《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 12 月 22 日

实验名称	C 程序设计入门

一、 实验目的要求

理解指针在结构体中的应用，复习

二、 实验内容

1.课本 P234：15
2.实验指导书 2.6 综合练习：4 动态规划算法
3.补充题 1：改写课本 5.3-1，对于 stu1 的赋值采用指针和 scanf 函数实现，并对代码添加注释（粘贴代码）
4.补充题 2：综合测试（2）的四：画流程图题 使用结构体数组 st[N]存储学生的学号、姓名和成绩，编写“成绩录入程序”

撰写实验报告，格式为 学号-姓名-实验 10.pdf，在上机平台提交

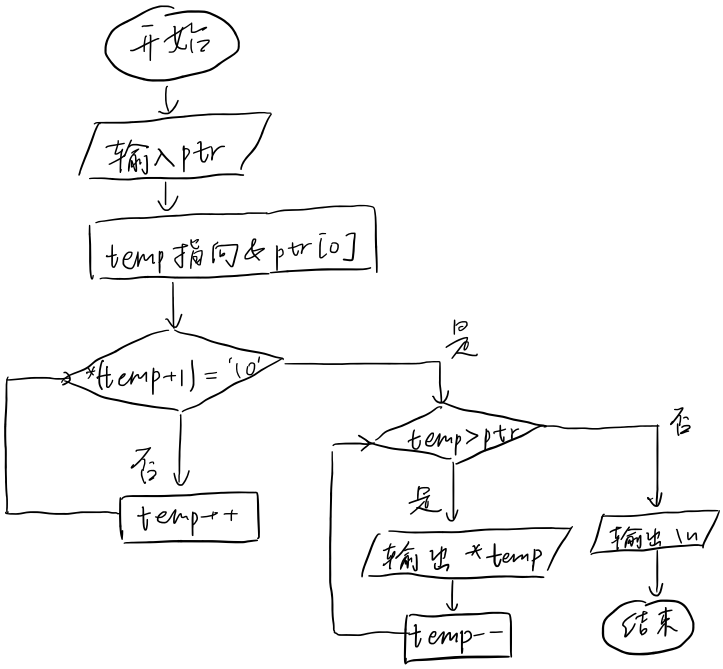
三、 上机程序（有注释）

1. 课本 P234：15

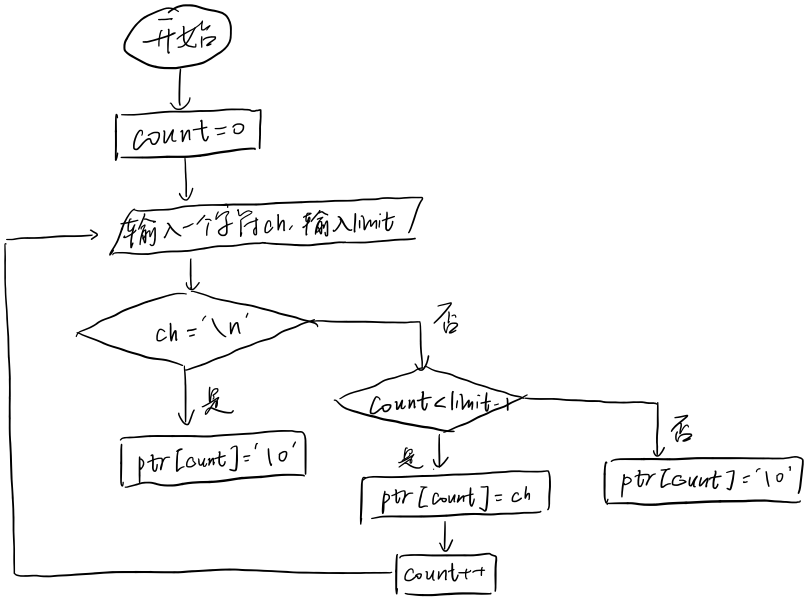
```
#include <stdio.h>

#define MAX_SIZE 100

// 函数 1：输入字符串
void inputString(char *ptr, int limit) {
    int count = 0;
    char ch;
    printf("请输入字符串（按回车结束）: ");

    // 使用 getchar() 逐个读取，直到换行或达到上限
    // 注意：limit - 1 是为了给 '\0' 留出空间
    while (count < limit - 1 && (ch = getchar()) != '\n') {
        *(ptr + count) = ch;
        count++;
    }
    // 在末尾补上字符串结束标志
    *(ptr + count) = '\0';
}

// 函数 2：逆序输出字符串
```

```
void reverseOutput(char *ptr) {
    char *temp = ptr;

    // 首先移动指针到字符串的末尾（'\0'的前一个位置）
    if (*temp == '\0') return; // 如果是空字符串则直接返回

    while (*(temp + 1) != '\0') {
        temp++;
    }

    printf("逆序输出结果为: ");
    // 从末尾指针开始往前遍历，直到回到首地址 ptr
    while (temp >= ptr) {
        putchar(*temp);
        temp--;
    }
    printf("\n");
}

int main() {
    char str[MAX_SIZE];

    // 调用输入函数
    inputString(str, MAX_SIZE);

    // 调用逆序输出函数
    reverseOutput(str);

    return 0;
}
```

