学习活动 3　制定方案

实现本任务方案

● **实现思路**

通过对本任务的分析及相关知识学习，制定方案如下：

（1）定义程序中使用到的变量（3 个）。

3 个变量分别为保存怪物的生命值、每次攻击值和总攻击次数。

（2）用死循环的方式实现的攻击过程。

输入攻击值，并用怪物生命值减去攻击值。

条件判断怪物生命值是否<=0。

如果不成立，则继续循环。

如果成立，则结束循环，显示“Game Over!” 和总攻击次数。

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 KillEnemy。

（2）在 main()中按实现思路编写程序代码。

学习活动 4　实施实现

任务实现

● **实现代码**

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 KillEnemy。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()中按实现思路实现任务，其代码如下。

```
int main()
{
    printf("***************模拟打怪物小游戏***************\n\n");
    printf("注意怪物来袭，开始攻击！\n\n");
    //（1）定义变量
    int HealthValue=10000;   //假设怪物的生命值
    int Killnum=0;   //记录打击次数
    int Attack=0;   //攻击力
    //（2）用死循环实现攻击过程
    while(1)
    {
        Killnum=Killnum+1;   //攻击次数加 1
        printf("输入你的攻击值（0-1000）：");
        scanf("%d",&Attack);
        //计算怪物生命值
        HealthValue=HealthValue-Attack;   //生命值-攻击值
        //判断怪物生命值是否被打光
```

```
            if(HealthValue<=0)
            {
                printf("Game Over! \n");
                break;   //结束循环
            }
            printf("怪物还活着，继续攻击\n");
        }
        //显示攻击次数
        printf("怪物被灭亡，  你一共攻击了：%d 次.",Killnum);
        return 0;
    }
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

利用 while 循环语句模拟实现游戏中“打怪”的程序，按制定的方案进行任务实现，在正确的情况下，任务实现的效果如图 4.8 所示。

```
**************模拟打怪物小游戏**************

注意怪物来袭，开始攻击！

输入你的攻击值（0-1000）：99
怪物还活着，继续攻击
输入你的攻击值（0-1000）：800
怪物还活着，继续攻击
输入你的攻击值（0-1000）：1000
怪物还活着，继续攻击
输入你的攻击值（0-1000）：980
怪物还活着，继续攻击
输入你的攻击值（0-1000）：888
怪物还活着，继续攻击
输入你的攻击值（0-1000）：900
怪物还活着，继续攻击
输入你的攻击值（0-1000）：350
怪物还活着，继续攻击
输入你的攻击值（0-1000）：600
怪物还活着，继续攻击
输入你的攻击值（0-1000）：800
怪物还活着，继续攻击
输入你的攻击值（0-1000）：900
怪物还活着，继续攻击
输入你的攻击值（0-1000）：999
怪物还活着，继续攻击
输入你的攻击值（0-1000）：1000
怪物还活着，继续攻击
输入你的攻击值（0-1000）：1000
Game Over!
怪物被灭亡，  你一共攻击了：13次.
```

图 4.8　任务运行效果

● **验收结果**

序号	验 收 内 容	实现效果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					

续表

序号	验 收 内 容	实现效果				
		A	B	C	D	E
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

利用 while 循环语句模拟实现了游戏中“打怪”的程序。

● **技术层面**

while 循环语句是实现程序重复执行某一段程序块的方式之一。

1．语法

```
while(条件表达式)
{
    程序块
}
```

说明：

while 是命令动词，后面紧跟一对小括号，括号内写条件表达式；

{}内是循环执行的程序代码。

执行流程如下。

（1）判断条件是否成立。

（2）如果成立则执行循环；如果不成立则结束循环。

（3）当程序执行到 while 循环的“}”时，返回第一步。

以此类推，从而实现循环执行程序的目的。

2．死循环

死循环指进入循环后，永远不结束的循环。真正的死循环是没有任何意义的，所以大家在写循环程序时，一定注意。但是我们也可以利用死循环的特性，在循环中设定结束循环的条件，就可以实现不确定循环次数的程序了。

● **课程思政**

通过本任务的学习，同学们应该明白，再好玩的游戏，也只是一堆“程序代码”而已。

在日常生活中，我们经常听到“游戏人生”这个词，它的核心是“人生”。因此，我们应该拥有一颗积极向上的心，去努力规划，实现美好的人生，而不应该被“游戏”左右。

另外，今天我们学习了死循环的知识，它也是有结束条件的。那么我们做人更应具有自己的底线。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

循环界的“先上车，后买票”

下一回：多久才能存够 100 元

任务 4　多久才能存够 100 元

目标描述

任务描述
● **编写程序实现** 编写 C 语言程序实现：小丽用多少天才能存够 100 元？ 描述： 小丽的妈妈每天给她 2.5 元，她都会存起来。但是，每当存到第 5 天或是 5 的倍数时，她都会花掉 6 元，请问经过多少天，小丽才能存够 100 元？ ● **技术层面** 掌握 do-while 循环语句的相关知识。 应用 do-while 循环语句实现本任务。 ● **课程思政** 家国情怀。 树立正确的金钱观。

学习活动 1　接领任务

领任务单
● 任务确认 编写 C 语言程序，计算存够 100 元所需的天数。 具体要求如下： （1）程序最终能正确展示输出结果； （2）掌握 C 语言代码的使用规范（变量命名及注释说明）； （3）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

本任务通过编写 C 语言程序，计算出小丽多少天能存够 100 元。

（1）目标金额总数为 100 元，初始为 0；

（2）使用循环来实现存钱的过程；

（3）每循环一次；

（4）存钱总数+2.5 元，存钱天数+1；

（5）条件判存储总数>=100；

（6）如果不成立，则继续循环；

（7）如果成立，则结束循环，并显示存钱天数。

知识学习：do-while 循环语句

知识学习

do-while 循环语句是 C 语言循环语句之一，在给定的条件成立时，可反复执行某段程序，直到条件不成立时，跳出循环执行循环后的代码。

1．语法

```
do{
        循环操作语句
} while ( 循环条件 );
...
```

说明：

（1）do-while 语句可以实现循环结构程序，{}内就是循环执行的程序代码；

（2）先执行一遍循环操作，再判断条件。如果符合条件，则循环继续执行，否则循环结束。

学习笔记

2．do-while 循环语句执行流程（见图 4.9）

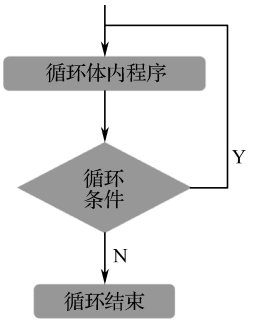


图 4.9　do-while 循环语句执行

3．do-while 循环语句举例

实现 100 以内（含 100）整数之和。

```
int main()
{
    //定义变量
    int i;   //循环变量
    int sum=0;   //保存结果的变量
    i=1;   //这里初值为 1
    do
    {
        sum=sum+i;
        i=i+1;
    }while(i<=100);

    printf("1+2+3+...+100=%d",sum);
    return 0;
}
```

while 实现的例子。

```
int main()
{
    //定义变量
    int i;   //循环变量
    int sum=0;   //保存结果的变量

    i=0;   //这里初值为 0
    while(i<=100)
    {
        sum=sum+i;
        i=i+1;
    }
    printf("1+2+3+...+100=%d",sum);
    return 0;
}
```

因为 do-while 循环是先执行再判断条件，所以会存在多执行一次循环的情况，所以循环变量初值是 1。

这也就是循环界的“先上车，后买票”。

学习活动 3　制定方案

实现本任务方案

● **实现思路**

通过对本任务的分析及相关知识学习，制定方案如下：

（1）定义程序中使用到的变量（2 个）

一个变量用于记录存钱总数；另一个变量用于记录存钱的天数

（2）利用循环实现存钱过程

由于不知道要存多少天，才能存够 100 元，所以使用死循环的方法。

每次循环：

存钱总数 ＋2.5 元；

判断存钱天数是不是 5 的倍数？如果为是，则存钱总数-6 元；

判断存钱总数 >= 100 元；

如果为是，则显示结果，结束循环，程序结束；

如果为否，则存钱天数 ＋1，继续循环。

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 SaveMoney。

（2）在 main()中按实现思路编写程序代码。

学习活动 4　实施实现

任务实现

● **实现代码**

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 SaveMoney。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()中按实现思路完成任务，其代码如下。

```
int main()
{
    //定义变量
    float moneysum=0;   //钱总数
    int   daycount=1;   //存钱天数
    printf("小丽每天存 2.5 元，每 5 天花 6 元，多少天才能存够 100 元？\n");
    do
    {
```

```
            moneysum=moneysum+2.5;   //每天存 2.5 元
            //判断是否是 5 的倍数
            if(daycount % 5 ==0 )
            {
                printf("第%d 天，已经存钱：%.1f 元(",daycount,moneysum);
                moneysum=moneysum-6;   //花掉 6 元
                printf("今天花出 6 元，还剩:%.1f 元)\n",moneysum);
            }
            //判断是否已经存够 100
            if(moneysum>=100)
            {
                printf("小丽存够 100 元，共用了 %d 天",daycount);
                break;   //结束循环
            }
            //天数增加 1
            daycount=daycount+1;

        }while(1);

        return 0;
    }
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

利用 do-while 循环语句实现了小丽存钱的程序，任务实现的效果如图 4.10 所示。

图 4.10　存钱任务的运行效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

利用 do-while 循环语句，实现了小丽存钱的程序。

● 技术层面

do-while 循环语句是实现程序重复执行某一段程序块的方式之一。

语法：

```
do
{
    程序块
}while(条件表达式);
```

说明：

do 和 while 是命令动词，while 后面紧跟一对小括号，括号里写条件表达式，结束时要有“;”；

{}内是循环执行的程序代码。

执行流程：

（1）进入循环体内执行程序；

（2）判断条件，成立执行循环，如果不成立则结束循环；

（3）当程序执行到 while 循环的“}”时，返回（2）。

以此类推，从而实现循环执行程序的目的。

注意：

do-while 循环语句可能会出现已经执行了，但结果不满足条件的情况，所以在使用时一定要注意这个特性。

● **课程思政**

通过本任务的学习，发现小丽用 74 天才能存够 100 元。

赚钱不易，花钱如流水。

在生活中，我们应该学会节约、不攀比，更不能去借贷。

树立正确的金钱观。

同时，我们也应该以生在中国而自豪，因为国家的强大，给了我们一个更安心的学习环境。

感恩父母及亲人，因为有他们做坚强的后盾，才能有这样的幸福生活。

● **教学拓展**

试着使用 do-while 循环语句实现上一次的“打怪”游戏。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

说说这两个兄弟

下一回：一对“孪生兄弟”

（我们已经知道循环中 **break** 语句的功能了，那它的另一个兄弟是谁）

任务 5　一对“孪生兄弟”

