习题

185-5

```c
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>

typedef struct
{
    int id;
    char name[32];
    int scores[4];
} Stu;

enum subjects { Chinese, Math, English, All };

static int input(Stu* students, Stu*stupt[], int n, int cnt)
{
    int i;
    for (i = cnt; i < n; i++)
    {
        scanf("%d", &students[i].id);
        if (students[i].id > 0)
        {
            scanf("%s%d%d%d", students[i].name, &students[i].scores[0], &students[i].scores[1], &students[i].scores[2]);
            students[i].scores[3] = students[i].scores[0] + students[i].scores[1] + students[i].scores[2];
            stupt[i] = students + i;
            fflush(stdin);
        }
        else
        {
            cnt = i;
            break;
        }
    }
    if (i == n)
    {
        cnt = n;
        printf("overload\n");
    }
    return cnt;
}
static void addScores(Stu* students, int n, int * total_scores)
{
    int i, j;
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < 4; j++)
        {
            total_scores[j] += students[i].scores[j];
        }
    }
```

```c
}
static void findByName(char* name, Stu* students, int n)
{
    int i, flag = 0;
    Stu tmp;
    for (i = 0; i < n; i++)
    {
        if (strcmp(students[i].name, name) == 0)
        {
            tmp = students[i];
            if (flag == 0)
            {
                printf("%8s%8s%8s%8s%8s%8s\n", "ID", "Name", "Chinese", "Math", "English", "Total");
                flag = 1;
            }
            printf("%8d%8s%8d%8d%8d%8d\n", tmp.id, tmp.name, tmp.scores[0], tmp.scores[1], tmp.scores[2], tmp.scores[3]);
        }
    }
    if (!flag)
    {
        printf("No such name\n");
    }
}
static void findByID(int id, Stu* students, int n)
{
    int i, flag = 0;
    Stu tmp;
    for (i = 0; i < n; i++)
    {
        if (students[i].id == id)
        {
            tmp = students[i];
            if (flag == 0)
            {
                printf("%8s%8s%8s%8s%8s%8s\n", "ID", "Name", "Chinese", "Math", "English", "Total");
                flag = 1;
            }
            printf("%8d%8s%8d%8d%8d%8d\n", tmp.id, tmp.name, tmp.scores[0], tmp.scores[1], tmp.scores[2], tmp.scores[3]);
        }
    }
    if (!flag)
    {
        printf("No such id\n");
    }
}
static void showAllInfos(Stu* stupt[], int n)
{
    int i;
    Stu* tmp;
    printf("%8s%8s%8s%8s%8s%8s\n", "ID", "Name", "Chinese", "Math", "English", "Total");
    for (i = 0; i < n; i++)
    {
        tmp = stupt[i];
        printf("%8d%8s%8d%8d%8d%8d\n", tmp->id, tmp->name, tmp->scores[0], tmp->scores[1], tmp->scores[2], tmp->scores[3]);
    }
}
static void sort(Stu* stupt[], int n, int subj)
