

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级：计算机类（二）2007班**

**学 号：U202017180**

**姓 名：吴彬杰**

**指导教师：万琳**

**报告日期：2020年10月28日**

**软件学院**

**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)

实验5 数组实验

5.1 实验目的

（1）掌握数组的说明、初始化和使用。

（2）掌握一维数组作为函数参数时实参和形参的用法。

（3）掌握字符串处理函数的设计，包括串操作函数及数字串与数之间转换函数实现算法。

（4）掌握基于分治策略的二分查找算法和选择法排序算法的思想，以及相关算法的实现。

5.2 实验内容及要求

1、 源程序改错与跟踪调试

在下面所给的源程序中，函数strcate(t,s)的功能是将字符串s连接到字符串t的尾部；函数strdelc(s,c)的功能是从字符串s中删除所有与给定字符c相同的字符，程序应该能够输出如下结果：

Programming Language

ProgrammingLanguage Language

ProgramingLnguage

跟踪和分析源程序中存在的问题，排除程序中的各种逻辑错误，使之能够输出正确的结果。

1. 单步执行源程序。进跟踪进入strcate时，观察字符数组t和s中的内容，分析结果是否正确。当单步执行光条刚落在第二个while语句所在行时，i为何值？t[i]为何值？分析该结果是否存在问题。当单步执行光条落在strcate函数块结束标记即右花括号“}”所在行时，字符数组t和s分别为何值？分析是否实现了字符串连接。

（2）跟踪进入函数strdelc时，观察字符数组s中的内容和字符c的值，分析结果是否正确。单步执行for语句过程中，观察字符数组s, j和k值的变化，分析该结果是否存在问题。当单步执行光条落在strdelc函数块结束标记“}”所在行时，字符串s为何值？分析是否实现了所要求的删除操作。

/\*实验5-1程序改错与跟踪调试题程序\*/

#include<stdio.h>

void strcate(char [],char []);

void strdelc(char [],char );

int main(void)

{

char a[]="Language", b[]="Programming";

printf("%s %s\n", b,a);

strcate(b,a); printf("%s %s\n",b,a);

strdelc(b, 'a'); printf("%s\n",b);

return 0;

}

void strcate(char t[],char s[])

{

int i = 0, j = 0;

while(t[i++]) ;

while((t[i++] = s[j++] )!= '\0');

}

void strdelc(char s[], char c)

{

int j,k;

for(j=k=0; s[j] != '\0'; j++)

if(s[j] != c) s[k++] = s[j];

}

解答:

/\* 实验5-1程序改错与跟踪调试题程序 \*/

#include <stdio.h>

void strcate(char[], char[]);

void strdelc(char[], char);

int main(void)

{

    char a[] = "Language", b[100] = "Programming"; //initialize the two arrays

    printf("%s %s\n", b, a);

    strcate(b, a); //execute the function strcat

    printf("%s %s\n", b, a);

    strdelc(b, 'a'); //execute the function strdelc

    printf("%s\n", b);

    return 0;

}

void strcate(char t[], char s[])

{

    int i = 0, j = 0;

    while (t[i++])

        ;

    i--;

    while ((t[i++] = s[j++]) != '\0')

        ; //merge the strings in a sequence

}

void strdelc(char s[], char c)

{

    int j, k;

    for (j = k = 0; s[j] != '\0'; j++)

    {

        if (s[j] != c)

            s[k++] = s[j]; //move the back of the array

    }

    s[k] = '\0'; //mark the end

}

1. 不正确。不正确。

12 L 存在

Programming Language

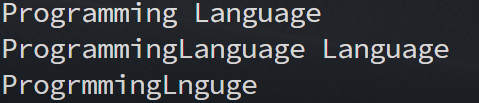
否。

1. 不正确。

存在。

Programming

否。



2、 源程序完善和修改替换

(1) 下面的源程序用于求解瑟夫问题：M个人围成一圈，从第一个人开始依次从1至N循环报数，每当报数为N时报数人出圈，直到圈中只剩下一个人为止。①请在源程序中的下划线处填写合适的代码来完善该程序。

#include<stdio.h>

#define M 10

#define N 3

int main(void)

