

## 《计算机程序设计》实验报告

姓名： 宋亦寒 学号： PB25000200 实验日期： 2025 年 10 月 27 日

实验名称	C 程序设计入门
------	----------

### 一、 实验目的要求

1. 熟练运用 C 语言的二维数组和字符数组；
2. 掌握二维数组的转置与排序等；
3. 尝试实现较为复杂的算法流程。

### 二、 实验内容

1. 实验指导书 2.4.3 自主编程练习 9、10

补充题：

2. 将一个 3\*3 的二维整型数组转置，即行列互换，并输出结果
3. 仿照课本数值数组的插入排序算法，对一维字符型数组进行插入排序，将排序结果写入文件 (\*.txt) 中；
4. 画出实验指导书 2.4.3 自主编程练习 9 的流程图。

选做题：

5. 概率的计算-百囚徒挑战问题描述：理想国有 100 名囚徒，国王决定给他们一次特赦的机会，但条件是必须通过一项挑战，挑战的规则如下：所有囚徒从 1-100 进行编号。将编号 1-100 的 100 个号码牌随机放在 100 个盒子中。每名囚徒可以打开最多 50 个盒子，如果找到对应自己编号的号码牌，则该囚徒挑战成功，否则挑战失败。所有囚徒全部挑战成功，整个挑战才算成功；任意一名囚徒挑战失败，则该项挑战失败。囚徒们在挑战前可以商定策略，挑战开始后则不再允许交流。请帮助囚徒们设计一个成功概率较高的挑战策略。实训要求：在了解其中的数学原理以及计算复杂度的基础上，编程验证你所提出的百囚徒挑战策略的成功概率。为便于思考与论证，可以先将其简化成 10 囚徒挑战问题。

撰写实验报告，格式为 学号-姓名-实验 3. pdf，在上机平台提交

### 三、 调试中的问题及解决方法（字数不限）

1. 不会写如果开头为 000000 则结束程序：查找资料，得知可以使用 `strncmp` 进行比较；
2. 忘记在变量名称中出现下划线也是合法标识符：对照期望输出与实际输出发现；
3. 忘记写入文件的方法、不会使用 `fputs`，经过查找资料，学会使用 `fprintf(fp, "%d ", a[i])` 向文件中写入数组；
4. 冒泡排序显示“运行时间过长”：询问老师，得知可能是数组越界，经过检查后修正。

## 四、 上机程序

### 1. 实验指导书 2.4.3 第 9 题

```
#include <stdio.h>

int main() {
    int k = 0;           // 记录已输入的字符串次数
    char a[1000];        // 存储输入的字符串（替换数字后的结果）
    char ch;             // 临时读取字符
    int num;             // 当前字符串的长度

    while (k < 3) {      // 共循环输入 3 次字符串
        num = 0;
        printf("输入一个字符串:");

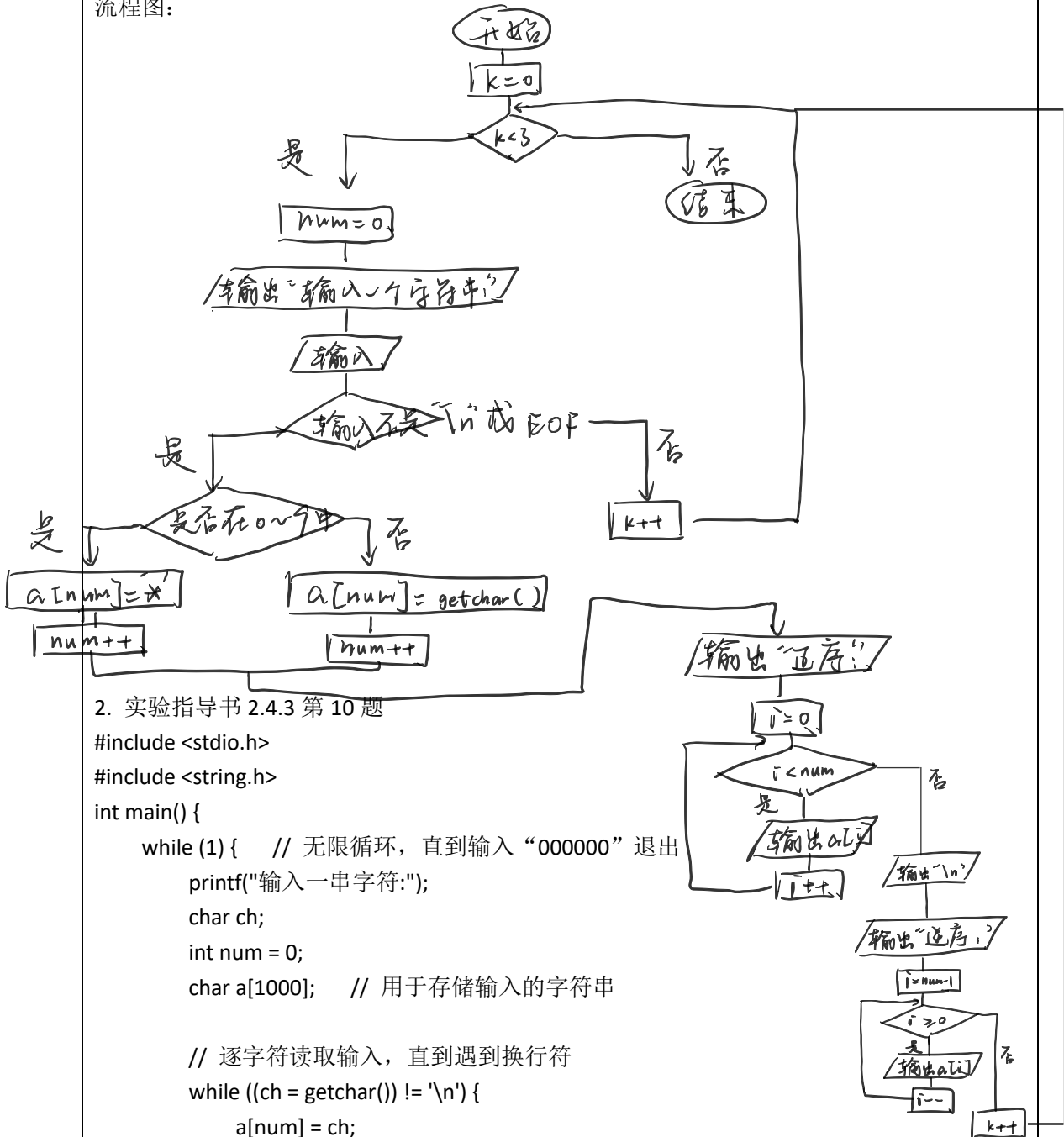
        // 逐字符读取一行输入
        while ((ch = getchar()) != '\n' && ch != EOF) {
            if (ch >= '0' && ch <= '9') {
                a[num++] = '*'; // 若为数字则替换为 '*'
            } else {
                a[num++] = ch;  // 否则原样保存
            }
        }

        printf("正序:");
        for (int i = 0; i < num; i++) {
            putchar(a[i]);     // 正序输出
        }
        printf("\n");

        printf("逆序:");
        for (int i = num - 1; i >= 0; i--) {
            putchar(a[i]);     // 逆序输出
        }
        printf("\n");

        k++; // 计数器加 1，准备下一轮输入
    }
}
```