{
    Stu* tmp;
    int i, j;
    for (i = 0; i < n - 1; i++)
    {
        for (j = n - 1; j > i; j--)
        {
            if (stupt[i]->scores[subj] < stupt[j]->scores[subj])
            {
                tmp = stupt[i];
                stupt[i] = stupt[j];
                stupt[j] = tmp;
            }
        }
    }
}

static void showTotalAvg(int* total_scores, int i, int cnt, char** subjects_str)
{
    printf("%8s%8s%8s\n", "subjects", "total", "average");
    printf("%8s%8d%8.2f\n", subjects_str[i], total_scores[i], 1. * total_scores[i] / cnt);
}

//int main185_5()
int main()
{
    Stu students[100];
    Stu* stupt[100];
    int i;
    int selection;
    int cnt = 0;
    char *subjects_str[4] = { "Chinese", "Math", "English", "All" };
    int total_scores[4] = { 0 };
    char input_name[100];
    int input_id;

    while (1)
    {
        fflush(stdin);
        printf("### Class * Management system ###\n");
        printf("### 1. add student info #########\n");
        printf("### 2. total/average scores #####\n");
        printf("### 3. find by student name #####\n");
        printf("### 4. find by student id #######\n");
        printf("### 5. scan all student info ####\n");
        printf("### 6. sort by some score #######\n");
        printf("### 7. exit management system ###\n");
        printf("Plz input the selection: ");

        scanf("%d", &selection);
        switch (selection)
        {
        case 1:
            printf("Plz input the student [id, name, Chi, Math, Eng]\n");
            cnt = input(students, stupt, 100, cnt);
            /*
            123 Li 10 20 30
            234 He 56 67 78
            345 Ma 90 70 50
            x
            */

            addScores(students, cnt, total_scores);
            break;

        case 2:
            printf("\n[total / average]\n");
            //printf("[0]Chinese;\n[1]Math;\n[2]English;\n[3]Total;\n");
            //printf("Plz input the subject index: ");
            //scanf("%d", &i);
            i = 1;
            showTotalAvg(total_scores, i, cnt, subjects_str);
            break;

        case 3:
            printf("\n[find by name]\n");
            //printf("Plz input the student name: ");
            //fflush(stdin);
            //scanf("%s", input_name);
            strcpy(input_name, "Ma");
            findByName(input_name, students, cnt);
            break;

        case 4:
            printf("\n[find by id]\n");
            //printf("Plz input the student id: ");
            //scanf("%d", &input_id);
            input_id = 234;
            findByID(input_id, students, cnt);
            break;

        case 5:
            printf("\n[show all info]\n");
            showAllInfos(stupt, cnt);
            break;

        case 6:
            printf("\n[sorted by scores]\n");
            //printf("[0]Chinese;\n[1]Math;\n[2]English;\n[3]Total;\n");
            //printf("Plz input the subject index:");
            //scanf("%d", &i);
            i = 0;
            sort(stupt, cnt, i);
            showAllInfos(stupt, cnt);
            break;

        case 7:
            printf("\n Thanks for using.\n");
            exit(0);
            break;

        default:
            break;
        }
    }
}

```

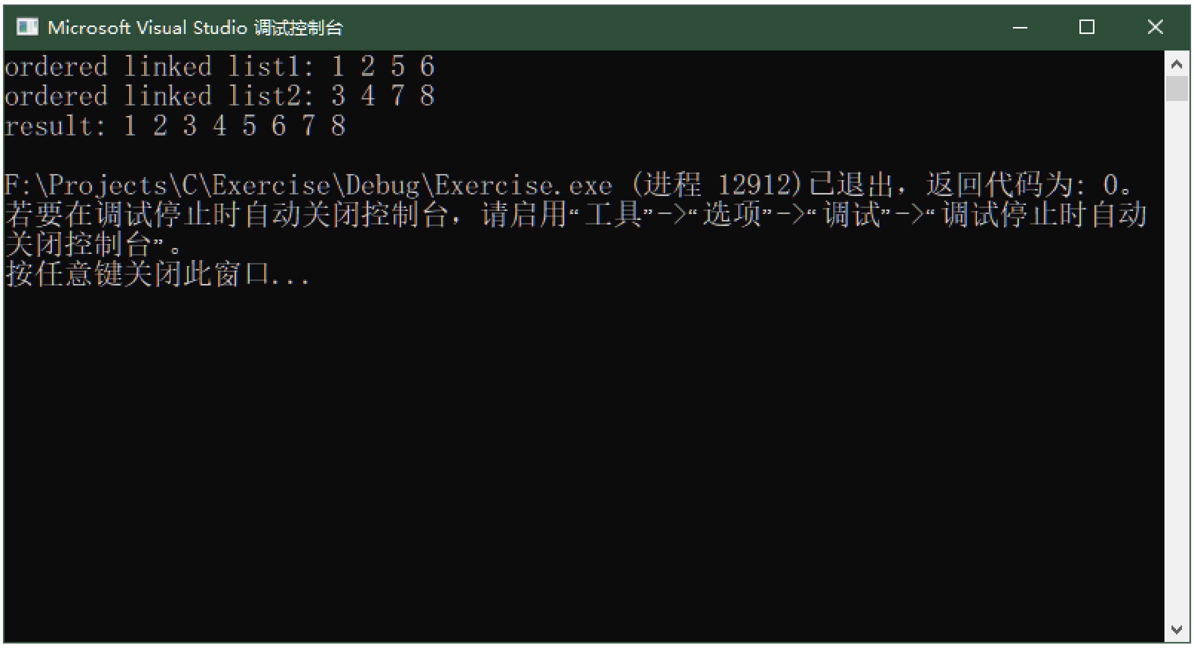
185-6

```
#include<stdio.h>
#include<malloc.h>
#include<time.h>

typedef struct
{
    int number;
    int suit;
}Poker;

Poker cards[52];
char* nums[13] = { "1","2","3","4","5","6","7","8","9","10","J","Q","K" };
char* suits[4] = { "spades", "heart", "club", "diamond" };

void sort186_6(double randval[], Poker*index[])
{
    int i, j;
    double temp1;
    Poker * temp2;
    for (i = 0; i < 51; i++)
    {
        for (j = 51; j > i; j--)
        {
            if (randval[i] > randval[j])
            {
                temp1 = randval[i];
                randval[i] = randval[j];
                randval[j] = temp1;
                temp2 = index[i];
                index[i] = index[j];
                index[j] = temp2;
            }
        }
    }
    1;
    return;
}

void shuffle186_6(Poker *index[])
{
    double randval[52];
    int i, j;
    srand(0);
    for (i = 0; i < 52; i++)
    {
        randval[i] = rand();
```

```c
    }
    sort186_6(randval, index);

    printf("\n\n");
    printf("### shuffled ###\n");
    for (i = 0; i < 52; i++)
    {
        printf("%2s %s\n", nums[index[i]->number], suits[index[i]->suit]);
    }
    return;
}

void deal186_6(Poker *index[], Poker *members[4][13])
{
    int i, j;
    for (i = 0; i < 52; i++)
    {
        members[i % 4][i / 4] = index[i];
    }

    printf("\n\n");
    for (i = 0; i < 4; i++)
    {
        printf("### member %d ###\n", i);
        for (j = 0; j < 13; j++)
        {
            printf("%2s %s\n", nums[members[i][j]->number], suits[members[i][j]->suit]);
        }
    }
    return;
}

int compare186_6(Poker x, Poker y)
{
    if (x.number != y.number)
    {
        return x.number - y.number > 0 ? 1 : -1;
    }
    else
    {
        if (x.suit == y.suit)
        {
            return 0;
        }
        return x.suit - y.suit > 0 ? 1 : -1;
    }
}

//int main186_6()
int main()
{
    Poker *index[52];
    Poker *members[4][13];
    char *cmp[] = { "<", "=", ">" };
    int i, j;
    for (i = 0; i < 52; i++)
    {