{

int a[M], b[M]; /\* 数组a存放圈中人的编号，数组b存放出圈人的编号 \*/

int i, j, k;

for(i = 0; i < M; i++) /\* 对圈中人按顺序编号1—M \*/

a[i] = i + 1;

for(i = M, j = 0; i > 1; i--){

/\* i表示圈中人个数，初始为M个，剩1个人时结束循环；j表示当前报数人的位置 \*/

for(k = 1; k <= N; k++) /\* 1至N报数 \*/

if(++j > i - 1) j = 0;/\* 最后一个人报数后第一个人接着报，形成一个圈 \*/

b[M-i] = j ? \_\_\_\_\_\_\_:\_\_\_\_\_\_; /\* 将报数为N的人的编号存入数组b \*/

if(j)

for(k = --j; k < i; k++) /\* 压缩数组a，使报数为N的人出圈 \*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

}

for(i = 0;i < M-1; i++) /\* 按次序输出出圈人的编号 \*/

printf(“%6d”, b[i]);

printf(“%6d\n”, a[0]); /\* 输出圈中最后一个人的编号 \*/

return 0;

}

解答：

#include <stdio.h>

#define M 10

#define N 3

int main(void)

{

    int a[M], b[M]; /\* 数组a存放圈中人的编号，数组b存放出圈人的编号

                                 \*/

    int i, j, k;

    for (i = 0; i < M; i++) /\* 对圈中人按顺序编号1—M \*/

        a[i] = i + 1;

    for (i = M, j = 0; i > 1; i--)

    {

        /\* i表示圈中人个数，初始为M个，剩1个人时结束循环；j表示当前报数人的位置

         \*/

        for (k = 1; k <= N; k++) /\* 1至N报数 \*/

            if (++j > i - 1)

                j = 0; /\* 最后一个人报数后第一个人接着报，形成一个圈

                                 \*/

        //b[M - i] = j ? \_\_\_\_\_\_\_ : \_\_\_\_\_\_;  /\* 将报数为N的人的编号存入数组b \*/

        b[M - i] = j ? a[j - 1] : a[i - 1];

        if (j)

            for (k = --j; k < i; k++) /\* 压缩数组a，使报数为N的人出圈

                                         \*/

                //\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

                a[k] = a[k + 1];

    }

    for (i = 0; i < M - 1; i++) /\* 按次序输出出圈人的编号 \*/

        printf("%6d", b[i]);

    printf("%6d\n", a[0]); /\* 输出圈中最后一个人的编号 \*/

    return 0;

}

②上面的程序中使用数组元素的值表示圈中人的编号，故每当有人出圈时都要压缩数组，这种算法不够精炼。如果采用做标记的办法，即每当有人出圈时对相应数组元素做标记，从而可省掉压缩数组的时间，这样处理效率会更高一些。请采用做标记的办法修改程序，并使修改后的程序与原程序具有相同的功能。

#include<stdio.h>

#define M 10

#define N 3

int main(void)

{

    int a[M], b[M];             /\* 数组a存放圈中人的编号，数组b存放出圈人的编号

                                 \*/

    int i, j, k;

    for (i = 0; i < M; i++)     /\* 对圈中人按顺序编号1—M \*/

        a[i] = i + 1;

    for (i = M, j = 0; i > 0; i--)

    {

        /\* i表示圈中人个数，初始为M个，剩1个人时结束循环；j表示当前报数人的位置 \*/

        for (k = 1; k <= N; k++)    /\* 1至N报数 \*/

        {

            while(!a[j])

                if (++j > M - 1)

                    j = 0;  /\* 最后一个人报数后第一个人接着报，形成一个圈 \*/

            if (++j > M - 1)

                j = 0;

        }

        // b[M - i] =j ? \_\_\_\_\_\_\_ : \_\_\_\_\_\_;  /\* 将报数为N的人的编号存入数组b \*/

        b[M - i] = j ? a[j - 1] : a[M - 1];

        if(--j == -1)

            j = M - 1;

        a[j] = 0;

        #if 0

        if (j)

            for (k = --j; k < i; k++)   /\* 压缩数组a，使报数为N的人出圈

                                         \*/

                \_\_\_\_\_\_\_\_\_\_\_\_\_\_;