逆序输出



输入：



2. 实验指导书 2.6 综合练习：4 动态规划算法

```
#include <stdio.h>

int dp[101][101];      // dp[i][j] 表示从第 i 个物品开始，在容量为 j 时的最大价值
int w[101];            // w[i] 表示第 i 个物品的重量
int v[101];            // v[i] 表示第 i 个物品的价值

// 返回两个整数中的较大值
int max(int a, int b)
{
    return (a>b)?a:b;
}

int main()
{
    int n, c;          // n 为物品数量，c 为背包容量
    //printf("输入物品的数量:");
    scanf("%d", &n);

    //printf("输入%d 个物品的重量:", n);
    for (int i=1; i<=n; i++){
        scanf("%d", &w[i]);
    }

    //printf("输入%d 个物品的价值:", n);
```

```
for (int i=1; i<=n; i++){
    scanf("%d", &v[i]);
}

//printf("输入背包可以容纳的最大重量:");
scanf("%d", &c);
printf("\n");

// 初始化边界条件：没有物品可选时，价值为 0
for (int j=0; j<=c; j++){
    dp[n+1][j]=0;
}

// 自底向上进行动态规划计算
for (int i=n; i>=1; i--){
    for (int j=0; j<=c; j++) {
        if (j<w[i]){
            // 当前容量不足，不能选第 i 个物品
            dp[i][j]=dp[i+1][j];
        } else {
            // 在选与不选第 i 个物品中取最大价值
            dp[i][j]=max(dp[i+1][j], v[i]+dp[i+1][j-w[i]]);
        }
    }
}

// 输出 dp 表
for (int i=n; i>=1; i--){
    printf(" %d :", i);
    for (int j=1; j<=c; j++){
        printf(" %d", dp[i][j]);
    }
    printf("\n");
}

int current_c=c;   // 当前剩余容量
int total_w=0;     // 背包中物品的总重量
int total_v=0;     // 背包中物品的总价值

// 根据 dp 表回溯选择的物品
for (int i=1; i<=n; i++){
    if (dp[i][current_c]>dp[i+1][current_c]){
        // 若选当前物品能使价值增加，则选入背包
        printf("%d 号物品(重量:%d,价值:%d)装入了背包\n", i, w[i], v[i]);
```

```
                total_w+=w[i];
                total_v+=v[i];
                current_c-=w[i];
            }
        }

        // 输出最终结果
        printf("背包里所装入物品的总重量为:%d,总价值为:%d\n", total_w, total_v);
        return 0;
}
```