        cards[i].number = i / 4;
        cards[i].suit = i % 4;
        index[i] = &cards[i];
    }
    printf("### initialized ###\n");
    for (i = 0; i < 52; i++)
    {
        printf("%2s %s\n", nums[index[i]->number], suits[index[i]->suit]);
    }

    shuffle186_6(index);

    deal186_6(index, members);

    printf("\n\n");
    printf("### compare ###\n");
    i = rand() % 52;
    j = rand() % 52;
    printf("%2s %s %s %2s %s\n", nums[index[i]->number], suits[index[i]->suit], cmp[compare186_6(*(index[i]), *(index[j])) + 1], nums[index[j]->number], suits[index[j]->suit]);

}
```

186-7

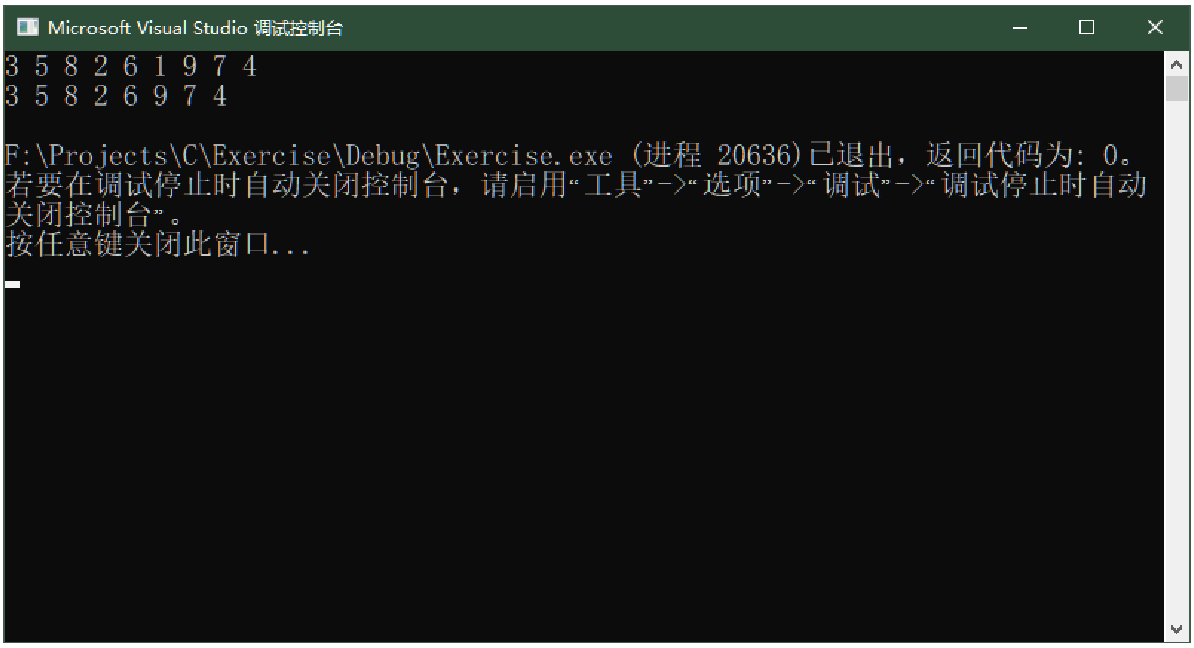
```
#include<stdio.h>
#include<malloc.h>

typedef struct Node
{
    int val;
    struct Node *next;
}Node;

static Node* createLList(int *a, int n)
{
    Node* head = (Node*)malloc(sizeof(Node)), *node, *pre;
    int i;
    head->next = NULL;
    pre = head;
    for (i = 0; i < n; i++)
    {
        node = (Node*)malloc(sizeof(Node));
        node->val = a[i];
        node->next = NULL;
        pre->next = node;
        pre = pre->next;
    }
    return head;

}
static void integrate(Node* head1, Node* head2)
{
    Node *pre1 = head1, *pre2 = head2, *tmp;
    int flag;
    int value;
    while (pre2->next)
    {
        value = pre2->next->val;
        flag = 0;
        while (pre1->next)
        {
            if (value < pre1->next->val)
            {
                tmp = pre2->next->next;
                pre2->next->next = pre1->next;
                pre1->next = pre2->next;
                pre2->next = tmp;
                flag = 1;
                break;
            }
            pre1 = pre1->next;
        }
        if (!flag)
        {
            tmp = pre2->next->next;
            pre2->next->next = pre1->next;
            pre1->next = pre2->next;
            pre2->next = tmp;
        }
    }
}
static void output(Node* head)
{
    while (head->next)
    {
        printf("%d ", head->next->val);
        head = head->next;
    }
}

//int main186_7()
int main()
{
    int arr1[] = { 1,2,5,6 }, arr2[] = { 3,4,7,8 };
    Node* head1 = createLList(arr1, sizeof(arr1) / sizeof(int)), *head2 = createLList(arr2, sizeof(arr2) / sizeof(int));
    printf("ordered linked list1: ");
    output(head1);
    printf("\nordered linked list2: ");
    output(head2);
    integrate(head1, head2);
    printf("\nresult: ");
    output(head1);
    printf("\n");
}
```



```
ordered linked list1: 1 2 5 6
ordered linked list2: 3 4 7 8
result: 1 2 3 4 5 6 7 8

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 12912)已退出, 返回代码为: 0。
若要在调试停止时自动关闭控制台, 请启用"工具"->"选项"->"调试"->"调试停止时自动
关闭控制台"。
按任意键关闭此窗口...