        #endif

    }

    for (i = 0; i < M; i++) /\* 按次序输出出圈人的编号 \*/

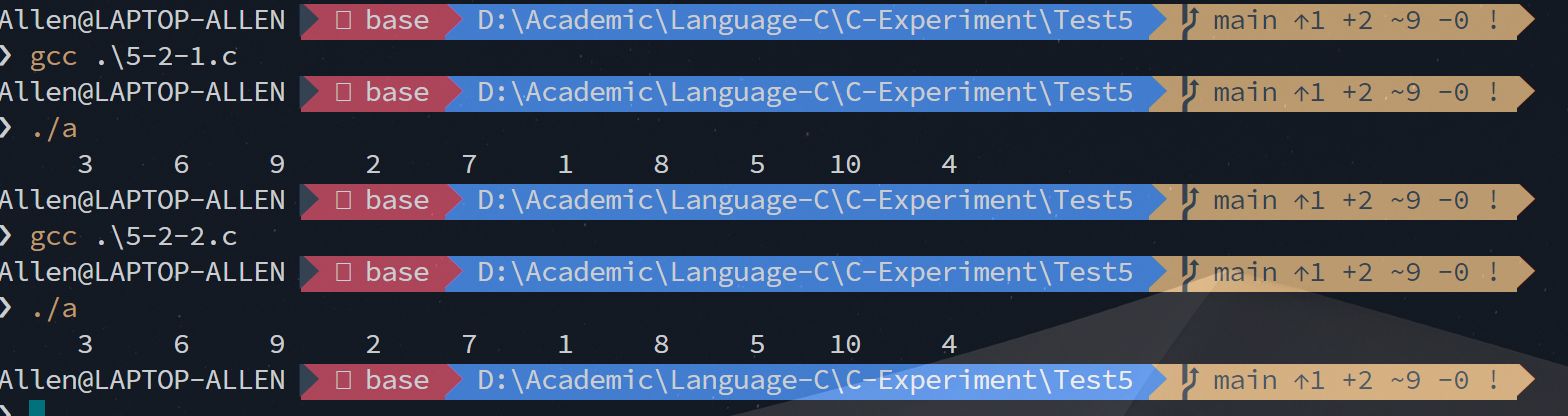
        printf("%6d", b[i]);

    // printf("%6d\n", a[0]);   /\* 输出圈中最后一个人的编号 \*/

    return 0;

}

测试数据：



3、 程序设计

（1）输入一个整数，将它在内存中二进制表示的每一位转化成对应的数字字符并且存放到一个字符数组中，然后输出该整数的二进制表示。

解答：

程序清单：

#include <stdio.h>

#include <string.h>

int main(void)

{

    int n, i = 0, a[32];

    memset(a, 0, sizeof(a));//initialize the array

    scanf("%d", &n);

    if (n < 0)

    {

        n = -n;

        printf("-");

    } //jugde whether it is below zero

    if (n != 0)

    {

        while (n > 0)

        {

            a[i] = n % 2;

            n /= 2;

            i++;

        } //break the number into binary form

        for (int j = i - 1; j >= 0; j--)

        {

            printf("%d", a[j]);

        }

    }

    else

    {

        printf("0");

    }

}

测试数据：



（2）编写一个C程序，要求采用模块化程序设计思想，将相关功能用函数实现，并提供菜单选项。该程序具有以下功能：

①输入n个学生的姓名和C语言课程的成绩。

②将成绩按从高到低的次序排序，姓名同时进行相应调整。

③输出所有学生的姓名和C语言课程的成绩。

（3）对程序设计第（2）题的程序增加查找功能：输入一个C语言课程成绩值，用二分查找进行搜索。如果查找到有该成绩，则输出该成绩学生的姓名和C语言课程的成绩；否则，输出提示“not found!”。

解答：

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

struct student

{

    char name[255];

    int score;

} a[255];

int sc[255];

int n;

void adjust\_heap(int \*array, int i, int lef)

{

    int temp = array[i];

    for (int k = i \* 2 + 1; k < lef; k = k \* 2 + 1)

    {

        if (k + 1 < lef && array[k] < array[k + 1])

