. 29263叶林      80      75
9. 22483陈翔      80      76
10. 71525王子      71      88
11. Sorted:
12. 71250 李霞:89.800003
13. 33262 谢涛:88.599998
14. 12254 东方亮:87.400002
15. 69753 李友友:87.199997
16. 11260 柯以乐:79.599998
17. 22483 陈翔:78.400002
18. 29263 叶林:78.000000
19. 71525 王子:77.800003
20. 61256 张男:77.800003
21. 30258 孙杰:50.200001
22. Delete:71250
23. Deleted:
24. 33262 谢涛:88.599998
25. 12254 东方亮:87.400002
26. 69753 李友友:87.199997
27. 11260 柯以乐:79.599998
28. 22483 陈翔:78.400002
29. 29263 叶林:78.000000
30. 71525 王子:77.800003
31. 61256 张男:77.800003
32. 30258 孙杰:50.200001
33. Insert:71525王子      71      88
34. Inserted:
35. 33262 谢涛:88.599998
36. 12254 东方亮:87.400002
37. 69753 李友友:87.199997
38. 11260 柯以乐:79.599998
39. 22483 陈翔:78.400002
40. 29263 叶林:78.000000
41. 61256 张男:77.800003
42. 71525 王子:77.800003
43. 71525 王子:77.800003
44. 30258 孙杰:50.200001
45. Program ended with exit code: 0
```

使用链表循环存储10组数据，然后通过遍历的方法来找出、删除元素，并冒泡排序

3 、 用链表实现求两个多项式的和。

说明：

- 1) 一个多项式可以表示为二元组序列 $\{(a_1, e_1), (a_2, e_2), \dots (a_n, e_n)\}$, 其中 a_i 表示第 i 项的系数(非零值), e_i 表示第 i 项的指数。
- 2) 编写函数建立多项式链表实现一个多项式的输入, 按指数从高到低有序, 返回链表的头指针。
- 3) 编写函数实现两个多项式相加, 返回结果多项式链表的头指针。
- 4) 编写函数输出一个多项式的二元组序列。
- 5) 在main函数中分别调用上述函数, 实现输入两个多项式, 求出它们的和并输出结果。
- 6) 输入数据分2行, 每行分别先给出多项式非零项的个数, 再输入每一对非零项系数和指数(假设为绝对值均为不超过10000的整数)。数字间仅以空格分隔。
- 7) 为简化处理, 限定系数与指数都为整数。

链表结点数据结构可定义为：

```
struct PolyNode{
    int a; //系数
    int e; //指数
    PolyNode * next; //指向下一个结点
};
```

输入样例：

4 3 4 -5 2 6 1 -2 0

[注]表示 $3x^4-5x^2+6x-2$

3 5 20 -7 4 3 1

[注]表示 $5x^{20}-7x^4+3x$

输出样例：

5 20 -4 4 -5 2 9 1 -2 0

[注]表示 $5x^{20}-4x^4-5x^2+9x-2$

(一)【源码】

```
22. //
23. //  main.c
24. //  121103
25. //
26. //  Created by 李佩哲 on 2021/12/8.
27. //
28.
29. #include <stdio.h>
30. #include <stdlib.h>
31.
32. typedef struct LinkList{
33.     double coefficient;
34.     int exponent;
35.     struct LinkList *next;
36.     struct LinkList *prior;
37. } SUM;
38.
39. SUM *define(int n){
40.     SUM *head, *node, *end;
41.     head=(SUM*)malloc(sizeof(SUM));
42.     head->prior=NULL;
43.     end=head;
44.     for(int i=1;i<=n;i++){
45.         node=(SUM*)malloc(sizeof(SUM));
46.         scanf("%lf%d",&node->coefficient,&node->exponent);
47.         if(node->coefficient==0)i--;
48.         node->prior=end;
49.         end->next=node;
50.         end=node;
51.     }
52.     end->next=NULL;
53.     return head;
54. }
55.
56. SUM *sort(SUM *head,SUM *head2){
57.     void print(SUM *head);
58.     SUM *p1=head, *p2=head2;
59.     for(;p1->next!=NULL;p1=p1->next);
60.     for(; ; p2=p2->next) {
61.         p1->next=p2;
62.         p1=p2;
63.         if(p2->next==NULL)break;
64.     }
65.     p1->next=NULL;
66.     for(int i=0;i<10;i++)
67.         for (SUM *q=head; (q!=NULL) && (q->next!=NULL) && (q->next->next!=NULL); q=q->next) {
68.             if ((q==head) && (q->exponent<=q->next->exponent)) {
69.                 SUM *p=q->next->next;
```

```

70.             head=q->next;
71.             q->next->next=q;
72.             q->next=p;
73.         }
74.         else if (q->next->exponent<=q->next->next-
>exponent) {
75.             if (q->next->next==p1) {
76.                 p1=q->next;
77.                 q->next=q->next->next;
78.                 q->next->next=p1;
79.                 p1->next=NULL;
80.             }
81.             else{
82.                 SUM *p1=q->next,*p2=q->next->next-
>next;
83.                 q->next=q->next->next;
84.                 q->next->next=p1;
85.                 q->next->next->next=p2;
86.             }
87.         }
88.     }
89.     for (p1=head; (p1!=NULL)&&(p1->next!=NULL); ) {
90.         if (p1->exponent==p1->next->exponent) {
91.             p1->coefficient=p1->coefficient+p1->next-
>coefficient;
92.             SUM *temp=p1->next;
93.             p1->next=p1->next->next;
94.             free(temp);
95.         }
96.         else p1=p1->next;
97.         if(p1->next==NULL)
98.             break;
99.     }
100.    return head;
101.}
102.
103.void print(SUM *head){
104.    int i=0,r=0;
105.    for (SUM *p=head; (p!=NULL); p=p->next) {
106.        if((p!=NULL)&&((int)(p->coefficient)!=0)){
107.            if((p->coefficient>0)&&(i!=0))printf("+");
108.            if(p->exponent==0)printf("%.01f",p-
>coefficient);
109.            else if(p->exponent==1)printf("%.01fx",p-
>coefficient);
110.            else printf("%.01fx^%d",p->coefficient,p-
>exponent);
111.            r=1;
112.        }
113.        i++;
114.    }
115.    if(r==0)printf("0");
116.    printf("\n");

```

```
117.}
118.
119.int main(int argc, const char * argv[]) {
120.    int n1,n2;
121.    scanf("%d",&n1);
122.    SUM *p1=define(n1);
123.    scanf("%d",&n2);
124.    SUM *p2=define(n2);
125.    SUM *p=sort(p1,p2);
126.    print(p);
127.    return 0;
128.}
```

(二)【运行结果】

1. 4 3 4 -5 2 6 1 -2 0
2. 3 5 20 -7 4 3 1
3. $5x^{20}-4x^4-5x^2+9x-2$
4. Program ended with exit code: 0

(三)【实验报告】

先冒泡排序，然后比较前后两项的指数，然后决定是否求和