

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级： 网安2003班**

**学 号： U202012068**

**姓 名： 叶礼亮**

**指导教师： 余林琛**

**报告日期： 2020/10/17**

**网络空间安全学院**

**目 录**

[**1 表达式和标准输入输出实验 1**](#_Toc404837920)

[1.1 实验目的 1](#_Toc404837921)

[1.2 实验内容 1](#_Toc404837922)

[1.3 实验小结 1](#_Toc404837923)

[**2 流程控制实验 2**](#_Toc404837924)

[2.1 实验目的 1](#_Toc404837921)

[2.2 实验内容 1](#_Toc404837922)

[2.3 实验小结 1](#_Toc404837923)

[**3 函数与程序结构实验 3**](#_Toc404837929)

[3.1 实验目的 1](#_Toc404837921)

[3.2 实验内容 1](#_Toc404837922)

[3.3 实验小结 1](#_Toc404837923)

[**4 编译预处理实验 4**](#_Toc404837934)

[4.1 实验目的 1](#_Toc404837921)

[4.2 实验内容 1](#_Toc404837922)

[4.3 实验小结 1](#_Toc404837923)

[**5 数组实验 5**](#_Toc404837938)

[5.1 实验目的 1](#_Toc404837921)

[5.2 实验内容 1](#_Toc404837922)

[5.3 实验小结 1](#_Toc404837923)

[**6 指针实验 6**](#_Toc404837943)

[6.1 实验目的 1](#_Toc404837921)

[6.2 实验内容 1](#_Toc404837922)

[6.3 实验小结 1](#_Toc404837923)

[**7 结构与联合实验 7**](#_Toc404837948)

[7.1 实验目的 1](#_Toc404837921)

[7.2 实验内容 1](#_Toc404837922)

[7.3 实验小结 1](#_Toc404837923)

[**8 文件实验 8**](#_Toc404837953)

[8.1 实验目的 1](#_Toc404837921)

[8.2 实验内容 1](#_Toc404837922)

[8.3 实验小结 1](#_Toc404837923)

[**参考文献 9**](#_Toc404837957)

# 7 结构与联合实验

## 7.1 实验目的

1．通过实验，熟悉和掌握结构的说明和引用、结构的指针、结构数组、以及函数中使用结构的方法。

2．通过实验，掌握动态储存分配函数的用法，掌握自引用结构，单向链表的创建、遍历、结点的增删、查找等操作。

3．了解字段结构和联合的用法。

## 7.2 实验题目及要求

**7.2.1 表达式求值的程序验证题**

设有说明：

char u[]="UVWXYZ";

char v[]="xyz";

struct T{

int x;

char c;

char \*t;

}a[]={{11,ˊAˊ,u},{100, ˊBˊ,v}},\*p=a;

请先自己计算下面表达式的值，然后通过编程计算来加以验证。(各表达式相互无关)

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **表达式** | **计算值** | **验证值** |
| 1 | (++p)->x | 100 | 100 |
| 2 | p++,p->c | B | B |
| 3 | \*p++->t,\*p->t | x | x |
| 4 | \*(++p)->t | x | x |
| 5 | \*++p->t | V | V |
| 6 | ++\*p->t | V | V |

**7.2.2 源程序修改替换题**

给定一批整数，以0作为结束标志且不作为结点，将其建成一个先进先出的链表，先进先出链表的指头指针始终指向最先创建的结点（链头），先建结点指向后建结点，后建结点始终是尾结点。

1. 源程序中存在什么样的错误（先观察执行结果）？对程序进行修改、调试，使之能够正确完成指定任务。

源程序如下：

#include "stdio.h"

#include "stdlib.h"

struct s\_list{

int data; /\* 数据域 \*/

struct s\_list \*next; /\* 指针域 \*/

} ;

void create\_list (struct s\_list \*headp,int \*p);

void main(void)

{

struct s\_list \*head=NULL,\*p;

int s[]={1,2,3,4,5,6,7,8,0}; /\* 0为结束标记 \*/

create\_list(head,s); /\* 创建新链表 \*/

p=head; /\*遍历指针p指向链头 \*/

while(p){

printf("%d\t",p->data); /\* 输出数据域的值 \*/

p=p->next; /\*遍历指针p指向下一结点 \*/

}

printf("\n");

}

void create\_list(struct s\_list \*headp,int \*p)

{

struct s\_list \* loc\_head=NULL,\*tail;

if(p[0]==0) /\* 相当于\*p==0 \*/

;

else { /\* loc\_head指向动态分配的第一个结点 \*/

loc\_head=(struct s\_list \*)malloc(sizeof(struct s\_list));

loc\_head->data=\*p++; /\* 对数据域赋值 \*/

tail=loc\_head; /\* tail指向第一个结点 \*/

while(\*p){ /\* tail所指结点的指针域指向动态创建的结点 \*/

tail->next=(struct s\_list \*)malloc(sizeof(struct s\_list));

tail=tail->next; /\* tail指向新创建的结点 \*/

tail->data=\*p++; /\* 向新创建的结点的数据域赋值 \*/

}

tail->next=NULL; /\* 对指针域赋NULL值 \*/

}

headp=loc\_head; /\* 使头指针headp指向新创建的链表 \*/

}

**解答：**

替换后的程序如下所示：

#include "stdio.h"