流程图:



## 2. 实验指导书 2.4.3 第 10 题

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

```
#include <string.h>
```

```
int main() {
```

```
    while (1) { // 无限循环，直到输入“000000”退出
```

```
        printf("输入一串字符:");
```

```
        char ch;
```

```
        int num = 0;
```

```
        char a[1000]; // 用于存储输入的字符串
```

```
        // 逐字符读取输入，直到遇到换行符
```

```
        while ((ch = getchar()) != '\n') {
```

```
            a[num] = ch;
```

```
            num++;
```

```
        }
```

```
        a[num] = '\0'; // 字符串末尾加结束符
```

```
        int flag = 1; // 标志变量：1 表示合法，0 表示不合法
```

```
        // 如果输入“000000”，程序结束
```

```
        if (strncmp(a, "000000", 6) == 0) return 0;
```

```
        // 若首字符是数字，则不合法
```

```
        else if (a[0] >= '0' && a[0] <= '9') flag = 0;
```

```
// 检查每个字符是否合法
else {
    for (int i = 0; a[i] != '\0'; i++) {
        if (a[i] >= 'a' && a[i] <= 'z') flag = 1;           // 小写字母合法
        else if (a[i] >= 'A' && a[i] <= 'Z') flag = 1;       // 大写字母合法
        else if (i != 0 && a[i] >= '0' && a[i] <= '9') flag = 1; // 数字仅限非首位
        else if (a[i] == '_') flag = 1;                     // 下划线合法
        else {                                              // 其他字符
            flag = 0;
            break;
        }
    }
}

// 输出判断结果
if (flag == 1)
    printf("\n%s\n是合法的 C 标识符\n", a);
else
    printf("\n%s\n不是合法的 C 标识符\n", a);
}
```

3. 将一个 3\*3 的二维整型数组转置，即行列互换，并输出结果

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

```
int main() {
    int a[3][3];    // 定义 3×3 整型数组
    int i, j;

    // 输入 9 个整数，按行存入数组
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++) {
            scanf("%d", &a[i][j]);
        }
    }

    // 输出转置矩阵：行列互换
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++) {
            printf("%d ", a[j][i]);    // 输出时交换行列索引
        }
        printf("\n");    // 每行输出结束后换行
    }
}
```

```
}

4. 仿照课本数值数组的插入排序算法，对一维字符型数组进行插入排序，将排序结果写入
文件 (*.txt) 中
#include <stdio.h>
int main()
{
    int n;
    scanf("%d", &n);          // 读入数组长度
    int a[n];                  // 定义可变长度数组

    // 输入数组元素
    for (int i = 0; i < n; i++) {
        scanf("%d", &a[i]);
    }

    int key;

    // 插入排序（从第二个元素开始）
    for (int i = 1; i < n; i++) {
        key = a[i];            // 当前待插入的元素
        int j = i - 1;
        // 将比 key 大的元素向后移动
        while (j >= 0 && a[j] > key) {
            a[j + 1] = a[j];
            j--;
        }
        a[j + 1] = key;        // 插入到正确位置
    }

    // 打开文件 queue.txt（写模式）
    FILE *fp;
    fp = fopen("queue.txt", "w");

    // 将排序结果写入文件
    for (int i = 0; i < n; i++) {
        fprintf(fp, "%d ", a[i]);
    }
    fclose(fp);                // 关闭文件
}
```