目标描述

任务描述
● **目标实现** 优化之前实现的 BMI 程序，让它变得更具有交互性。 ● **技术层面** 掌握循环中的 break 语句、continue 语句的相关知识。 应用 break 语句、continue 语句解决实际问题。 ● **课程思政** 站在用户的角度思考问题。

学习活动 1　接领任务

领任务单
● 任务确认 实现对之前已经完成的 BMI 程序的优化。 具体要求如下： （1）实现可重复计算任意个人的 BMI 值； （2）对 BMI 程序增加询问功能，让其变得更具有交互性； （3）掌握 C 语言代码的使用规范（变量命名及注释说明）； （4）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

实现对之前已经完成 BMI 程序的优化。

（1）根据计算所得的 BMI 值进行区间判断。

① 结论：属于哪种类型。

② 建议：给出你的建议。

但是，仍存在问题。

（2）程序启动一次只能计算一个人的 BMI 值，如果要计算 20 个人的 BMI 值，就要重复启动 20 次程序。

（3）程序的交互性差。

所以要实现对程序的优化。

增加询问功能如下。

例如，你还继续计算吗（Y/N）？

Y：重新输入数据进行计算。

N：退出程序。

知识学习：break 语句和 continue 语句

知识学习

学习笔记

break 语句用于结束循环。如果在循环中遇到它，则直接结束循环，其示例如图 **4.11** 所示。

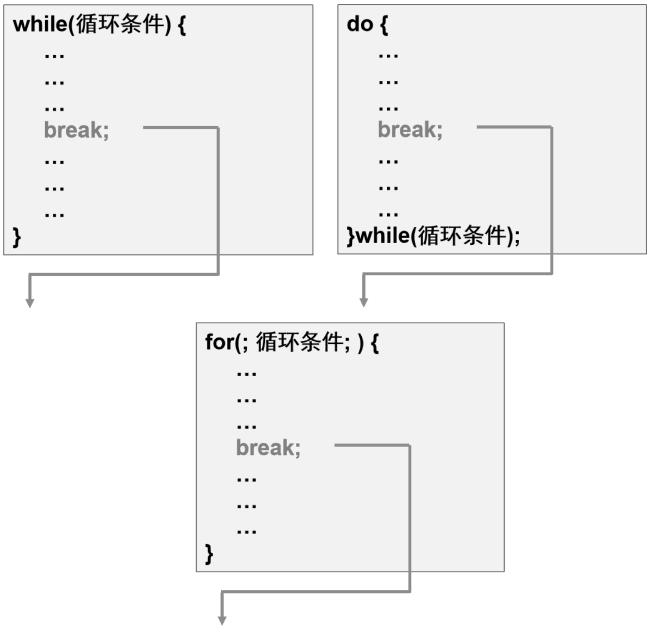


图 4.11　break 语句示例

continue 语句结束本次循环，直接进行下一次循环，其示例如图 **4.12** 所示。

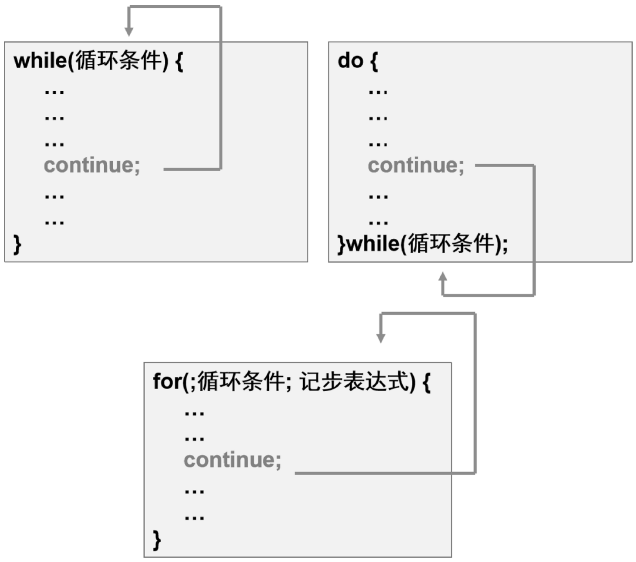


图 4.12　continue 语句示例

举例：

```
int main()
{
    int i;
    for(i=1;i<=10;i++)
    {
        printf("%d ",i);
    }
    return 0;
}
```

这个循环没有使用 break 语句和 continue 语句，循环 10 次后，输出结果为 1 2 3 4 5 6 7 8 9 10。

```
int main()
{
    int i;
    for(i=1;i<=10;i++)
    {
        if(i%2==0)
        {
            continue;
        }
        printf("%d",i);
    }
    return 0;
}
```

这个例子在循环中加了判断，如果这个数是偶数，则执行 continue 语句，否则输出这个数。

输出结果为 1　3　5　7　9。

举例：

```
int main()
{
    int i;
    for(i=1;i<=10;i++)
    {
        if(i%2==0)
        {
            break;
        }
        printf("%d\n",i);
    }

    return 0;
}
```

这个例子在循环中加了判断，如果这个数是偶数，则执行 break 语句，否则输出这个数。

第 1 次循环，i=1，条件不成立，输出 1；
第 2 次循环，i=2，条件成立，执行 break 语句结束循环了。
所以这个例子的输出结果为 1。

学习活动 3　制定方案

实现本任务方案

● **实现思路**

综合应用所学的循环知识，对之前实现的 BMI 程序进行优化，让其变得更具有交互性。BMI 程序的优化思路如图 4.13 所示。

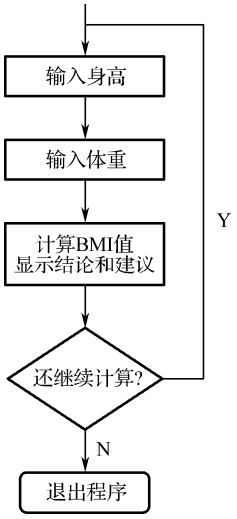


图 4.13　BMI 程序的优化思路

（1）增加循环；
（2）增加询问功能。
例如，你还继续计算吗（Y/N)?
Y：继续循环，重新输入新数据进行计算；
N：退出程序。

● **实现步骤**

（1）启动 CodeBlocks 软件；
（2）打开之前完成的 BMI 程序；
（3）按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）启动 CodeBlocks 软件，打开之前优化过的 BMI 程序。

（2）按实现思路进行程序优化，其代码如下。

```
int main()
{
    //定义保存数据的变量
    float height=0;
    float weight =0;
    float bmi=0;
    //在这里新加循环
    while(1)
    {
        system("cls");
        //输入数据
        printf("请输入你的身高（单位：米）:");
        scanf("%f",&height);   //输入数据到变量中
        printf("请输入你的体重（单位：千克）:");
        scanf("%f",&weight);   //输入数据到变量中
        //计算 BMI 值
        bmi=weight/(height*height);
        //显示数据
        printf("你的体质指数（BMI）为:%f \n",bmi);
        //判断 BMI 值的区间，给出结论和建议
        printf("----------------------------------------\n");
        if(bmi<18.5)
        {
            printf("结论：你的 BMI<18.5，体重过轻，属于低危险群体\n");
            printf("建议：多注意营养，增加体重！");
        }
        else if(bmi>=18.5 && bmi<24.0)
        {
            printf("结论：你的 18.5≤BMI<24，体重正常\n");
            printf("建议：别骄傲，要保持哦！");
        }
        else if(bmi>=24 && bmi<27)
        {
            printf("结论：你的 24≤BMI<27，体重过重，属于低危险群体\n");
            printf("建议：该减重啦！加强运动。");
        }
        else if(bmi>=27 && bmi<30)
        {
```

```
            printf("结论：你的 27≤BMI<30，轻度肥胖，属于中危险群体\n");
            printf("建议：控制饮食，加强运动，必须减重了。");
        }
        else if(bmi>=30 && bmi<35)
        {
            printf("结论：你的 30≤BMI<35，中度肥胖，属于重危险群体\n");
            printf("建议：减重！减重！减重！");
        }
        else if(bmi>=35)
        {
            printf("结论：你的 BMI≥35，病状肥胖，属于非常危险群体\n");
            printf("建议：在医生的指导下进行减重吧，我不能给你建议了！");
        }
        printf("\n----------------------------------------\n");

        //在这里增加询问
        char   yesorno;
        printf("请问还继续计算吗(y/n)?");
        scanf("%s",&yesorno);
        if(yesorno=='Y' || yesorno=='y')
        {
            continue;   //继续循环
        }
        else
        {
            break;   //退出循环，程序结束
        }
    }  //循环在这里结束
    return 0;
}
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

对以前任务实现的 BMI 程序进行了优化，让其变得更加有交互性，即计算完一个人的 BMI 值后，询问是否继续，如果回答是则继续；如果回答否则结束程序。任务实现的效果如图 4.14 所示。