#include "stdlib.h"

struct s\_list

{

int data; /\* 数据域 \*/

struct s\_list \*next; /\* 指针域 \*/

};

void create\_list(struct s\_list \*\*headp, int \*p); //传入二级指针

void main(void)

{

struct s\_list \*head = NULL, \*p;

int s[] = {1, 2, 3, 4, 5, 6, 7, 8, 0}; /\* 0为结束标记 \*/

create\_list(&head, s); /\* 创建新链表 \*/

//传入head的地址，否则复制型的传参不会改变head的值

p = head; /\*遍历指针p指向链头 \*/

while (p)

{

printf("%d\t", p->data); /\* 输出数据域的值 \*/

p = p->next; /\*遍历指针p指向下一结点 \*/

}

printf("\n");

}

void create\_list(struct s\_list \*\*headp, int \*p) //传入二级指针

{

struct s\_list \*loc\_head = NULL, \*tail;

if (p[0] == 0) /\* 相当于\*p==0 \*/

;

else

{ /\* loc\_head指向动态分配的第一个结点 \*/

loc\_head = (struct s\_list \*)malloc(sizeof(struct s\_list));

loc\_head->data = \*p++; /\* 对数据域赋值 \*/

tail = loc\_head; /\* tail指向第一个结点 \*/

while (\*p)

{ /\* tail所指结点的指针域指向动态创建的结点 \*/

tail->next = (struct s\_list \*)malloc(sizeof(struct s\_list));

tail = tail->next; /\* tail指向新创建的结点 \*/

tail->data = \*p++; /\* 向新创建的结点的数据域赋值 \*/

}

tail->next = NULL; /\* 对指针域赋NULL值 \*/

}

\*headp = loc\_head; // /\* 使头指针headp指向新创建的链表 \*/

}

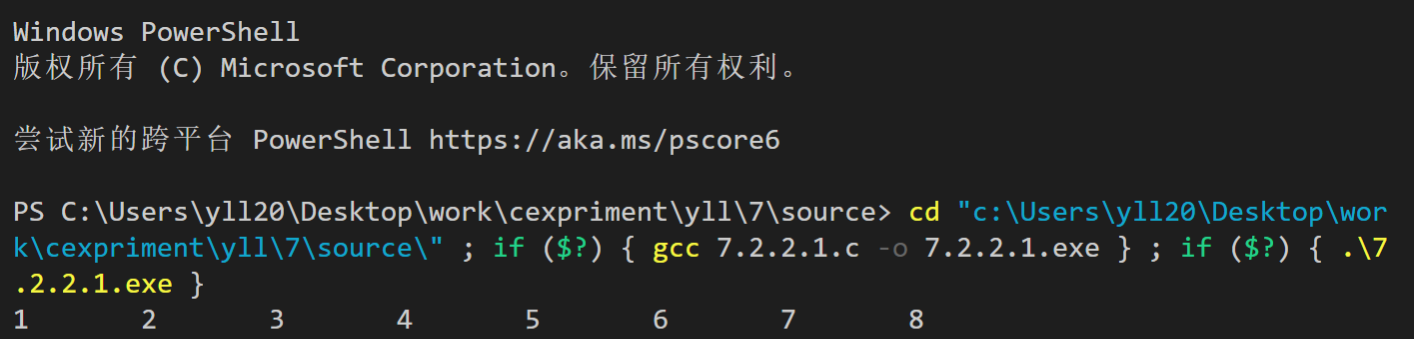


图7-1 程序替换题1的运行结果

（2）修改替换create\_list函数，将其建成一个后进先出的链表，后进先出链表的头指针始终指向最后创建的结点（链头），后建结点指向先建结点，先建结点始终是尾结点。

**解答：**

替换后的程序如下所示：

#include "stdio.h"

#include "stdlib.h"

struct s\_list

{

int data; /\* 数据域 \*/

struct s\_list \*next; /\* 指针域 \*/

};

void create\_list(struct s\_list \*\*headp, int \*p);

void main(void)

{

struct s\_list \*head = NULL, \*p;

int s[] = {1, 2, 3, 4, 5, 6, 7, 8, 0}; /\* 0为结束标记 \*/

create\_list(&head, s); /\* 创建新链表 \*/

//传入head的地址，否则复制型的传参不会改变head的值

p = head; /\*遍历指针p指向链头 \*/

while (p)

{

printf("%d\t", p->data); /\* 输出数据域的值 \*/

p = p->next; /\*遍历指针p指向下一结点 \*/

}

printf("\n");

}

void create\_list(struct s\_list \*\*headp, int \*p)

{

struct s\_list \*loc\_head = NULL, \*tail = NULL;

int len = 0;

int \*tmp = p;

while (\*p++)

{

len++;

}

p -= 2;

for (; len > 0; len--)

{

loc\_head = (struct s\_list \*)malloc(sizeof(struct s\_list));

loc\_head->data = \*tmp++; /\* 对数据域赋值 \*/

loc\_head->next = tail;

tail = loc\_head;