```

186-11

```c
#include<stdio.h>
#include<malloc.h>

typedef struct Node
{
    int val;
    struct Node *next;
}Node;

static Node* createLList(int *a, int n)
{
    Node *head = NULL, *node, *pre;
    int i;
    head = (Node*)malloc(sizeof(Node));
    head->next = NULL;
    pre = head;
    for (i = 0; i < n; i++)
    {
        node = (Node*)malloc(sizeof(Node));
        node->val = a[i];
        node->next = NULL;
        pre->next = node;
        pre = pre->next;
    }
    return head;
}

static void removeMinimum(Node *head)
{
    Node *pre = head, *min_node, *tmp;
    int min_val = pre->next->val;
    min_node = pre;
    pre = pre->next;

    while (pre->next)
    {
        if (pre->next->val < min_val)
        {
            min_node = pre;
            min_val = pre->next->val;
        }
        pre = pre->next;
    }
    tmp = min_node->next;
    min_node->next = tmp->next;
    tmp->next = NULL;
}

static void outputLList(Node* p)
{
    while (p->next)
    {
        printf("%d ", p->next->val);
        p = p->next;
    }
    printf("\n");
}
int main()
{
    int arr[] = { 3,5,8,2,6,1,9,7,4 };
    Node *head = createLList(arr, sizeof(arr) / sizeof(int));
    outputLList(head);
    removeMinimum(head);
    outputLList(head);
}
```

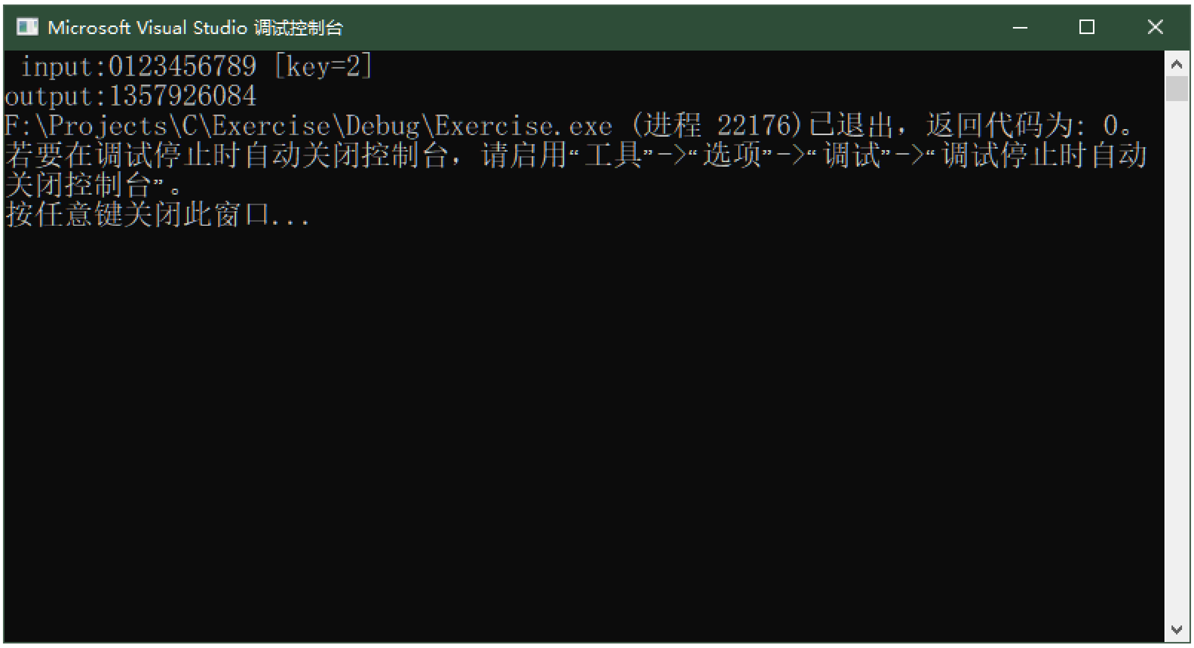


```
3 5 8 2 6 1 9 7 4
3 5 8 2 6 9 7 4

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 20636)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口...