```
请输入你的身高（单位：米）:1.7
请输入你的体重（单位：千克）:68
你的体质指数（BMI）为:23.529411
----------------------------------------
结论：你的 18.5≤BMI<24，体重正常
建议：别骄傲，要保持哦！
----------------------------------------
请问还继续计算吗(y/n)?
```

图 4.14　BIM 程序优化任务的运行效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

对以前任务实现的 BMI 程序进行了优化，让其变得更加有交互性，即计算完一个人的 BMI 值后询问是否继续，如果回答是则继续；如果回答否则结束程序。

● 技术层面

（1）继续完成计算功能（重复执行实现计算的代码），可以采用循环语句来实现。

（2）使用 while 循环相对来说要好些。因为用户继续操作的次数不确定，所以使用 while 进行死循环，在程序中做结束判断。

（3）循环加的位置。

（4）询问功能中根据判断，巧妙使用 continue 语句和 break 语句实现继续计算和结束程序。

● 课程思政

没有优化的 BMI 程序运行一次只能计算一个人的 BMI 值，大家试想一下，如果用户想计算多个人的 BMI 值，那就得不断地启动程序，这样就会很烦琐且低效。

所以，我们学了循环语句的知识后，就可以应用这些知识来解决这样的问题。程序计算完后，询问是否继续计算，如果用户不计算了，则选择 N；如果用户还要计算，则选择 Y，这样是不是感觉很亲切呀。所以作为一个程序员来说，在以后的职业生涯中，应该多站在用户的角度思考问题，这样开发出的产品才可能更适合于用户。

- **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

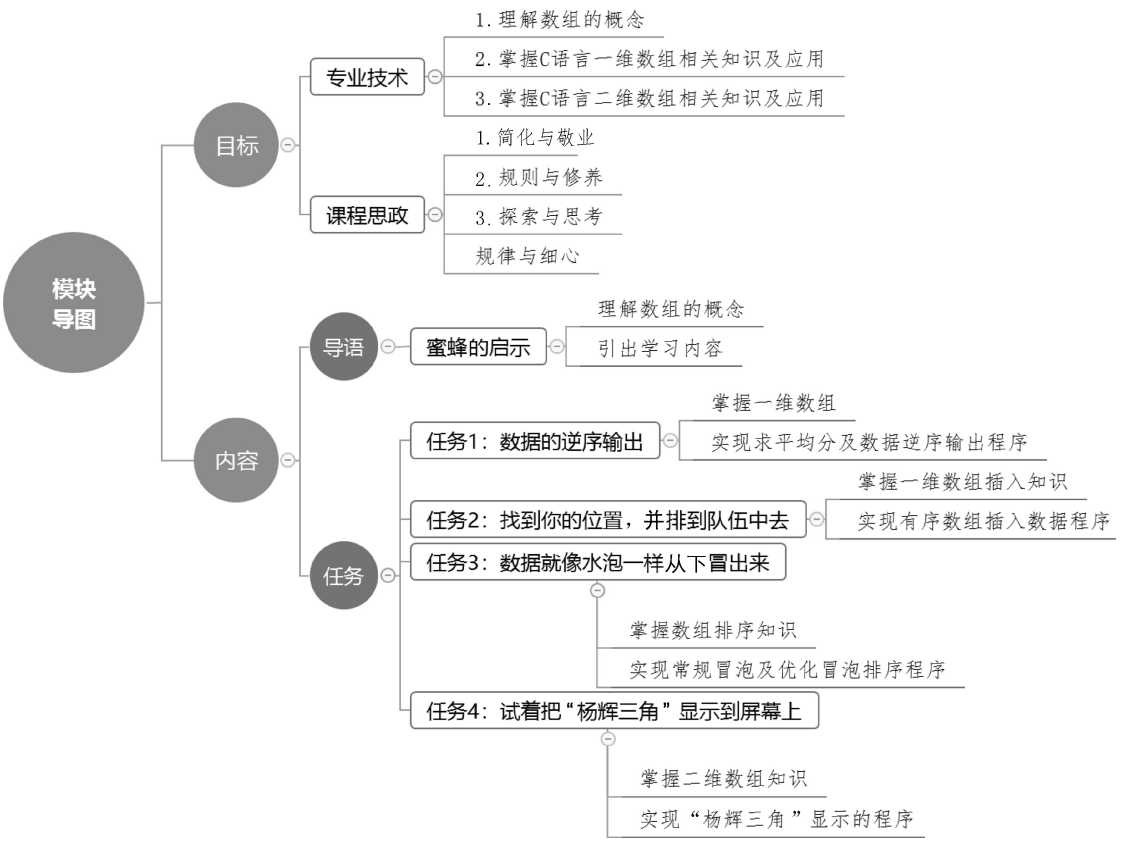
开启新征程

下一回：进入模块 5　C 语言程序中的数组应用

（介绍数组的相关知识）

模块 5

C 语言程序中的数组应用



项目导语：蜜蜂的启示

1. 从问题说起

求某个班学生数学成绩的平均分，我们可以这样编写程序（为了说明问题，假设这个班只有 5 名学生），程序实现的思路如下：

（1）定义 5 个变量，用于存放 5 名学生的数学成绩；

（2）键盘输入 5 名学生的成绩，并保存到 5 个变量中；

（3）将 5 个变量相加的和除以 5 计算平均分；

（4）输出计算结果，程序结束。

程序实现代码如下：

```
int main()
{
    //定义 5 个变量
    int stu1,stu2,stu3,stu4,stu5;
    int stuavg=0;   //存放平均分

    //键盘输入 5 个成绩
    printf("请输入这 5 名学生的数学成绩：");
    scanf("%d%d%d%d%d",&stu1,&stu2,&stu3,&stu4,&stu5);

    //计算平均分
    stuavg=(stu1+stu2+stu3+stu4+stu5)/5;
    //显示结果
    printf("平均分：%d",stuavg);
    return 0;
}
```

说明：

上面程序实现了求 5 名学生成绩的平均分。在程序中定义了 stu1、stu2、stu3、stu4、stu5 5 个变量，分别用于保存 5 名学生的成绩，然后相加求和后，除以 5 得到平均分。

发现什么问题了吗

如果这个班有 50 名学生，或者更多，该怎么办

当同一种类型的数据不止一个时，

如果使用一个个的变量来处理，则效率会很低。

2. 蜜蜂的启示

我们来观察一下大自然界中最辛勤的工程师——蜜蜂的生活。

蜜蜂通过蜂巢（见图 5.1）来存放酿造的蜂蜜，想存放更多的蜂蜜时，蜂巢就需要做大一

些，反之就做小一些，即根据存放蜂蜜的多少来决定蜂巢的大小。蜂巢一旦被破坏，所有的蜂蜜就都毁了。所以，蜂巢就是存放蜂蜜的容器，它里面存放的蜂蜜类型都是一样的。

牵一发则动全身

图 5.1　蜂巢

基于蜜蜂酿蜜的过程，能给我们带来什么启发呢？

处理学生的成绩时，由于每个学生成绩的数据类型都是一样的，是不是可以当作蜂巢来看待。每个班的学生人数就决定了蜂巢的大小（见图 5.2）。

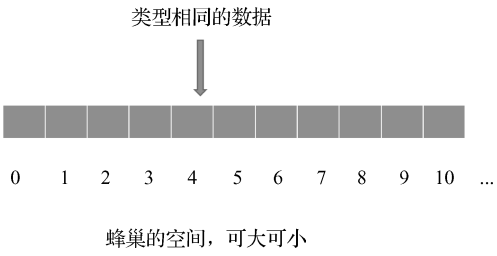


图 5.2　程序中的蜂巢——数组

数组就是程序中的“蜂巢”

数组：

（1）存放同一数据类型的数据集合；

（2）数组存放空间可大可小。

在程序中如何使用数组

为了介绍程序中的数组知识，本模块共设计以下 4 个任务。

任务 1：数据的逆序输出；

任务 2：找到你的位置，并排到队伍中去；

任务 3：数据就像水泡一样从下冒出来；

任务 4：试着把“杨辉三角”显示到屏幕上。

任务 1　数据的逆序输出

目标描述

任务描述
● 目标实现 （1）利用数组知识实现求某班 50 名学生某门课程的平均分。 输入：50 名学生的成绩。 输出：成绩的平均分。 （2）利用数组知识实现“数据逆序输出”。 输入：1 3 5 9 7 6 8 2 4 0。 输出：0 4 2 8 6 7 9 5 3 1。 ● 技术层面 掌握一维数组的定义及应用。 ● 课程思政 化繁为简。 敬业精神。

学习活动 1　接领任务

领任务单
● 任务确认 通过编写 C 语言程序，分别实现统计班级 50 名学生的平均分和将一组数据逆序输出。具体要求如下： （1）程序能正确统计，并输出 50 名学生成绩的平均分； （2）程序能将任意的 10 个整数按逆序排序并输出； （3）掌握 C 语言代码的使用规范性（变量取名及注释说明）； （4）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

本任务实现的内容：

（1）编程实现统计 50 名学生的平均分；

（2）编程实现任意输入 10 个整数保存到数组中，然后逆序输出。

统计 50 名学生成绩的平均分。这是一个数据集合的处理，上面已经介绍使用数组的方式来实现将变得更加高效。

将 10 个整数按输入的顺序进行逆序输出，同时也是使用数组来实现的最佳选择。

所以，本任务将开启对数组的学习与应用，下面就先从认识数组开始吧。

知识学习：C 语言的数组

学习笔记

我们知道一个变量只能存放单一数据。但在实际中，我们经常需要处理批量数据，如输入 30 名学生的成绩，并对这些成绩进行排序，就需要保存 30 个数据。就要在程序中使用 30 个变量来保存数据，程序会变得很烦琐。如果随着学生人数增加到 200 人，则需要大量修改程序，程序的扩展性会变得非常的差。

所以，数组是相同类型的数据的集合。通过定义一个可存放 200 个数据的数组就可以解决以上问题。

1．数组的特点

（1）名称：数组名；

（2）容量：存放多少个数据；

（3）元素：数组中的每一个数据；

（4）下标：访问数组中某一个元素的标号，数组的下标是从 0 开始的。

如下定义了一个 A 数组：

```
int A[5];
```

说明：

（1）数组名：A。

（2）容量：5，可以存放 5 个整型数据。

（3）元素：其中这 5 个整型数据的访问地址为 A[0]～A[4]。

A[0]	A[1]	A[2]	A[3]	A[4]

2．数组定义

数组必须先定义，后使用。

数组定义的语法如下：

```
Type arrayName [ arraySize ];
```

说明：

（1）Type：数据类型，可以是任意有效的数据类型。

（2）arrayName：数组名。

（3）arraySize：数组的大小决定可以存放多少个数据，必须是一个大于零的整数常量。

例如：

```
int    A[10];    //可以存放 10 个整数
float   B[30];  //可以存放 30 个小数
char   C[50];  //可以存放 50 个字符
```

3．初始化数组

初始化数组就是给数组赋初值的过程，有以下两种方式。

（1）逐个初始化数组。

int A[5] = {10,20,1,3,5};

提示：{}内值的数目不能大于在数组声明时[]内指定的元素数目。

A[0]	A[1]	A[2]	A[3]	A[4]
10	20	1	3	5

（2）省略数组的大小初始化数组。

int A[] = {10,20,1,3,5,100};

提示：如果省略数组的大小，数组的大小则为初始化时元素的个数。由于初始化时元素个数是 6，所以数组 A 的元素个数就是 6。

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
10	20	1	3	5	100

4．访问数组元素

数组声明后，就可以对其进行访问操作了。

如下定义了一个 A 数组，就可以存放 5 个整型数据：

```
int A[5];
```

操作：

A[0]=6;

A[4]=3;

A[2]=5;

int x=A[2];

A[0]	A[1]	A[2]	A[3]	A[4]
6		5		3

计算 x=?

因为 x=A[2]，而 A[2]=5，所以 x=5。

小结：

（1）数组定义后就拥有存放数据的元素，元素个数为数组的大小，元素的下标从 0 开始。定义一个存放 5 个数据的 A 数组，则 A 数组的元素个数为 5，下标从 0 开始，即 A[0]到 A[4]，共 5 个元素。

（2）数组的每一个元素相当于一个变量，可以进行各种操作。

（3）可以将数组的元素赋值给变量，也可以对其进行赋值等运算。

学习活动 3　制定方案

实现本任务方案
● 实现思路 本任务完成的实现思路如下：

1．利用数组知识实现某班 50 名学生某门课程的平均分

（1）定义一个能存放 50 个整数的一维数组；

（2）使用 for 循环进行 50 次循环；

（3）从键盘输入 50 名学生的成绩后，计算总分；

（4）循环结束后，计算平均分（总分/50）；

（5）输出结果，程序结束。

2．利用数组知识实现“数据逆序输出”

（1）定义一个能存放 10 个整数的一维数组；

（2）使用 for 循环从键盘输入 10 个整数；

（3）利用 for 循环 5 次交换数组中的值后，顺序输出；

（4）输出结果。

利用 for 循环输出数组的结果（直接将循环变量从最大值到 1 进行循环输出）

● **实现步骤**

（1）在 CodeBlocks 软件中创建两个新项目：

① 求平均分的项目名称为 getAVG。

② 数据逆序输出项目名称为 outArray。

（2）分别在各个项目的 main()中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

1．实现 50 名学生的平均分

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 getAVG。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()中按实现思路完成任务，其代码如下。