}

\*headp = loc\_head; /\* 使头指针headp指向新创建的链表 \*/

}

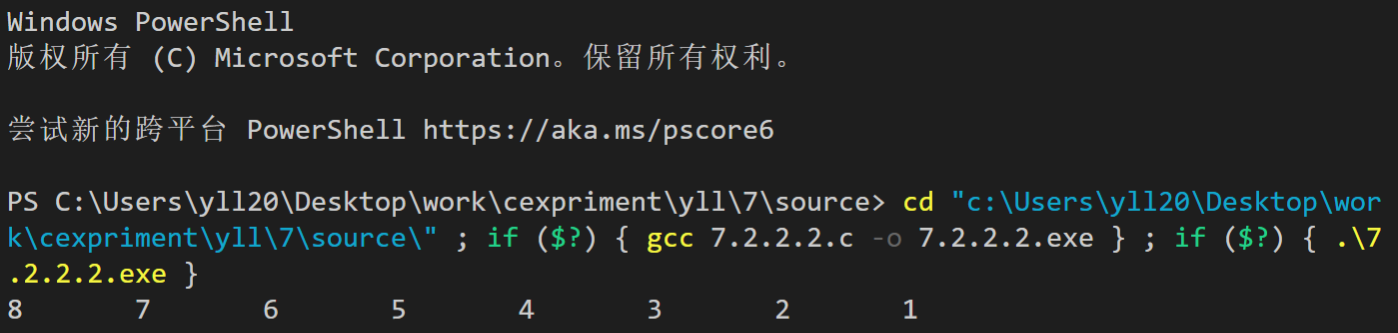


图7-2 程序替换题1的运行结果

**7.2.3 程序设计题**

（1）设计一个字段结构struct bits，它将一个8位无符号字节从最低位向最高位声明为8个字段，各字段依次为bit0, bit1, …, bit7，且bit0的优先级最高。同时设计8个函数，第i个函数以biti(i=0,1,2,…,7)为参数，并且在函数体内输出biti的值。将8个函数的名字存入一个函数指针数组p\_fun。如果bit0为1，调用p\_fun[0]指向的函数。如果struct bits中有多位为1，则根据优先级从高到低依次调用函数指针数组p\_fun中相应元素指向的函数。8个函数中的第0个函数可以设计为：

void f0(struct bits b)

{

Printf(“the function %d is called!\n”,b);

}

**解答：**

1. 算法流程如图7-3所示。

图7-3 程序设计题1的程序流程图

2）源程序清单

#include <stdio.h>

struct bits

{

unsigned short int bit0 : 1;

unsigned short int bit1 : 1;

unsigned short int bit2 : 1;

unsigned short int bit3 : 1;

unsigned short int bit4 : 1;

unsigned short int bit5 : 1;

unsigned short int bit6 : 1;

unsigned short int bit7 : 1;

} a;

void f0(int tmp)

{

printf("the function % d is called !\n", tmp);

}

void f1(int tmp)

{

printf("the function % d is called !\n", tmp);

}

void f2(int tmp)

{

printf("the function % d is called !\n", tmp);

}

void f3(int tmp)

{

printf("the function % d is called !\n", tmp);

}

void f4(int tmp)

{

printf("the function % d is called !\n", tmp);

}

void f5(int tmp)

{

printf("the function % d is called !\n", tmp);

}

void f6(int tmp)

{

printf("the function % d is called !\n", tmp);

}

void f7(int tmp)

{

printf("the function % d is called !\n", tmp);

}

int main()

{

void (\*p[8])(int) = {f0, f1, f2, f3, f4, f5, f6, f7};

unsigned short int n = 15;

a.bit0 = n & 0x1;

n >>= 1;

a.bit1 = n & 0x1;

n >>= 1;

a.bit2 = n & 0x1;

n >>= 1;

a.bit3 = n & 0x1;

n >>= 1;

a.bit4 = n & 0x1;

n >>= 1;

a.bit5 = n & 0x1;

n >>= 1;

a.bit6 = n & 0x1;

n >>= 1;

a.bit7 = n & 0x1;

if (a.bit0)

p[0](0);

if (a.bit1)

p[1](1);

if (a.bit2)

p[2](2);

if (a.bit3)

p[3](3);

if (a.bit4)

p[4](4);

if (a.bit5)

p[5](5);

if (a.bit6)

p[6](6);

if (a.bit7)

p[7](7);

return 0;