3. 补充题 1：改写课本 5.3-1，对于 stu1 的赋值采用指针和 scanf 函数实现，并对代码添加注释

```
    // 例 5.3-1
    #include <stdio.h>
    #include <stdlib.h>
    #include <string.h>

    struct strStudent {
        long int lNum;
        char stName[20];
        char cGender;
        float fScore;
    };

    int main(void) {
        struct strStudent stu1;
        struct strStudent *ps;

        ps = &stu1; // 使指针 ps 指向结构体变量 stu1

        // 使用指针 ps 和 scanf 函数为结构体成员输入数据
        printf("请输入学号(long): ");
        scanf("%ld", &ps->lNum);          // 通过指针获取学号

        printf("请输入姓名(string): ");
        scanf("%s", ps->stName);

        printf("请输入性别(char M/F): ");
        scanf(" %c", &ps->cGender);

        printf("请输入分数(float): ");
        scanf("%f", &ps->fScore);         // 通过指针获取分数
```

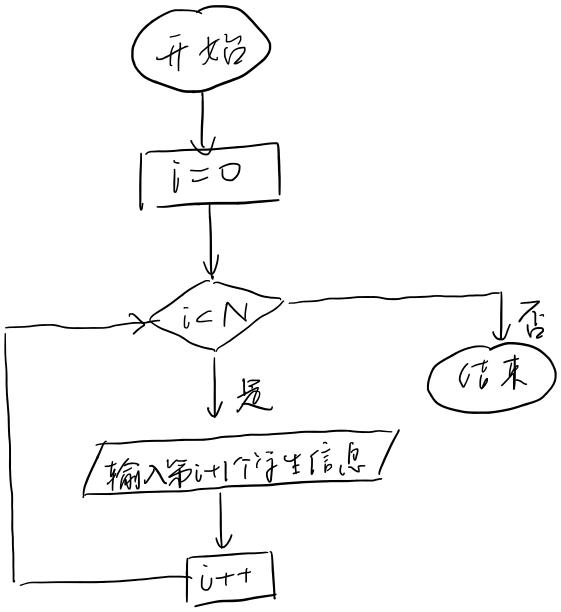
```
        printf("----------学生信息----------\n");

        // 方式 1：通过结构体变量名直接访问
        printf("No.:%ld, Name:%s, Gender:%c, Score:%5.1f\n", \
               stu1.lNum, stu1.stName, stu1.cGender, stu1.fScore);

        // 方式 2：通过指针解引用 (*ps) 访问
        printf("No.:%ld, Name:%s, Gender:%c, Score:%5.1f\n", \
               (*ps).lNum, (*ps).stName, (*ps).cGender, (*ps).fScore);

        // 方式 3：通过指针箭头运算符 -> 访问（最常用）
        printf("No.:%ld, Name:%s, Gender:%c, Score:%5.1f\n", \
               ps->lNum, ps->stName, ps->cGender, ps->fScore);

        return 0;
}
```

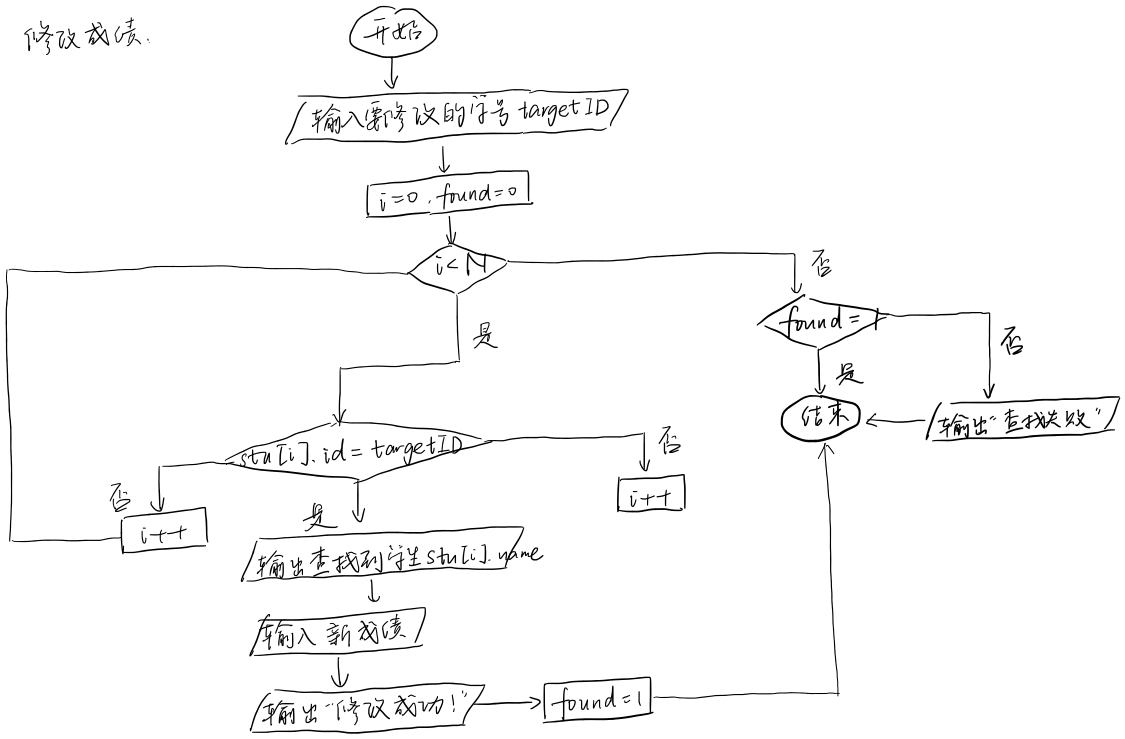
4. 补充题 2：综合测试（2）的四：画流程图题 使用结构体数组 st[N]存储学生的学号、姓名和成绩，编写“成绩录入程序”