```
int main()
{
    //定义 1 个数组，可存放 50 名学生的成绩
    int stu[50];
    int stuavg=0;   //存放平均分
    int i;//循环变量
    for(i=0; i<50; i++)
    {
        //键盘输入学生成绩
        printf("请输入第 %d 个学生成绩：", i+1);
        scanf("%d",&stu[i]);
        //求和
        stuavg=stuavg+stu[i];
    }
```

```
    //计算平均分
    stuavg=stuavg/50;
    //显示结果
    printf("平均分：%d",stuavg);
    return 0;
}
```

2．数据逆序输出

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 outArray。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()中按实现思路完成任务，其代码如下。

```
int main()
{
    int i, a[10];
    for(i=0; i<10; i++)                    //输入数据保存到数组
    {
        scanf("%d",&a[i]);
    }
    for(i=9; i>=0; i--)                    //逆序输出
    {
        printf(" %d",a[i]);
    }
    return 0;
}
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

利用一维数组的知识，实现求某班 50 名学生某门课程的平均分，如图 5.3 所示。实现数据逆序输出的效果如图 5.4 所示。

```
请输入第 1 个学生成绩：60
请输入第 2 个学生成绩：60
请输入第 3 个学生成绩：70
请输入第 4 个学生成绩：80
请输入第 5 个学生成绩：90
请输入第 6 个学生成绩：100
请输入第 7 个学生成绩：65
请输入第 8 个学生成绩：76
请输入第 9 个学生成绩：88
请输入第 10 个学生成绩：90
请输入第 11 个学生成绩：100
请输入第 12 个学生成绩：30
请输入第 13 个学生成绩：56
请输入第 14 个学生成绩：76
请输入第 15 个学生成绩：87
请输入第 37 个学生成绩：78
请输入第 38 个学生成绩：90
请输入第 39 个学生成绩：76
请输入第 40 个学生成绩：68
请输入第 41 个学生成绩：69
请输入第 42 个学生成绩：7
请输入第 43 个学生成绩：78
请输入第 44 个学生成绩：78
请输入第 45 个学生成绩：89
请输入第 46 个学生成绩：98
请输入第 47 个学生成绩：97
请输入第 48 个学生成绩：96
请输入第 49 个学生成绩：95
请输入第 50 个学生成绩：78
平均分：73
```

图 5.3　求 50 名学生平均分的实现效果

```
1 2 3 4 5 6 7 8 9 10
10 9 8 7 6 5 4 3 2 1Press any key to continue
```

图 5.4　数据逆序输出的效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	开发环境安装与置换情况					
3	掌握知识的情况					
4	程序运行情况					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

利用一维数组的知识，实现了两个任务：

（1）求某班 50 名学生某课程的平均分；

（2）数据的逆序输出。

● 技术层面

介绍了一维数组的相关知识，包括数组的特点、数组的定义、数组的初始化、数组元素的操作等。

● 课程思政

通过本任务的学习，大家对一维数组的相关知识有了较全面的认识及理解，希望同学们不断加强练习，同时也能有更多的思考。

（1）化繁为简

求某班 50 名学生某门课程的平均分，只需要定义一个能存放 50 个数据的数组（一行代码）即可，省去了传统定义 50 个变量的思路；同时利用一个 for 循环完成对 50 名学生成绩的输入，提高了工作效率。

实现数据逆序输出时，将 10 个数据输入数组中，利用数组元素下标的连续性，在输出时将 for 循环初值从 9 开始递减到 0，即可完成数据的逆序输出。希望同学们在日常的生活中，多思考、多分析，尽量做到把繁杂的事情简单化。

（2）敬业精神

希望同学们要像蜜蜂一样，敬业于现在的学业，以及未来将从事的工作。

- **教学拓展**

利用一维数组的知识，尝试使用数组实现模块 3 中任务 3 的内容。

- **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

想知道，怎样在数组中插入数据吗

下一回：找到你的位置，并排到队伍中去

（解决数组插入数据的问题）

任务 2　找到你的位置，并排到队伍中去

目标描述

任务描述
● **编写程序实现** 在一个从小到大排序的有序数组中插入一个新数，并保证该数组有序。 ● **技术层面** 掌握一维数组的应用。 掌握插入新数据到现有数组中的原理。 ● **课程思政** 遵守社会规则。 加强自身修养。

学习活动 1　接领任务

领任务单
● 任务确认 通过编写 C 语言程序，实现在一个从小到大排序的有序数组中插入一个新数，并保证插入后该数组依然有序。 具体要求如下： （1）程序最终能在有序数组中正确插入数据，并保证数组仍然有序； （2）掌握 C 语言代码的使用规范（变量取名及注释说明）； （3）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

编写 C 语言程序，实现在一个从小到大排序的有序数组中插入一个新数（见图 5.5），并保证插入后该数组仍然有序，其分析如下：

（1）假如在一个具有 5 个元素的有序数组中，插入一个新数据 40；

（2）找到新数据 40 应该插入的位置；

（3）将该位置开始后的所有数据依次往后移位；

（4）腾出位置以插入这个新数据，使其仍是一个有序数组。

实现有序数组插入数据

（1）这个数组必须是有序的。

（2）定义数组时要多定义一个元素。

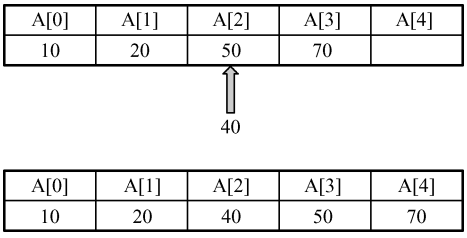


图 5.5　有序数组中插入数据

知识学习：有序数组插入数据介绍

知识学习

示例代码如下：

```
int main()
{
//定义一个可存放 6 个数的数组
    int a[6]= {1,3,5,7,9};
    int i, j, n;
```

学习笔记

```
        scanf("%d",&n);                    //输入要插入的数
        for(i=0; i<=4; i++)
        {
            if(n>a[4])                     //插入的这个数最大（特殊）
            {
                a[5]=n;
            }
            else if(n>=a[i] && n<=a[i+1])
            {
                for(j=5 ; j>i+1 ; j--)
                {
                    a[j]=a[j-1];           //依次往后退位
                }
                a[i+1]=n;                  //插入这个数
                break;                     //循环结束
            }
        }
        //显示输出
        for(i=0; i<6 ; i++)
        {
            printf(" %d", a[i]);
        }
        return 0;
    }
```

分析：

（1）如果插入的数据是 40，则直接写到最后一个元素（特殊）；

（2）如果插入的数据是 6，则找到插入的位置，也就是 i=2 时逐位后移动数据，所以，j 循环两次 5=>4，实现 a[5]=a[4],a[4]=a[3]；

（3）执行 a[i+1]=n；即 a[3]=6，完成数组的插入。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]
1	3	5	7	9	

学习活动 3　制定方案

实现本任务方案

● **实现思路**

（1）定义一个数组；

（2）给数组赋值，最后一个元素空着；

（3）输入要插入的数；

（4）完成数组的插入。

先要找到新数据应该插入的位置，然后将该位置开始以后的所有数据依次往后移位，腾出位置以插入这个新数据，并使其插入后仍然是一个有序数组。

● 实现步骤

（1）在 CodeBlocks 中创建一个新项目，项目名称为 insertArray。

（2）在 main()中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 insertArray。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()中按实现思路完成任务，其代码如下。

```
int main()
{
    int a[6]= {1,3,5,7,9};                  //定义一个可存放 6 个数的数组
            int i, j, n;
            scanf("%d",&n);                 //输入要插入的数
    for(i=0; i<=4; i++)
    {
        if(n>a[4])                          //插入的这个数最大（特殊）
        {
            a[5]=n;
        }
        else if(n>=a[i] && n<=a[i+1])
        {
            for(j=5 ; j>i+1 ; j--)
            {
                a[j]=a[j-1];                //依次往后退位
            }
            a[i+1]=n;                       //插入这个数
            break;                          //循环结束
        }
    }
    //显示输出
    for(i=0; i<6 ; i++)
    {
        printf(" %d", a[i]);
    }
    return 0;
}
```

分析：

（1）如果插入数据为 40，则插入的数据比现有所有数据都大（特殊），可以直接写到最后一个元素。

（2）如果插入数据为 6，则要找到插入的位置为 i=2，往后移动位置 j: 5,4，

a[5]=a[4]

a[4]=a[3]

数据移动后，可直接插入新数据 a[3]=6。

学习活动 5　测试验收

任务测试验收单

● 实现效果

在一个有序的一维数组中插入新数据，并保持该数组仍有序。

按制定的方案进行任务实现，在正确的情况下，其效果如图 5.6 所示。

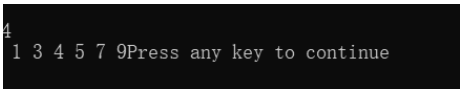


图 5.6　有序数组插入数据的实现效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量取名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

在一个有序的一维数组中插入新数据，并保持该数组仍有序。

● 技术层面

一维数组插入数据的应用与算法。

● **课程思政**

本任务实现了对一个有序的一维数组中插入新数据，并保持该数组仍有序，要完成本任务的前提是这个数组必须是有序的。这点同学们在定义数组时一定要注意，否则实现不了本任务。

在我们生活中也有无数的规则，如遵守交通规则，红灯停绿灯行；18 岁以后才可以去网吧之类的场所等。

我们应该做一个遵守社会规则、有修养的人。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

如果不是有序的数组该怎么办呢

下一回：数据就像水泡一样从下往上冒出来

（解决数组有序的问题）

任务 3　数据就像水泡一样从下往上冒出来

目标描述

任务描述
● **编写程序实现** 编写程序实现对保存 N 个整数的数组进行排序，排序指升序/降序。 ● **技术层面** 利用“冒泡排序法”实现对数组的排序。 掌握“冒泡排序法”的优化实现。 ● **课程思政** 勇于探索。 勤于思考。

学习活动 1　接领任务

领任务单
● 任务确认 编写 C 语言程序，使用“冒泡排序法”对数组进行排序。 具体要求如下： （1）程序最终能正确展示数组排序后的结果； （2）使用传统“冒泡排序法”和优化“冒泡排序法”分别实现； （3）掌握 C 语言代码的使用规范性（变量取名及注释说明）； （4）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