        {

            k++;

        }

        if (array[k] > temp)

        {

            array[i] = array[k];

            i = k;

        }

        else

        {

            break;

        }

    }

    array[i] = temp;

}

void swap(int \*array, int a, int b) //the swap function

{

    int temp = array[a];

    array[a] = array[b];

    array[b] = temp;

}

void heap\_sort(int \*array, int n) //heap sort

{

    for (int i = n / 2 - 1; i >= 0; i--)

    {

        adjust\_heap(array, i, n);

    }

    for (int j = n - 1; j > 0; j--)

    {

        swap(array, 0, j);

        adjust\_heap(array, 0, j);

    }

}

int binary\_search(int array[], int low, int high, int target) //bin search

{

    while (low <= high)

    {

        int mid = low + (high - low) / 2;

        if (array[mid] > target)

            high = mid - 1;

        else if (array[mid] < target)

            low = mid + 1;

        else

            return mid;

    }

    return -1;

}

void set\_student(void)

{

    scanf("%d", &n);

    getchar();

    for (int i = 0; i < n; i++)

    {

        scanf("%s %d", a[i].name, &a[i].score);

        getchar();

    }

}

void sort\_student(void)

{

    for (int i = 0; i < 255; i++)

        sc[i] = a[i].score;

    heap\_sort(sc, n);

}

void get\_student()

{

    for (int i = n - 1; i >= 0; i--)

    {

        for (int j = 0; j < n; j++)

            if (sc[i] == a[j].score)

            /\*

            \* I should have used the binary sort function,

            \* but in case the two students get the same score,

            \* I kept a traditional approach.

            \*/

                printf("%s %d\n", a[j].name, sc[i]);

    }

}

void search\_student(int target)

{

    int c = 0;

    for (int i = 1; i < n; i++)

    {

        if (a[i].score == target)

        {

            printf("%s %d\n", a[i].name, a[i].score);

            c++;

        }

    }

    if (c == 0)

        printf("not found");

}

int main(void)

{

    int sel;

    while (1)

    {

        printf("1.Input\n2.Sort\n3.Output\n");

        scanf("%d", &sel);

        if (sel == 1)

            set\_student();

        if (sel == 2)

            sort\_student();

        if (sel == 3)

            get\_student();

        if (sel == 4)

        {

            int sea;

            scanf("%d", &sea);

            search\_student(sea);

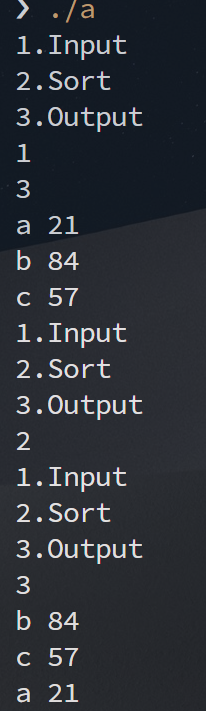
        }

    }

    return 0;

}

测试数据：



4、程序设计选做题

编写并上机调试运行能实现以下功能的函数和程序。

编写函数strnins(s,t,n),其功能是：可将字符数组 t中的字符串插入到字符数组 s中字符串的第n个字符的后面。

#include <stdio.h>

#include <string.h>

void strnins(char \*, char \*, int);

int main(void)

{

    char s[255], t[255];

    int n;

    scanf("%s", s);

    scanf("%s", t);

    scanf("%d", &n);

    strnins(s, t, n);

    for (int i = 0; i <= strlen(s) + strlen(t); i++)

        printf("%c", s[i]);

    return 0;

}

void strnins(char \*s, char \*t, int n)

{

    n--;

    for (int i = strlen(s) + strlen(t); i >= strlen(t) + n; i--)

    {

        s[i] = s[i - (strlen(t))];

    } //move the array

    for (int i = n + 1; i <= strlen(t) + n; i++)

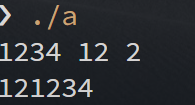
    {

        s[i] = t[i - n - 1];

    } //insert the new array into the target array

}

# 测试数据



# 实验总结

C语言的数组实现本质是连续内存，编写时需要尤其注意内存安全性。