}

3）测试

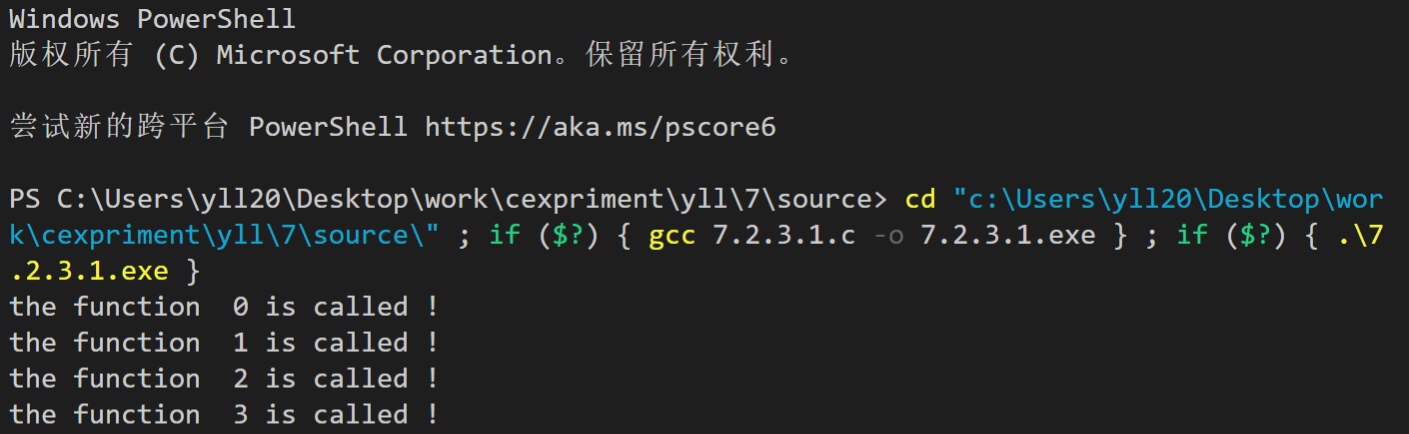


图7-4 程序设计题1的运行结果图

（2）用单向链表建立一张班级成绩单，包括每个学生的学号、姓名、英语、高等数学、普通物理、C语言程序设计四门课程的成绩。用函数编程实现下列功能：

(1) 输入每个学生的各项信息。

(2) 输出每个学生的各项信息。

(3) 修改指定学生的指定数据项的内容。

(4) 统计每个同学的平均成绩（保留2位小数）。

(5) 输出各位同学的学号、姓名、四门课程的总成绩和平均成绩。

**解答：**

1） 算法流程如图7-5所示。

图7-5 程序设计题2的程序流程图

2）源程序清单

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct data

{

char uid[16];

char name[16];

int eng;

int math;

int phy;

int c\_lang;

int all;

double aver;

struct data \*next;

} data;

void creat(data \*\*head\_p)

{

data \*tail = NULL;

\*head\_p = (data \*)malloc(sizeof(data));

tail = \*head\_p;

data \*p = NULL;

printf("please input uid,name|eng,math,phy and c\_lang\n");

while (scanf("%s %s %d %d %d %d", tail->uid, tail->name, &tail->eng, &tail->math, &tail->phy, &tail->c\_lang) > 0)

{

tail->all = tail->c\_lang + tail->eng + tail->math + tail->phy;

tail->aver = ((double)tail->all) / 4.0;

tail->next = (data \*)malloc(sizeof(data));

p = tail;

tail = tail->next;

printf("please input uid,name|eng,math,phy and c\_lang\n");

}

free(tail);

p->next = NULL;

}

void edit(data \*head\_p)

{

printf("please input the uid you want to edit\n");

char tmp[16];

scanf("%s", tmp);

data \*p = head\_p;

while (1)

{

if (strcmp(p->uid, tmp) == 0)

{

printf("\

which data do you want to edit?\n\

1.name\n\

2.eng\n\

3.math\n\

4.phy\n\

5.c\_lang\n\

");

int cmd;

scanf("%d", &cmd);

if (cmd == 1)

{

printf("input the new name:");

scanf("%s", p->name);

break;

}

if (cmd == 2)

{

printf("input the new eng:");

scanf("%s", p->eng);

}

if (cmd == 3)

{

printf("input the new math:");

scanf("%s", p->math);

}

if (cmd == 4)

{

printf("input the new phy:");

scanf("%s", p->phy);

}

if (cmd == 5)

{

printf("input the new c\_lang:");

scanf("%s", p->c\_lang);

}

p->all = p->c\_lang + p->eng + p->math + p->phy;

p->aver = ((double)p->all) / 4.0;

break;

}

p = p->next;

if (p == NULL)

{

printf("can't find this student!\n");

break;

}

}

}

void output\_all(data \*head\_p)

{

data \*p = head\_p;

while (1)

{

printf("uid:%s,name:%s,eng:%d,math:%d,phy:%d,c\_lang:%d\n", p->uid, p->name, p->eng, p->math, p->phy, p->c\_lang);

p = p->next;

if (p == NULL)

break;

}

}

void output\_score(data \*head\_p)

{

data \*p = head\_p;

while (1)

{

printf("uid:%s,name:%s,all:%d,aver:%.2lf\n", p->uid, p->name, p->all, p->aver);

p = p->next;

if (p == NULL)

break;

}

}

int main()

{

data \*head;

int cmd;

printf("\

choose the function:\n \

1.creat all\n \

2.output all\n \

3.edit data\n \

4.output everyone's average\n\

");

while (scanf("%d", &cmd) > 0)

{

if (cmd == 1)

{

creat(&head);

}

if (cmd == 2)

{

output\_all(head);

}

if (cmd == 3)

{

edit(head);

}

if (cmd == 4)

{

output\_score(head);

}

printf("\

choose the function:\n \

1.creat all\n \

2.output all\n \

3.edit data\n \

4.output everyone's average\n\

");

}

return 0;