编写 C 语言程序，利用“冒泡排序法”对数组进行排序。

知识学习：冒泡排序算法

知识学习

1．原理

比较相邻两个元素，如果第 1 个比第 2 个大，则交换两个数（升序），并依次对每一对相邻元素进行同样操作，直到最后一对，找出最大的数放到最后一个元素中。持续对越来越少的元素重复上面的步骤，直到没有任何一对数字需要比较，如图 5.7 所示。

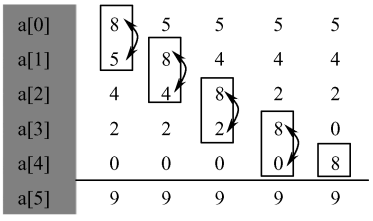


图 5.7 “冒泡排序法”的原理

说明：

（1）对一组数中的相邻两个数进行比较、交换，将最大（小）数交换至尾（首）部，即完成了一次冒泡排序。

（2）要想对 N 个数字进行排序，循环 N 次即可。

学习笔记

2．举例（见图 5.8）

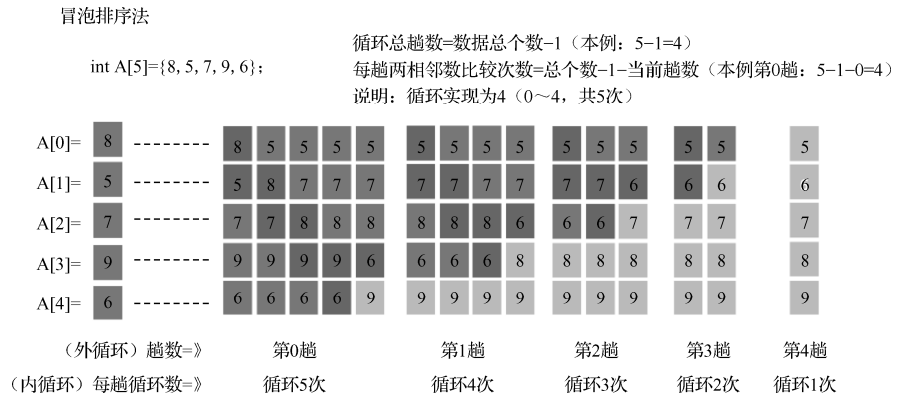


图 5.8　冒泡排序法的执行过程

说明：

（1）定义一个存放 5 个数据，且无序的数组 A；

（2）“冒泡排序法”的执行过程。

采用双重循环实现，外循环决定循环趟数，内循环决定每趟循环中相邻数比较的次数。（数组下标从 0 开始）

第 0 趟：内循环 5 次，得到这一趟最大数为 9；

第 1 趟：内循环 4 次，得到这一趟最大数为 8；

第 2 趟：内循环 3 次，得到这一趟最大数为 7；

第 3 趟：内循环 2 次，得到这一趟最大数为 6；

第 4 趟：内循环 1 次，得到这一趟最大数为 5。

循环结束，实现排序过程。

小结：

（1）循环总趟数=数据总个数-1（本例：5-1=4）

（2）每趟两相邻数比较次数=总个数-1-当前趟数（本例第 0 趟：5-1-0=4）

特别说明：循环实现为 4（0～4，共 5 次，因为数组下标从 0 开始）

学习活动 3　制定方案

实现本任务方案

● **实现思路**

采用“冒泡排序法”实现对数组排序，涉及传统算法及优化算法，大致思路如下：

（1）定义一个能存放 10 个整数的一维数组，并存放 10 个无序的数。

（2）使用双重循环利用“冒泡排序法”进行操作。

外循环决定总共循环多少趟（总个数-1），

内循环实现两相邻数比较（每趟循环次数：总个数-1-当前趟数）

（3）输出有序数列。

● 实现步骤

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 maoPao。

（2）在 main()中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

1．传统“冒泡排序法”实现对数组排序的代码

```
int main()
{
    int a[10]= {12,43,9,13,67,98,101,89,3,35};      //10 个数的无序数列
    int i, j, t;
    printf("此程序使用“冒泡排序法”排列无序数列！\n");
    //冒泡排序
    for(i=0; i<10-1; i++)                           //趟数：总个数-1
    {
        for(j=0; j<10-1-i; j++)                     //每趟比较次数：总个数-1-当前趟数
        {
            if(a[j]>a[j+1])                         //两相邻数比较，交换两个数的位置（升序）
            {
                t=a[j+1];
                a[j+1]=a[j];
                a[j]=t;
            }
        }
    }
    printf("排列好的数列是：\n");                   //输出排列好的数列
    for(i=0; i<10; i++)
    {
        printf("%d ",a[i]);
    }
    return 0;
}
```

2．“冒泡排序法”优化方法一

按照“冒泡排序法”可以实现对数组的排序。如果在没有循环完总趟数时数组就已经有序了，就没必要再做后面的比较了，下面就对其进行优化。

如图 5.9 所示，在第 2 趟时，数组就已经有序，因此就没必要进行后面的计算了。

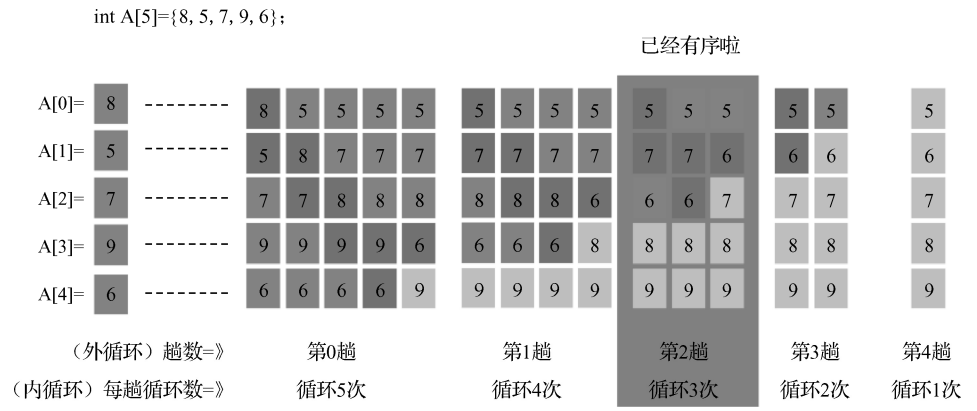


图 5.9　数组已有序

优化思路

（1）判断数列是否已有序。

（2）做出标记。

（3）剩下的几轮排序就可以不必执行，提早结束工作了。

```
int main()
{
    int a[10]= {12,43,9,13,67,98,101,89,3,35};      //10 个数的无序数列
    int i, j, t;
    int isSorted=0;                                  //有序标记
    printf("此程序使用“冒泡排序法”排列无序数列！\n");
    //冒泡排序
    for(i=0; i<10-1; i++)                            //趟数：总个数-1
    {
        isSorted=1;                                  //假设为有序
        for(j=0; j<10-1-i; j++)                      //每趟比较次数：总个数-1-当前趟数
        {
            if(a[j]>a[j+1])                          //两个相邻数比较，交换两个数的位置（升序）
            {
                t=a[j+1];
                a[j+1]=a[j];
                a[j]=t;
                isSorted=0;                          //有交换，说明无序
            }
        }
        if(isSorted)                                 //条件成立，说明没有进行过交换，已有序了
        {
            break;                                   //结束循环，提前收工
        }
    }
}
```

3.“冒泡排序法”优化方法二

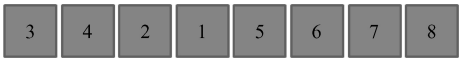
假如有以下这样一个数组：

这个数列的特点是前半部分为（3，4，2，1）无序，后半部分为（5，6，7，8）有序（升序）。并且后半部分的元素已是数列最大值。

思考一下，如何进一步优化

分析：

按照现有的逻辑，有序区的长度和排序的轮数是相等的。例如，第 1 趟排序过后的有序区长度是 1，第 2 趟排序过后的有序区长度是 2……



实际上，数列真正的有序区可能会大于这个长度，如例子中仅第 2 趟，后面 5 个元素实际都属于有序区，因此对后面的元素进行比较是没有意义的。

思考：

在每趟排序的最后，都可以记录下最后一次元素交换的位置，那个位置就是无序数列的边界，再往后就是有序区了。

优化思路：

问题的关键在于对数列有序区的界定。

优化后的代码如下：

```
int main()
{
    int a[8]= {3,4,2,1,5,6,7,8};              //8 个数的无序数列
    int i, j, t;
    int isSorted=0;
    int lastExchangeIndex = 0;                 //记录最后一次交换的位置
    int sortBorder = 7 ;                       //无序数列的边界，每次只需要比较到这里
    for(i=0; i<8-1; i++)                       //趟数：总个数-1
    {
        isSorted=1;
        for(j=0; j<sortBorder; j++)            //每一趟比较次数：无序的边界
        {
            if(a[j]>a[j+1])                    //两个相邻数比较，交换两个数的位置（升序）
            {
                t=a[j+1];
                a[j+1]=a[j];
                a[j]=t;
                isSorted=0;
                lastExchangeIndex = j;         //记录最后一次元素交换的位置
            }
        }
        sortBorder = lastExchangeIndex;        //记录无序的边界
```

```
            if(isSorted) {
                break;
            }
        }
    }
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

利用“冒泡排序法”实现对数组进行排序。按制定方案进行任务实现，在正确的情况下，任务实现的效果如图 5.10 所示。

图 5.10 “冒泡排序法”的运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量取名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

利用“冒泡排序法”实现对数组进行排序。

● **技术层面**

详细介绍了“冒泡排序法”的算法实现。

“冒泡排序法”优化一解决了中途数组已经有序。

“冒泡排序法”优化二解决了有序与无序的边界，更大地优化了算法。

● **课程思政**

通过本任务的学习，同学们掌握了“冒泡排序法”的相关知识。

同时，对“冒泡排序法”又进行了二次优化，说明做任何事，我们应保持“没有最好，只有更优。”的心态，让自己成为一个勇于探索、勤于思考的人。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

玩了这么久的一维数组，有没有二维数组

下一回：试着把“杨辉三角”显示到屏幕上

（二维数组的应用）

任务 4　试着把“杨辉三角”显示到屏幕上

目标描述

任务描述

● **编写程序实现**

按“等腰三角形”的形态显示出“杨辉三角”前 10 行数据，如图 5.11 所示。