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

    #define N 5

    struct Student {
        char id[20];
        char name[50];
        float score;
    } st[N];

    // 1. 输入成绩模块
    void inputScores() {
        for (int i=0; i<N; i++) {
            printf("请输入第 % d 个学生的学号、姓名、成绩: ", i+1);
            scanf("%s %s %f", st[i].id, st[i].name, &st[i].score);
        }
    }

    // 2. 修改成绩模块
    void modifyScore() {
        char targetID[20];
        int found=0;
        printf("请输入要修改的学号: ");
```

```
        scanf("%s", targetID);

        for (int i=0; i<N; i++){
            if (strcmp(st[i].id, targetID)==0){
                printf("查找到学生 %s，请输入新成绩: ", st[i].name);
                scanf("%f", &st[i].score);
                printf("修改成功! \n");
                found=1;
                break; // 找到后立即结束模块
            }
        }

        if (!found){
            printf("查询失败，未找到该学号。\n");
        }
}
```

输入成绩：





四、 调试中的问题及解决方法（字数不限）

1. 字符串逆序输出中的指针边界问题

- **问题：** 在 reverseOutput 函数中，初始尝试移动指针时，发生越界。
- **解决方法：** * 通过 while (*(temp+1) != '\0') 来定位到最后一个有效字符，而不是定位到 \0。

2. 结构体输入中的缓冲区残留

- **问题：** 在连续使用 scanf 输入学号、姓名后，输入性别（字符型）时，程序直接跳过了输入步骤。
- **解决方法：**之前的 scanf 在输入缓冲区留下了换行符 \n，%c 会直接读取这个换行符。在 %c 前加一个空格，写成 scanf(" %c", &ps->cGender);。这个空格会告诉编译器跳过所有空白符（空格、换行、制表符），直到读到第一个有效字符。

3. 结构体指针与运算符优先级

- **问题：** 混淆了 *ps.lNum 和 (*ps).lNum，导致编译报错。
- **解决方法：**在 C 语言中，点运算符 . 的优先级高于解引用运算符 *。使用圆括号明确优先级：(*ps).lNum 或箭头运算符 ps->lNum，这在处理结构体指针时更加直观且不易出错。

4. 字符串比较逻辑错误

- **问题：** 在修改成绩模块中，尝试使用 if (st[i].id == targetID) 来判断学号是否相等，结果错误。
- **解决方法：** id 和 targetID 都是字符数组（指针），直接使用 == 比较的是它们的内存地址而非内容。引入 <string.h> 头文件，使用 strcmp(st[i].id, targetID) == 0 来进行内容的逻辑比较。