}

3）测试

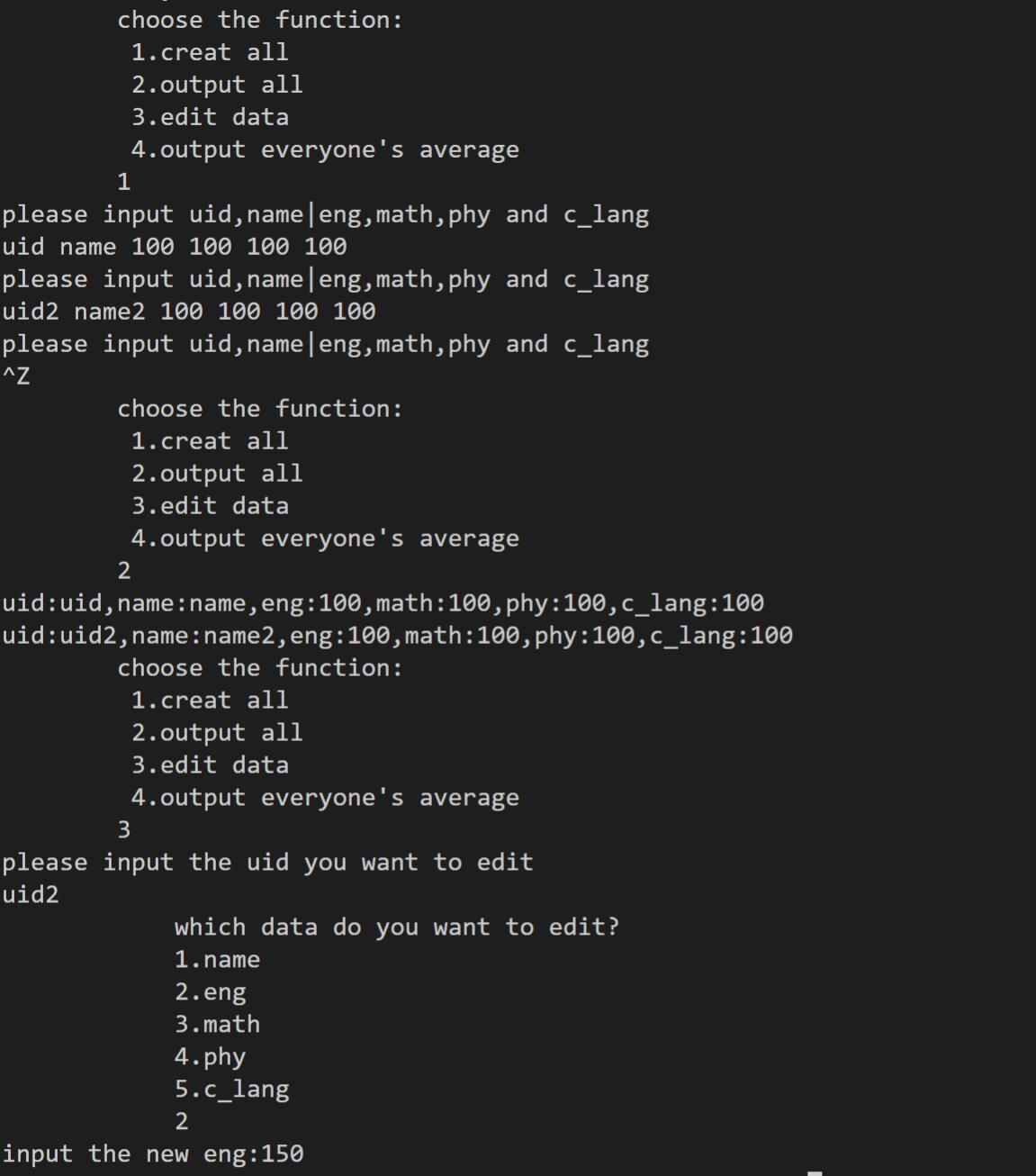


图7-6 程序设计题2的运行结果图

**7.2.4 选做题**

（1）对编程设计题第（2）题的程序，增加按照平均成绩进行升序排序的函数，写出用交换结点数据域的方法升序排序的函数，排序可用选择法或冒泡法。

**解答：**

1） 算法流程如图7-5所示。

图7-5 选做题1的程序流程图

2）源程序清单

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct scores

{

char card\_num[20];

char name[20];

int math;

int english;

int physics;

int c\_lan;

struct scores \*next;

};

void newstudent(struct scores \*\*head);

void print\_info(struct scores \*head);

void print\_average(struct scores \*head);

void change\_info(struct scores \*head);

struct scores \*find\_student(struct scores \*head, char \*target);

void print\_all(struct scores \*head);

void sortdata(struct scores \*head); //添加排序函数

int main(void)

{

int choice;

int flag;

struct scores \*head = NULL, \*\*tail = &head;

do

{

flag = 0;

printf("Please choose:\n");

printf("1. add student\n");

printf("2. print information\n");

printf("3. print average scores\n");

printf("4. change information\n");

printf("5. print information and average scores\n");

printf("6. sort data\n");

printf("7. quit\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

newstudent(tail);

flag = 1;

break;

case 2:

print\_info(head);

break;

case 3:

print\_average(head);

break;

case 4:

change\_info(head);

break;

case 5:

print\_all(head);

break;

case 6:

sortdata(head);

break;

default:

return 0;

}

if (head == NULL)

head = \*tail;

if (flag)

tail = &(\*tail)->next;

} while (choice != 7);

return 0;

}

void newstudent(struct scores \*\*head)