```
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5   10  10  5   1
        1   6   15  20  15  6   1
      1   7   21  35  35  21  7   1
    1   8   28  56  70  56  28  8   1
  1   9   36  84  126 126 84  36  9   1
```

图 5.11　“杨辉三角”显示示意

● **技术层面**

掌握二维数组的定义及应用。

掌握“杨辉三角”的计算方法。

● **课程思政**

做事细心。

探寻规律。

学习活动 1　接领任务

领任务单
● 任务确认 利用二维数组的知识实现显示“杨辉三角”的前 10 行数据。 具体要求如下： （1）程序最终能正确展示“杨辉三角”的前 10 行数据； （2）掌握 C 语言代码的使用规范（变量取名及注释说明）； （3）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

利用二维数组的知识实现显示“杨辉三角”的前 10 行数据。

那就先来认识一下杨辉这个人吧。

杨辉，字谦光，南宋时期钱塘（今浙江省杭州）人，数学家。1261 年，他在所著的《详解九章算法》中提出了“杨辉三角”。在欧洲，帕斯卡于 1654 年发现这个规律，所以这个表又被称为“帕斯卡三角形”。

发现什么问题了吗

我国的数学家杨辉发现这个规律比帕斯卡要早 393 年。

我们来了解一下“杨辉三角”吧（见图 5.12）。

剖析

（1）每行数的个数与行数对应，如第 1 行只有 1 个数，第 5 行就得有 5 个数；

（2）每行的第 1 个数和最后 1 个数是 1；

（3）每行除了第 1 个数和最后 1 个数之间的数为 F(i,j)=F(i−1,j−1)+F(i−1,j)。其中 i 和 j 分别为行数和列数

如 F(4,2)=F(3,1)+F(3,2)。

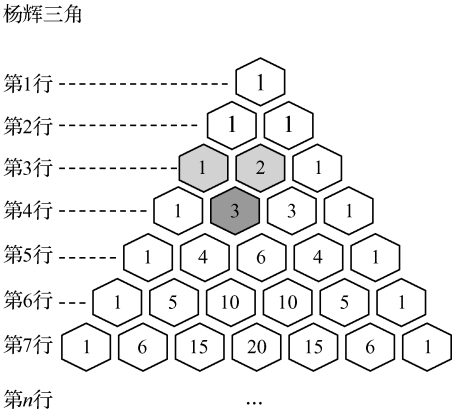


图 5.12　“杨辉三角”示意

行和列

（1）空间：二维空间；

（2）程序：二维数组。

知识学习：C 语言的二维数组

知识学习

学习笔记

在很多实际问题中，数据的逻辑结构是二维或多维的，C 语言允许构造多维数组。多维数组元素有多个下标。

二维数组就是由行和列构成的一个二维的存储空间。

行＼列	0	1	2
0	0	0	0
1	0	0	0
2	0	0	0
3	0	0	0

说明：

（1）行和列构成一个存储空间；

（2）容量可以存放数据的空间个数，由行×列构成。

1．二维数组定义

数组必须先定义，再使用。

数组定义的语法：

```
dataType arrayName[num1][num2];
```

说明：

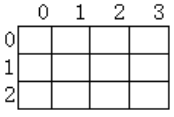
（1）dataType：表示数据类型；

（2）arrayName：表示数组名；

（3）num1 和 num2：表示数组的行下标和列下标。

如 int temp[3][4];

定义了一个 3 行 4 列的二维数组（或称矩阵），数组名为 temp，共有 12 个元素。



2．二维数组的初始化：

（1）按行分段赋值。

int arr[3][3] = {{1,2,3}, {4,5,6}, {7,8,9}};

初始化后如下所示。

1 2 3

4 5 6

7 8 9

（2）按行连续赋值。

int arr[3][3] = {1, 2, 3, 4, 5, 6, 7, 8, 9};

初始化后如下所示。

1 2 3

4 5 6

7 8 9

（3）可以只对部分元素赋初值，未赋初值的元素自动取 0 值。

int a[3][3] = {{1},{2},{3}};

初始化后如下所示：

1 0 0

2 0 0

3 0 0

（4）还可以这样：

int a[3][3] ;

a[0][0]=1;

初始化后如下所示：

1 0 0

0 0 0

0 0 0

3．二维数组的引用

二维数组的元素也称为双下标变量，其表示形式如下：

数组名[下标][下标]

如定义一个 int x[3][2]的数组，则 x 数组拥有 6 个元素，在程序中可以对其进行赋值和引用。

```
int main()
{
    int x[3][2];
    x[1][1]=0;
    x[2][1]=3;
    x[0][0]=x[1][1]+x[2][1];
    return 0;
}
```

学习活动 3　制定方案

实现本任务方案

● **实现思路**

（1）定义一个 10×10 的二维数组，形成一个方阵。

（2）采用二重循环利用“杨辉三角”的算法，计算出数据并存储到二维数组对应位置上。

① 每行数的个数与行数对应；

② 每行的第 1 个数和最后 1 个数是 1；

③ 每行除了第 1 个数和最后 1 个数之间的数为 F(i,j)=F(i-1,j-1)+F(i-1,j)。

其中 i 和 j 为行数和列数。

	0	1	2	3	4	5	6	7	8	9
0	1									
1	1	1								
2	1	2	1							
3	1	3	3	1						
4	1	4	6	4	1					
5	1	5	10	10	5	1				
6	1	6	15	20	15	6	1			
7	1	7	21	35	35	21	7	1		
8	1	8	28	56	70	56	28	8	1	
9	1	9	36	84	126	126	84	36	9	1

（3）按要求显示二维数组中的数据。

● 实现步骤

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 yangHuiSJ；

（2）在 main()中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 yangHuiSJ。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()按实现思路完成任务，其代码如下。

```
int main()
{
    //定义行数
    int n=10;
    int nums[n][n];

    int i, j; //循环变量
    // 计算“杨辉三角”
    for(i=0; i<n; i++)
    {
        //每行的第 1 个数为 1
        nums[i][0] = 1;
        //每行的最后 1 个数为 1
        nums[i][i] = 1;
        //计算每行除第 1 个数和最后 1 个数以外的数
        for(j=1; j<i; j++)
        {
            //等于上一行的前一列数+上一行的当前列的数之和
            nums[i][j] = nums[i-1][j-1] + nums[i-1][j];
        }
```

```
        }
        //显示输出
        for(i=0; i<n; i++)
        {
            for(j=0; j<n-i-1; j++)
            {
                printf("    ");   //3 个空格
            }
            for(j=0; j<=i; j++)
            {
                printf("%-5d ", nums[i][j]);
            }
            printf("\n");   //换行
        }
        return 0;
    }
```

（4）运行程序。

学习活动 5　测试验收

任务测试验收单

● 实现效果

利用二维数组实现了“杨辉三角”前 10 行数据的计算与显示。

按制定方案进行任务实现，在正确的情况下，任务实现的效果如图 5.13 所示。