```

186-13

```
#include<stdio.h>
#include<malloc.h>
#define N 10

typedef struct Text
{
    char val;
    struct Text *next;
}Text;

static Text* createLList(char *source, int n)
{
    Text *head = NULL, *node, *pre = NULL;

    int i;
    for (i = 0; i < n; i++)
    {
        node = (Text*)malloc(sizeof(Text));
        node->val = source[i];
        node->next = NULL;
        if (!head)
        {
            head = pre = node;
        }
        else
        {
            pre->next = node;
            pre = pre->next;
        }
    }
    pre->next = head;
    return head;
}

static char* generateCipher(char *source, int key, int n)
{
    Text* head = createLList(source, n);
    Text *pre = NULL, *cur = head;
    char *res = (char*)malloc(sizeof(char) * (n + 1));
    int i, j;
    while (cur->next != head)
    {
        cur = cur->next;
    }
    pre = cur;
    cur = head;
    for (i = 0; i < n; i++)
    {
        j = key;
        while (--j)
        {
            pre = cur;
            cur = cur->next;
        }
        res[i] = cur->val;
        pre->next = cur->next;
        cur->next = NULL;
        cur = pre->next;
    }
    res[n] = '\0';
    return res;
}

int main186_13()
```

```
//int main()
{
    int key = 2;
    char source[N+1] = "0123456789";
    char *cipher = generateCipher(source, key, N);
    printf(" input:%s [key=%d]\n", source, key);
    printf("output:%s", cipher);
}
```



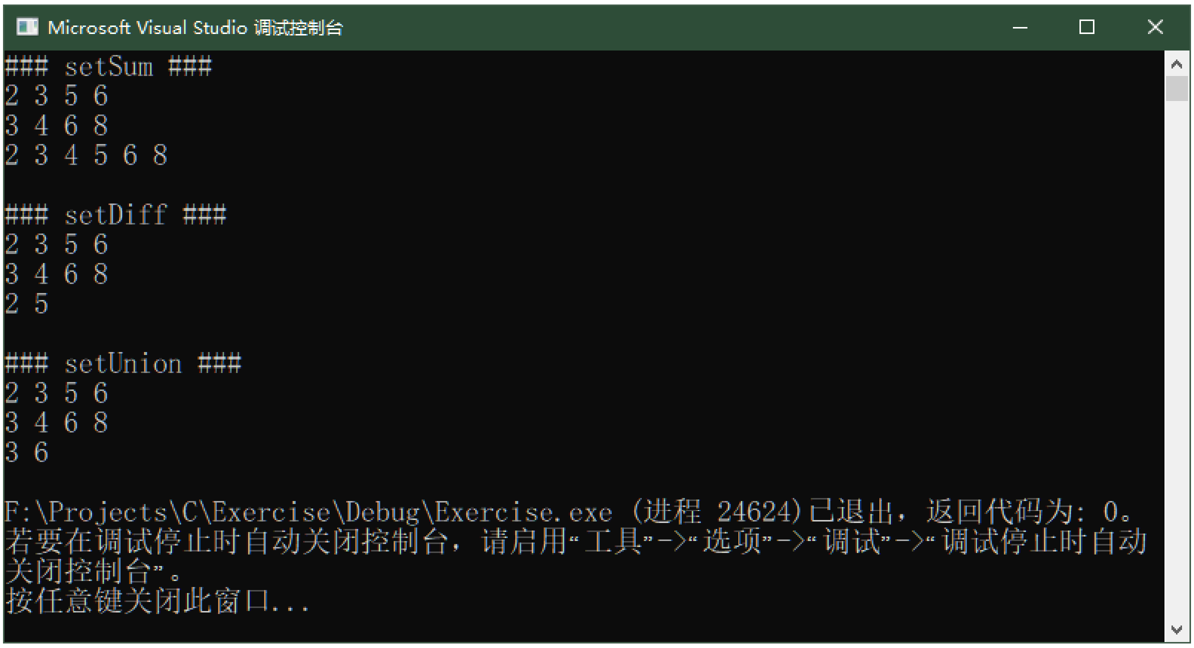
```
 input:0123456789 [key=2]
output:1357926084
F:\Projects\C\Exercise\Debug\Exercise.exe (进程 22176)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口. . .