{

struct scores \*p = (struct scores \*)malloc(sizeof(struct scores));

printf("Please enter the card number of student:\n");

scanf("%s", p->card\_num);

printf("Please enter the student's name:\n");

scanf("%s", p->name);

printf("Please enter the score of math:\n");

scanf("%d", &p->math);

printf("Please enter the score of English:\n");

scanf("%d", &p->english);

printf("Please enter the score of physics:\n");

scanf("%d", &p->physics);

printf("Please enter the score of c\_language:\n");

scanf("%d", &p->c\_lan);

p->next = NULL;

\*head = p;

}

void print\_info(struct scores \*head)

{

while (head)

{

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n\n", head->c\_lan);

head = head->next;

}

}

void print\_average(struct scores \*head)

{

int sum;

while (head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("The average score of %s is %.2f\n", head->name, sum / 4.0);

head = head->next;

}

printf("\n");

}

void change\_info(struct scores \*head)

{

char target[10];

int choice;

int data;

printf("Please enter the name of the wanted student:\n");

scanf("%s", target);

struct scores \*temp = find\_student(head, target);

if (temp)

{

printf("Please choose the subject you want to change:\n");

printf("1. Math\n");

printf("2. Physics\n");

printf("3. English\n");

printf("4. c language\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

printf("Please enter the new score of math:\n");

scanf("%d", &data);

temp->math = data;

break;

case 2:

printf("Please enter the new score of physics:\n");

scanf("%d", &data);

temp->physics = data;

break;

case 3:

printf("Please enter the new score of English:\n");

scanf("%d", &data);

temp->english = data;

break;

case 4:

printf("Please enter the new score of c language:\n");

scanf("%d", &data);

temp->c\_lan = data;

break;

}

}

else

printf("Student not found\n");

}

struct scores \*find\_student(struct scores \*head, char \*target)

{

while (head && strcmp(head->name, target))

head = head->next;

return head;

}

void print\_all(struct scores \*head)

{

int sum = 0;

while (head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n", head->c\_lan);

printf("sum: %d\n", sum);

printf("The average score of %s is %.2f\n\n", head->name, sum / 4.0);

head = head->next;

}

}

void sortdata(struct scores \*head)

{

int flag = 1;

struct scores \*p = head;

struct scores \*ptr, \*q, temp;

int sum1, sum2;

while (flag)

{

q = p;

flag = 0;

while (q->next != NULL)

{

sum1 = sum2 = 0;

sum1 += q->c\_lan;

sum1 += q->english;

sum1 += q->math;

sum1 += q->physics;

sum2 += q->next->c\_lan;

sum2 += q->next->english;

sum2 += q->next->math;

sum2 += q->next->physics;

if (sum1 > sum2)

{

flag = 1;

temp = \*q->next;

ptr = q->next->next;

\*q->next = \*q;

q->next->next = ptr;

ptr = q->next;

\*q = temp;

q->next = ptr;

}

q = q->next;

}

}

}

3）

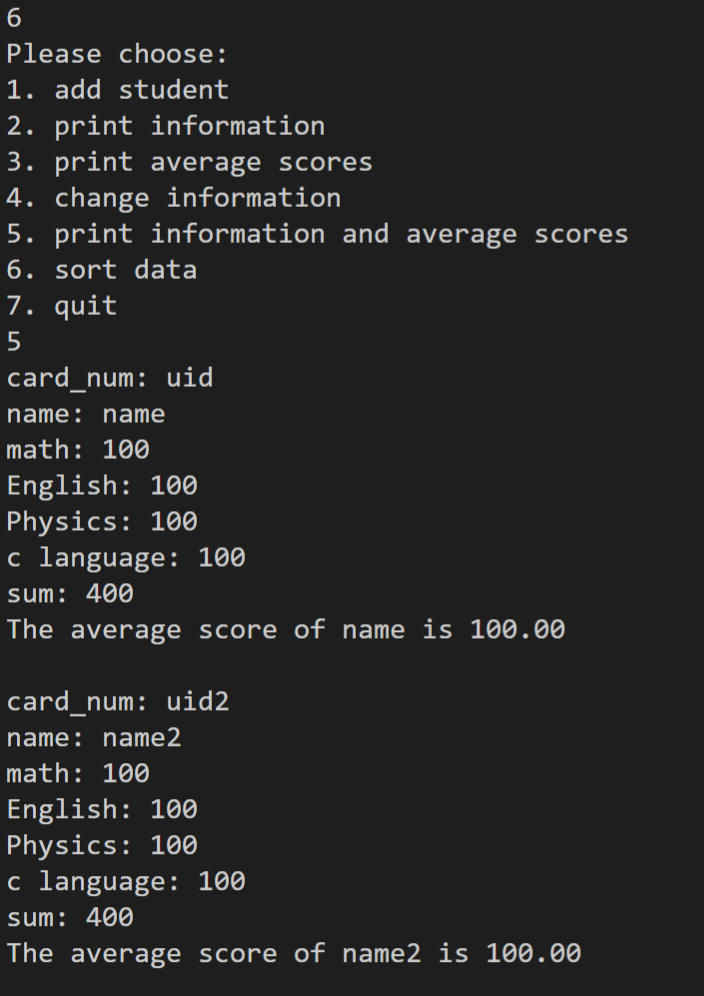


图7-7 选做题1的运行结果图

（2）对选做题第（1）题，进一步写出用交换结点指针域的方法升序排序的函数。

**解答：**

       1） 算法流程如图7-8所示。

图7-8 选做题2的程序流程图

2）源程序清单

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct stu

{

char num[16];

char name[21];

float E;

float M;

float P;

float C;

struct stu \*next;