```
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5   10  10  5   1
        1   6   15  20  15  6   1
      1   7   21  35  35  21  7   1
    1   8   28  56  70  56  28  8   1
  1   9   36  84  126 126 84  36  9   1
```

图 5.13 “杨辉三角”的运行效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量取名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

利用二维数组实现了“杨辉三角”前 10 行数据的计算与显示。

● **技术层面**

二维数组相关知识。

“杨辉三角”的计算方法。

● **课程思政**

通过本任务的学习，希望同学们在日常的生活学习中能养成以下好习惯。

找规律：多思考、找出事物的规律。

重细节：细心做事，细节往往能够决定成败。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

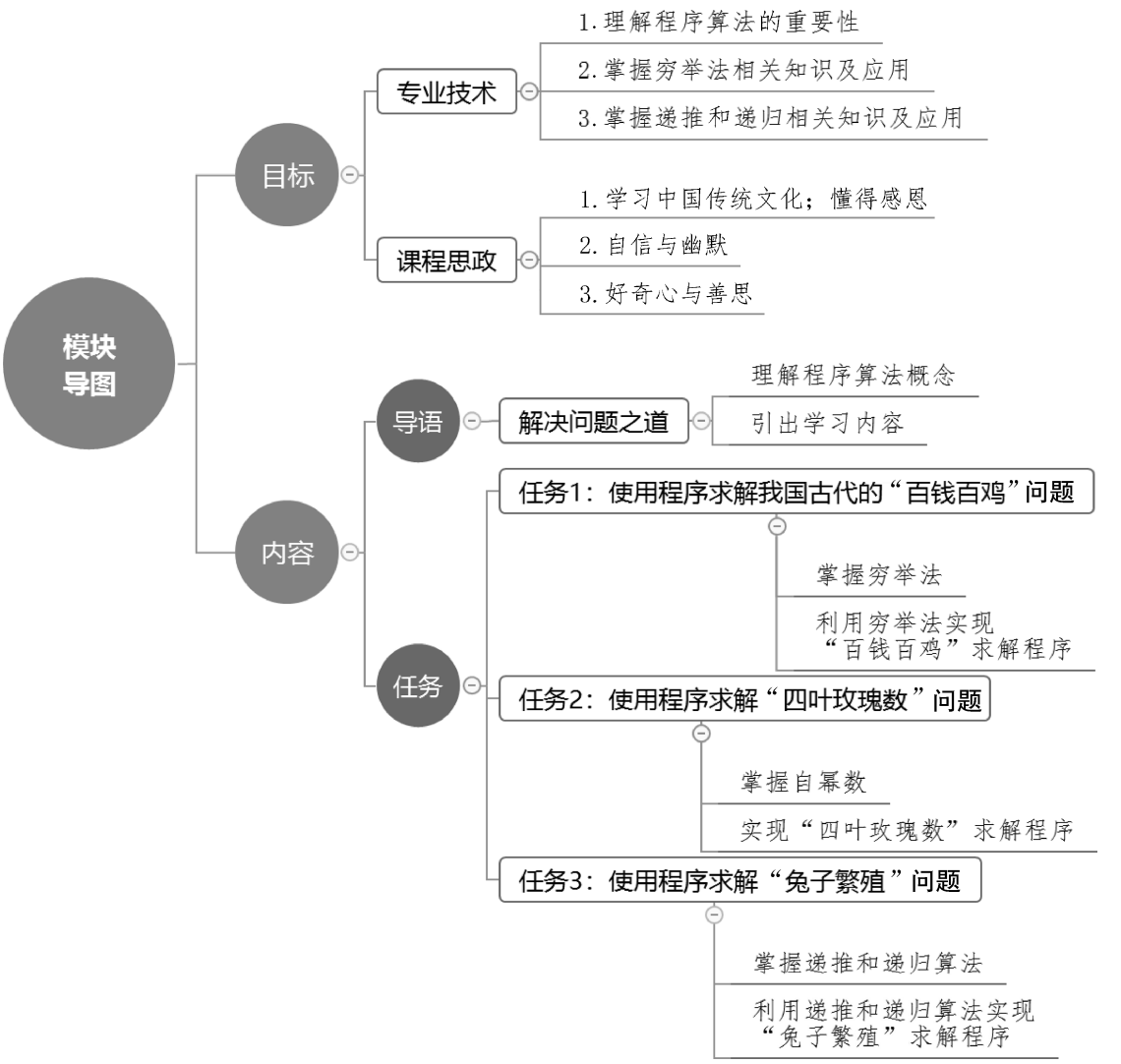
是不是感觉到实现思路的重要性啦

下一回：进入模块 6　C 语言程序中的算法应用

（介绍一些常用的算法）

模块 6

C 语言程序中的算法应用



项目导语：解决问题之道

1. 解决问题之道

以计算机的角度去思考问题的解法之道，称为算法。

那么什么是算法呢？

简单地说，算法就是解决问题的方法和步骤。

那么什么是程序呢？

程序是为了解决特定问题的计算机语言有穷操作规则的有序集合。

程序=数据结构+算法

所以，算法就是使用程序去解决实际问题的方法与实现步骤。

2. 生活中的算法

日常的生活中我们也会遇到算法，如做菜的算法如下：

（1）把锅放在火上；

（2）放 2 两油，烧热，加少许盐、蒜和花椒爆香；

（3）将切好的肉片，煎炒到适当的时候放水；

（4）煮沸时，加入少量秘制的调味品；

（5）放少量酱油，拌匀，就可出锅了。

还有我们去办事时，会有具体的办事流程，这也是算法，等等。

3. 程序算法

如求 1+2+3+…+100 的程序，算法可以设计如下。

算法表达方式一：

设变量 X 表示加数，Y 表示被加数。

（1）将 1 赋值给 X；

（2）将 2 赋值给 Y；

（3）将 X 与 Y 相加，结果存放在 X 中；

（4）将 Y 加 1，结果存放在 Y 中；

（5）若 Y 小于或等于 100，就转到（3）继续执行；否则，算法结束，结果为 X。

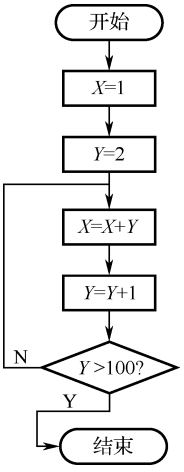


图 6.1　算法流程图

算法表达方式二：

采用流程图的形式来设计算法，如图 6.1 所示。

流程图

采用美国国家标准协会（American National Standard Institute，ANSI）规定的一组图形符号来表示算法。

流程图可以很方便地表示顺序、选择和循环的结构。

用流程图表示的算法不依赖于任何具体的计算机和计算机程序设计语言。

流程图使用的符号及说明如图 6.2 所示。

图 6.3 就是使用流程图设计的添加信息流程。

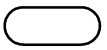
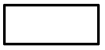
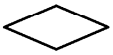
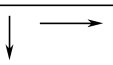
符号	符号名称	功能说明
	起止框	表示算法的开始和结束
	处理框	表示执行一个步骤
	判断框	表示要根据条件选择执行路线
	输入/输出框	表示需要用户输入或由计算机自动输出的信息
	流程线	指示流程的方向

图 6.2　流程图符号与功能

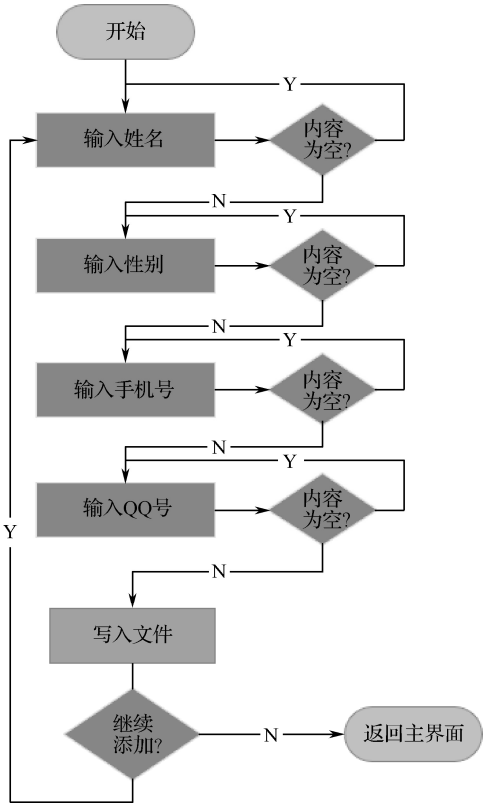


图 6.3　添加信息的流程

开启 C 语言程序中的算法应用

下面介绍算法基础的相关知识，本模块共设计以下 3 个任务。

任务 1：使用程序求解我国古代的“百钱百鸡”问题；

任务 2：使用程序求解“四叶玫瑰数”问题；

任务 3：使用程序求解“兔子繁殖”问题。

任务 1　使用程序求解我国古代的“百钱百鸡”问题

目标描述

任务描述
● **编写程序实现** 编写 C 语言程序，求解我国古代的“百钱百鸡”的问题。 ● **技术层面** 掌握穷举法。 ● **课程思政** 学习中国的传统文化。

家国情怀。

善思。

学习活动 1　接领任务

领任务单

● 任务确认

编写 C 语言程序，实现“百钱百鸡”问题。

具体要求如下：

（1）程序最终能正确展示求解结果；

（2）掌握 C 语言代码的使用，（变量取名及注释说明）；

（3）程序能正确运行，并具有可扩展性。

● 确认签字

学习活动 2　分析任务

编写 C 语言程序，实现“百钱百鸡”问题。

（1）认识“百钱百鸡”问题。

公元 5 世纪末，我国古代数学家张丘建在《算经》中，提出了这样的一个问题：

“鸡翁一，值钱五，鸡母一，值钱三，鸡雏三，值钱一。百钱买百鸡，问鸡翁、鸡母、鸡雏各几何？”

即公鸡 5 元/只，母鸡 3 元/只，小鸡 1 元/3 只。用 100 元买 100 只鸡，求公鸡、母鸡、小鸡各买几只？如图 6.4 所示。

图 6.4　“百钱百鸡”图意

知识学习：穷举法

穷举法：

（1）穷举法的基本思想是，根据题目的部分条件确定答案的大致范围；

（2）并在此范围内对所有可能的情况逐一验证，直到全部情况验证完毕；

（3）若某个情况验证符合题目的全部条件，则为本问题的一个解；

（4）若全部情况验证后都不符合题目的全部条件，则本题无解。

使用穷举法来解“百钱百鸡”问题：

设：公鸡 a 只，母鸡 b 只，小鸡 c 只，根据问题可得出下面的约束方程：

（1）a+b+c=100 （说明数量之和为 100 只）

（2）5a+3b+c/3=100（说明花钱总和为 100 元）

（3）c%3=0（说明小鸡的数量必须是 3 的倍数）

穷举范围：

公鸡：0≤a≤100

母鸡：0≤b≤100

小鸡：0≤c≤100

判断：

在穷举范围中只要同时满足以上 3 个条件则为解，并输出结果。

学习笔记

学习活动 3　制定方案

实现本任务方案

● **实现思路**

通过对本任务的分析及相关知识学习，制定方案如下：

（1）定义分别代表公鸡、母鸡、小鸡的变量 a，b，c；

（2）第一层 for 循环从 0～100 来穷举公鸡数；

（3）第二层 for 循环从 0～100 来穷举母鸡数；

（4）第三层 for 循环从 0～100 来穷举小鸡数；

（5）在第三层 for 循环中判断条件。

如果条件成立，则输出结果。

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 bqbj。

（2）分别在项目的 main()中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 bqbj。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()中按实现思路完成任务，其代码如下：

```
int main()
{
   int a,b,c;                                  //定义三种鸡
   int n=100;                                  //定义鸡的总数和钱的总数
       for (a = 0;a <= n;a++)                  //枚举公鸡
       {
             for (b = 0;b <= n;b++)            //枚举母鸡
             {
                   for (c = 0;c <= n;c++)      //枚举小鸡
                   {
                         if((a + b + c == n) && (5*a+3*b+c/3==n)&& (c%3==0))
                               printf("公鸡:%d,  母鸡:%d,小鸡:%d\n",a,b,c);
                   }
             }
       }
}
```

但这里发现有问题。

下面分析一下实现的代码：

（1）3 个循环 100 次的循环嵌套；

（2）循环次数（100×100×100=1 000 000），如图 6.5 所示。

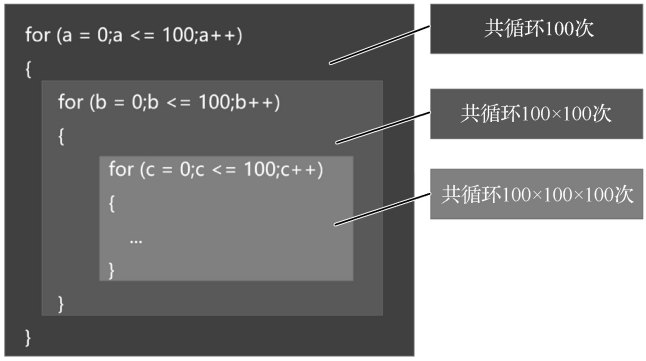


图 6.5　穷举法循环示意

也就是说，使用上面的程序需要 100 万次循环才能求得结果。

算法指标

空间复杂度

时间复杂度

有问题不可怕，我们来改进它。

分析：

公鸡 5 元/只，母鸡 3 元/只，小鸡 1 元/3 只。用 100 元买 100 只鸡，求公鸡、母鸡、小鸡各买几只?

开启“小宇宙”

考虑到 n 元（n=100)只能买到 n/5 只公鸡或 n/3 只母鸡，而小鸡的数目又取决于公鸡和母鸡的只数，所以，

（1）只需 2 个循环嵌套（一个穷举公鸡，另一个穷举母鸡，然后，用总数−公鸡数−母鸡数=小鸡数）。

（2）外循环 n/5 次，内循环 n/3 次。

改进后只需要(100/5)×(100/3)=660 次循环就可求得结果，优化了算法，其参考代码如下：

```
int main()
{
    int a,b,c;                    //定义 3 种鸡
    int n=100;                    //定义鸡的总数和钱的总数
    int i=n/5,j=n/3;              //公鸡和母鸡的最大可能数

    for (a = 0; a <= i; a++)      //枚举公鸡
    {
        for (b = 0; b <= j; b++)  //枚举母鸡
        {
            c=n-a-b;//小鸡
            if((a + b + c == n) && (5*a+3*b+c/3==n)&& (c%3==0))
                printf("公鸡:%d,母鸡:%d,小鸡:%d\n",a,b,c);
        }
    }
}
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

编写 C 语言程序，实现“百钱百鸡”问题。按制定方案进行任务实现，在正确的情况下，任务实现的效果如图 6.6 所示。

```
公鸡：0,母鸡：25,小鸡：75
公鸡：4,母鸡：18,小鸡：78
公鸡：8,母鸡：11,小鸡：81
公鸡：12,母鸡：4,小鸡：84

Process returned 101 (0x65)   execution time : 0.521 s
```

图 6.6 “百钱百鸡”的运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	开发环境安装与置换情况					
3	掌握知识的情况					
4	程序运行情况					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