```

186-14

```
#include<stdio.h>
#include<malloc.h>

typedef struct Node_186_14
{
    int val;
    struct Node_186_14 *next;
}Node;

static Node* createLList(int *a, int n)
{
    int i;
    Node* head = (Node*)malloc(sizeof(Node)), *pre = head, *node;
    head->next = NULL;
    for (i = 0; i < n; i++)
    {
        node = (Node*)malloc(sizeof(Node));
        node->val = a[i];
        node -> next = NULL;
        pre->next = node;
        pre = pre->next;
    }
    return head;
}
static void outputLList(Node* set)
{
    Node* cur = set;
```

```c
    while (cur->next)
    {
        printf("%d ", cur->next->val);
        cur = cur->next;
    }
    printf("\n");
}
static void sumSet(Node *set1, Node* set2)
{
    Node *cur1, *cur2 = set2, *tmp;
    int value;
    int flag;
    while (cur2->next)
    {
        value = cur2->next->val;
        cur1 = set1;
        flag = 0;
        while (cur1->next)
        {
            if (cur1->next->val == value)
            {
                tmp = cur2->next;
                cur2->next = tmp->next;
                tmp->next = NULL;
                flag = 1;
                break;
            }
            else if (cur1->next->val > value)
            {
                tmp = cur2->next->next;
                cur2->next->next = cur1->next;
                cur1->next = cur2->next;
                cur2->next = tmp;
                flag = 1;
                break;
            }
            cur1 = cur1->next;
        }
        //尾结点
        if (!flag)
        {
            tmp = cur2->next->next;
            cur2->next->next = cur1->next;
            cur1->next = cur2->next;
            cur2->next = tmp;
            flag = 1;
        }
        //cur2 = cur2->next;
    }
}
static void minusSet(Node *set1, Node* set2)
{
    Node *cur1 = set1, *cur2 = set2, *tmp;
    int value;
    while (cur2->next)
    {
        value = cur2->next->val;
        cur1 = set1;
        while (cur1->next)
        {
            if (cur1->next->val == value)
            {
                tmp = cur1->next;
                cur1->next = tmp->next;
                tmp->next = NULL;
            }
            else if (cur1->next->val > value)
            {
                break;
            }
            if (!cur1->next)
            {
                break;
            }
            cur1 = cur1->next;
        }
        cur2 = cur2->next;
    }
}
static void unionSet(Node *set1, Node* set2)
{
    Node *cur1 = set1, *cur2 = set2, *tmp;
    int flag;
    int value;
    while (cur1->next)
    {
        cur2 = set2;
        value = cur1->next->val;
        flag = 0;
        while (cur2->next)
        {
            if (value == cur2->next->val)
            {
                flag = 1;
                break;
            }
            cur2 = cur2->next;
        }
        if (!flag)
        {
            tmp = cur1->next;
            cur1->next = tmp->next;
            tmp->next = NULL;
        }
        else
        {
            cur1 = cur1->next;
        }

    }
}

//int main_186_14()
int main()
{
    int arr1[] = { 2,3,5,6 }, arr2[] = { 3,4,6,8 };
```

```c
    Node *set1, *set2;

    printf("### setSum ###\n");
    set1 = createLList(arr1, sizeof(arr1) / sizeof(int));
    set2 = createLList(arr2, sizeof(arr2) / sizeof(int));
    outputLList(set1);
    outputLList(set2);
    sumSet(set1, set2);
    outputLList(set1);

    printf("\n### setDiff ###\n");
    set1 = createLList(arr1, sizeof(arr1) / sizeof(int));
    set2 = createLList(arr2, sizeof(arr2) / sizeof(int));
    outputLList(set1);
    outputLList(set2);
    minusSet(set1, set2);
    outputLList(set1);

    printf("\n### setUnion ###\n");
    set1 = createLList(arr1, sizeof(arr1) / sizeof(int));
    set2 = createLList(arr2, sizeof(arr2) / sizeof(int));
    outputLList(set1);
    outputLList(set2);
    unionSet(set1, set2);
    outputLList(set1);
}
```



```
### setSum ###
2 3 5 6
3 4 6 8
2 3 4 5 6 8

### setDiff ###
2 3 5 6
3 4 6 8
2 5

### setUnion ###
2 3 5 6
3 4 6 8
3 6

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 24624)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口...
```