} Stu;

int main(void)

{

int N;

scanf("%d", &N);

Stu \*head = (Stu \*)malloc(sizeof(Stu));

head->num[0] = '\0';

Stu \*temp = head;

for (int i = 0; i < N; i++)

{

temp->next = (Stu \*)malloc(sizeof(Stu));

temp = temp->next;

scanf("%s %s %f %f %f %f", (char \*)temp->num, (char \*)temp->name, &temp->E, &temp->M, &temp->P, &temp->C);

}

printf("please input uid,name|eng,math,phy and c\_lang\n");

temp = head;

for (int i = 0; i < N; i++)

{

temp = temp->next;

printf("%-15s%-20s%-10.2f%-10.2f%-10.2f%-10.2f\n", temp->num, temp->name, temp->E, temp->M, temp->P, temp->C);

}

int M;

char num[16], sub[20];

float score;

scanf("%d", &M);

for (int i = 0; i < M; i++)

{

scanf("%s %s %f", (char \*)num, (char \*)sub, &score);

for (temp = head->next; strcmp(num, temp->num) != 0; temp = temp->next)

;

switch (sub[0])

{

case 'E':

temp->E = score;

break;

case 'M':

temp->M = score;

break;

case 'P':

temp->P = score;

break;

case 'C':

temp->C = score;

break;

}

}

printf("\nAlter:\nID Name English Math Physics C \n");

temp = head;

for (int i = 0; i < N; i++)

{

temp = temp->next;

printf("%-15s%-20s%-10.2f%-10.2f%-10.2f%-10.2f\n", temp->num, temp->name, temp->E, temp->M, temp->P, temp->C);

}

printf("\nSumAndAvg:\nID Name SUM AVG \n");

temp = head;

float sum;

for (int i = 0; i < N; i++)

{

temp = temp->next;

sum = (temp->E + temp->M + temp->P + temp->C);

printf("%-15s%-20s%-10.2f%-10.2f\n", temp->num, temp->name, sum, sum / 4.0);

}

temp = head;

Stu \*p[N];

for (int i = 0; i < N; i++)

temp = head->next, p[i] = temp;

int tt1, tt2;

for (tt1 = 0; tt1 < N - 1; tt1++)

for (tt2 = tt1 + 1; tt2 < N; tt2++)

if (((p[tt1])->E + (p[tt1])->M + (p[tt1])->P + (p[tt1])->C) > ((p[tt2])->E + (p[tt2])->M + (p[tt2])->P + (p[tt2])->C))

temp = p[tt1];

p[tt1] = p[tt2], p[tt2] = temp;

printf("\nSort:\nID Name AVG \n");

temp = head;

for (int i = 0; i < N; i++)

{

temp = temp->next;

sum = (temp->E + temp->M + temp->P + temp->C);

printf("%-15s%-20s%-10.2f\n", temp->num, temp->name, sum / 4.0);

}

putchar('\n');

return 0;

}

3）测试

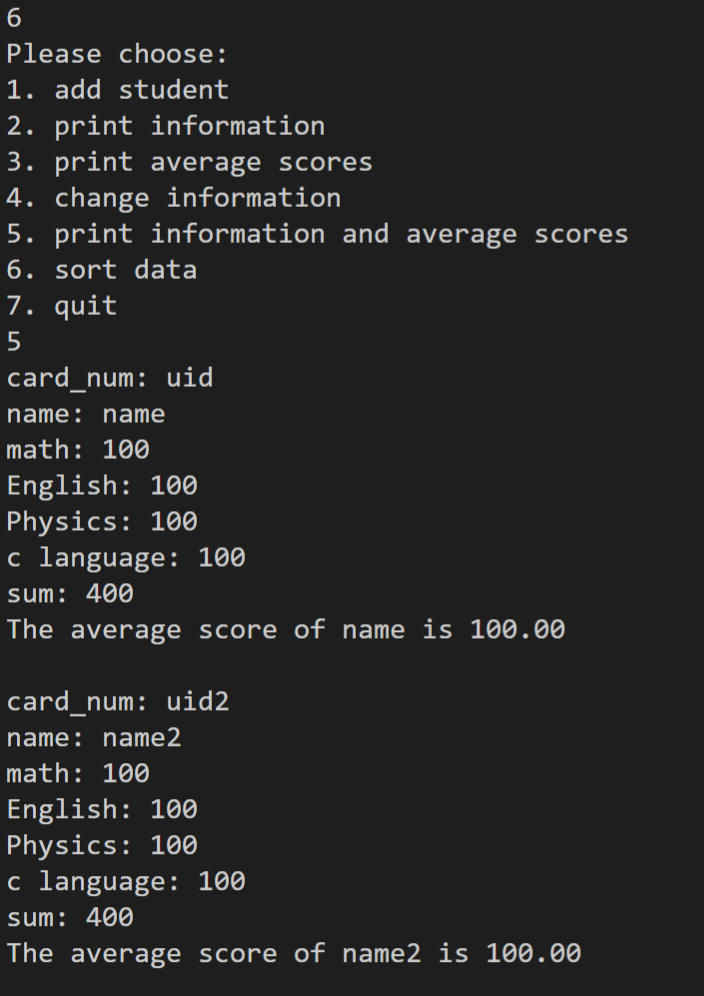


图7-9 选做题2的运行结果图

（3）采用双向链表重做编程设计题第（2）题。

**解答：**

       1） 算法流程如图7-10所示。

图7-10 选做题3的程序流程图

2）源程序清单

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct scores

{

char card\_num[20];

char name[10];

int math;

int english;

int physics;

int c\_lan;

struct scores \*next;

struct scores \*prior;

};

void newstudent(struct scores \*\*head);

void print\_info(struct scores \*head);

void print\_average(struct scores \*head);

void change\_info(struct scores \*head);

struct scores \*find\_student(struct scores \*head, char \*target);

void print\_all(struct scores \*head);

int main(void)

{

int choice;

int flag;

struct scores \*head = NULL, \*\*tail = &head;

do

{

flag = 0;

printf("Please choose:\n");

printf("1. add student\n");

printf("2. print information\n");

printf("3. print average scores\n");

printf("4. change information\n");

printf("5. print information and average scores\n");

printf("6. quit\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

newstudent(tail);

flag = 1;

break;

case 2:

print\_info(head);

break;

case 3:

print\_average(head);

break;

case 4:

change\_info(head);

break;

case 5:

print\_all(head);

break;

default:

return 0;

}

if (head->next == NULL)

tail = &head;

if (head->next != NULL && flag)

tail = &(head->next);

} while (choice != 6);

return 0;

}

void newstudent(struct scores \*\*head)

{

struct scores \*p = (struct scores \*)malloc(sizeof(struct scores));

printf("Please enter the card number of student:\n");

scanf("%s", p->card\_num);

printf("Please enter the student's name:\n");

scanf("%s", p->name);

printf("Please enter the score of math:\n");

scanf("%d", &p->math);

printf("Please enter the score of English:\n");

scanf("%d", &p->english);

printf("Please enter the score of physics:\n");

scanf("%d", &p->physics);

printf("Please enter the score of c\_language:\n");

scanf("%d", &p->c\_lan);

if (\*head == NULL)

{

\*head = p;

p->next = NULL;

p->prior = NULL;

}

else

{

(\*head)->next = p;

p->next = NULL;

p->prior = (\*head);

}

}

void print\_info(struct scores \*head)

{

while (head)

{

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n\n", head->c\_lan);

head = head->next;

}

}

void print\_average(struct scores \*head)

{

int sum = 0;

while (head)

{

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("The average score of %s is %.2f\n", head->name, sum / 4.0);

head = head->next;

}

}

void change\_info(struct scores \*head)

{

char target[10];

int choice;

int data;

printf("Please enter the name of the wanted student:\n");

scanf("%s", target);

struct scores \*temp = find\_student(head, target);

if (temp)

{

printf("Please choose the subject you want to change:\n");

printf("1. Math\n");

printf("2. Physics\n");

printf("3. English\n");

printf("4. c language\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

printf("Please enter the new score of math:\n");

scanf("%d", &data);

temp->math = data;

break;

case 2:

printf("Please enter the new score of physics:\n");

scanf("%d", &data);

temp->physics = data;

break;

case 3:

printf("Please enter the new score of English:\n");

scanf("%d", &data);

temp->english = data;

break;

case 4:

printf("Please enter the new score of c language:\n");

scanf("%d", &data);

temp->c\_lan = data;

break;

}

}

else

printf("Student not found\n");

}

struct scores \*find\_student(struct scores \*head, char \*target)

{

while (head && strcmp(head->name, target))

head = head->next;

return head;

}

void print\_all(struct scores \*head)

{

int sum = 0;

while (head)

{

sum = 0;

sum += head->english;

sum += head->math;

sum += head->physics;

sum += head->c\_lan;

printf("card\_num: %s\n", head->card\_num);

printf("name: %s\n", head->name);

printf("math: %d\n", head->math);

printf("English: %d\n", head->english);

printf("Physics: %d\n", head->physics);

printf("c language: %d\n", head->c\_lan);

printf("sum: %d\n", sum);

printf("The average score of %s is %.2f\n\n", head->name, sum / 4.0);

head = head->next;

}

}

3）测试

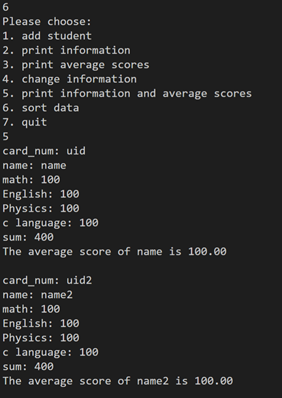


图7-11 选做题3的运行结果图

## 7.4 实验小结

实验中体会到了链表存储结构的方便性，但是也发现了链表的一些局限和自己的问题。

参考文献

[1] 曹计昌,卢萍,李开. C语言程序设计,北京： 科学出版社,2013

[2] 李开,卢萍,曹计昌. C语言实验与课程设计, 北京：科学出版社,2011