利用穷举法的思路，求解了我国古代的“百钱百鸡”问题。

（1）传统求解；

（2）优化求解。

● **技术层面**

分析问题找出对应的条件，利用之前所学知识进行实现。

● **课程思政**

通过本任务的学习，实现了我国古代的“百钱百鸡”问题。同学们在不断加强练习的同时也要有更多的思考。

（1）学习中国的传统文化。

如原文“鸡翁一，值钱五，鸡母一，值钱三，鸡雏三，值钱一。百钱买百鸡，问鸡翁、鸡母、鸡雏各几何？”是典型的文言文写法，希望同学们能感受到我国古汉语的魅力。

（2）懂得感恩。

从文中对鸡翁、鸡母、鸡雏的描述，让我们联想到家、亲人的爱，正是这份爱成就了你

的今天，所以大家要学会感恩，感谢家人给予的爱。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

这次求“百钱百鸡”，下次做什么呢

你知道“四叶玫瑰数”吗

（自幂数的自恋）

任务 2　使用程序求解“四叶玫瑰数”问题

目标描述

任务描述
● **编写程序实现** 编写程序求解“四叶玫瑰数”问题。 ● **技术层面** 掌握自幂数的定义。 掌握“四叶玫瑰数”的计算方法。 ● **课程思政** 自信与幽默。

学习活动 1　**接领任务**

领任务单
● **任务确认** 编写 C 语言程序，求解“四叶玫瑰数”。 具体要求如下： （1）程序最终能正确展示求解结果； （2）掌握 C 语言代码的使用规范（变量取名及注释说明）；

（3）程序能正确运行，并具有可扩展性。

● 确认签字

学习活动 2　分析任务

编写 C 语言程序，求解“四叶玫瑰数”问题。

那么什么是“四叶玫瑰数”问题呢？

知识学习：自幂数

知识学习

学习笔记

1．自幂数

“四叶玫瑰数”是自幂数的一种。

自幂数指每个位数字的 *n* 次幂之和等于它本身。

例如：1^3 + 5^3+ 3^3 = 153。

那不同的自幂数有什么名称吗？

一位自幂数：独身数。

两位自幂数：没有。

三位自幂数：水仙花数。

四位自幂数：四叶玫瑰数。

五位自幂数：五角星数。

六位自幂数：六合数。

七位自幂数：北斗七星数。

八位自幂数：八仙数。

九位自幂数：九九重阳数。

十位自幂数：十全十美数。

例如：

四叶玫瑰数：$1634=1^4+6^4+3^4+4^4$；

水仙花数：$153=1^3+5^3+3^3$；

北斗七星数：$1741725=1^7+7^7+4^7+1^7+7^7+2^7+5^7$。

***n* 位数，就是 *n* 次幂**

2．“四叶玫瑰数”求解说明

“四叶玫瑰数”是一个 4 位数的整数，关键在于先把这个 4 位数的个位、十位、百位、千位取出来，再进行 4 次幂之和判断是不是等于本身？

获取四位数的个位、十位、百位、千位：

千位=数/1000;　　//获取千位

百位=数/100%10;　　//获取百位

```
十位=数/10%10;          //获取十位
个位=数%10;             //获取个位
```

学习活动 3　制定方案

实现本任务方案

● 实现思路

通过对本任务的分析及相关知识学习，制定方案如下：

（1）定义分别保存个位、十位、百位、千位的变量；

（2）使用 for 循环实现所有 4 位数的列举；

（3）获取 4 位数的个位、十位、百位、千位；

（4）对 4 次幂之和判断是不是等于本身，如果是则输出。

● 实现步骤

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 rose。

（2）在 main.c 文件中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 rose。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()中按实现思路完成任务，参考代码如下。

```
int main()
{
    int i;                                  //循环变量
    int gewei,shiwei,baiwei,qianwei;        //用于记录个位、十位、百位、千位上的数
    int temp=0;                             //临时用于记录表达式的值

    printf("四叶玫瑰数有：\n");

    for(i=1000; i<=9999; i++)
    {
        qianwei=i/1000;                     //获取千位
        baiwei=i/100%10;                    //获取百位
        shiwei=i/10%10;                     //获取十位
        gewei=i%10;                         //获取个位
        //计算个位、十位、百位、千位数的 4 次方之和
        temp=gewei*gewei*gewei*gewei+shiwei*shiwei*shiwei*shiwei
            +baiwei*baiwei*baiwei*baiwei+qianwei*qianwei*qianwei*qianwei;

        if(i==temp)
```

```
            {
                printf("%d \t",i);
            }
        }
        return 0;
    }
```

（4）运行程序。

学习活动 5　测试验收

任务测试验收单

● 实现效果

编写 C 语言程序，实现对“四叶玫瑰数”这种自幂数的求解。

按制定的方案进行任务实现，在正确的情况下，任务实现的效果如图 6.7 所示。

```
四叶玫瑰数有:
1634    8208    9474    Press any key to continue
```

图 6.7 “四叶玫瑰数”任务的运行效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量取名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

实现对“四叶玫瑰数”这种自幂数的求解。

● **技术层面**

对问题进行分析，设计出对应的求解算法。

● **课程思政**

通过本任务实现的学习，同学们除了好好训练，还应该充满自信与幽默。

如“我现在的主要任务是好好学习，虽然我还没能力送你 999 朵玫瑰，但我可以用程序写出‘四叶玫瑰数’送你呀！”，哈哈。

这样既充分体现了自信的自己，也表现出了程序员的幽默。

● **教学拓展**

同学们可以试着求解“北斗七星数”。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

求了“鸡”和“玫瑰”下次说说“兔子”

使用程序求解“兔子繁殖”问题

（引出斐波那契数列）

任务 3　使用程序求解“兔子繁殖”问题

目标描述

任务描述
● **编写程序实现** 求解“兔子繁殖”问题。 即求解一年后兔子繁殖了多少对？ ● **技术层面** 掌握递推算法的含义及应用。 掌握递归算法的含义及应用。 ● **课程思政** 探索与思考。

学习活动 1　接领任务

领任务单
● 任务确认 编写 C 语言程序，求解“兔子繁殖”问题，即求解 1 年后兔子繁殖了多少对？ 具体要求如下： （1）程序最终能正确展示求解结果； （2）掌握 C 语言代码的使用规范（变量取名及注释说明）； （3）程序能正确运行，并应具有可扩展性。 ● 确认签字

学习活动 2　分析任务

编写 C 语言程序，求解 1 年后兔子繁殖了多少对的问题。

历史上有一个关于兔子繁殖的问题：

假设有一对兔子，两个月后它们就算长大成年了，

以后每个月都会生出 1 对兔子，

生下来的兔子也都是长两个月就算成年，然后每个月也都会生出 1 对兔子，

这里假设兔子不会死，每次都是只生 1 对兔子。

这样过了 1 年之后，会有多少对兔子呢？

推算一下：

第 1 个月，只有 1 对小兔子；

第 2 个月，小兔子还没长成年，还是只有 1 对兔子；

第 3 个月，兔子长成年了，同时生了 1 对小兔子，因此有两对兔子；

第 4 个月，成年兔子又生了 1 对兔子，加上自己及上个月生的小兔子，共有 3 对兔子；

第 5 个月，成年兔子又生了 1 对兔子，第 3 个月生的小兔子现在已长成年了，并且生了 1 对小兔子，加上本身两只成年兔子及上个月生的小兔子，共 5 对兔子；

……

过了 1 年之后，总共有 144 对兔子。

1、1、2、3、5、8、13、21、34、55、89、144

根据兔子繁殖引出的数列，就是大名鼎鼎的“斐波那契数列”。

斐波那契数列（Fibonacci Sequence），又称黄金分割数列，是由数学家莱昂纳多·斐波那契以兔子繁殖为例子而引入的，故又称为“兔子数列”。

它指的是这样一个数列：

1、1、2、3、5、8、13、21、34、55…

在数学上，斐波那契数列以递推的方法定义：

$F(1)=1, F(2)=1, F(n)=F(n-1)+F(n-2)\ (n \geqslant 3, n \in \mathbf{N}^*)$

现代在物理、准晶体结构、化学等领域，斐波那契数列都有应用。

知识学习：递推/递归算法

知识学习

学习笔记

1．递推算法

递推算法是设计中最常用的重要方法之一，有时也称为迭代。

虽然对求解的问题不能归纳出简单的关系式，但在其前、后项之间能够找出某种普遍适用的关系。利用这种关系，便可从已知项的值递推出未知项的值。

递推算法的方向既可以由前向后，也可以由后向前。

广义地说，凡在某一算式的基础上从已知的值推出未知的值，都可以视为递推算法。

2．递归算法

递归算法是一个非常有趣且实用的设计方法。

递推算法：从已知递推出未知项的值。

递归算法：先从未知项的值递推出已知项的值，再从已知项的值推出未知项的值。

讲一个故事，来介绍递归算法的过程：

有一个家庭，夫妇俩生养了 6 个孩子，个个活泼、调皮、可爱。

有一天，家里来了一位客人，见了这一群孩子，难免喜爱和好奇。

遂问老大：“你今年多大了？”老大脑子一转，故意说：“我不告诉你，但我比老二大 2 岁。”

客人遂问老二：“你今年多大了？”老二见老大那样回答，也调皮地说：“我也不告诉你，我只知道比老三大 2 岁。”

……客人挨个问下去，孩子们的回答都一样。

轮到最小的老六时，他诚实地回答：“3 岁啦。”

于是，客人就知道老五的年龄了，再往回就推算出了老四、老三、老二和老大的年龄了。

庆幸的是老六说出了自己的年龄，不然客人就要尴尬了。

3．递归算法举例

递归算法是构造的一种基本方法，如果一个过程直接或间接地调用其自身，则称该过程是递归算法。

如在数学中求 n 的阶乘的递归函数：

$$n!=\begin{cases}1 & n=0\\ n(n-1)! & n>0\end{cases}$$

说明：

$4!=1\times2\times3\times4$

5！=1×2×3×4×5=4！×5

使用递归算法实现求 10!的程序，其代码如下。

```
int main()
{
    //调用
    int reslut=fac(10);   //求 10!
    printf("10!=%d",reslut);
}

//实现递归的函数
int fac(int n)
{
    if(n==0)
    {
        return 1;
    }
    else
    {
        return n*fac(n-1);   //自己调用自己
    }
}
```

学习活动 3　制定方案

实现本任务方案

● 实现思路

通过对本任务的分析及相关知识学习，制定方案如下：

（1）递推算法

从已知递推出未知的过程。

本任务实现求 1 年后共生多少对小兔子。

这是一个著名的“兔子数列”，即斐波那契数列。

那么这里的 1 年，其实就是 12 个月后，也就是第 12 个斐波那契数，如图 6.8 所示。

实现思路：

F(1)=1

F(2)=1

F(n)=F(n-1)+F(n-2)　　n>=3

使用递推算法设计如下：

F1=1（已知）

F2=1（已知）

Fi=Fi-1+Fi-2（未知）　i>=3

f1=1
f2=1
f3=f1+f2=2
f4=f2+f3=3
f5=f3+f4=5
f6=f4+f5=8
f7=f5+f6=13
f8=f6+f7=21
f9=f7+f8=34
f10=f8+f9=55
f11=f9+f10=89
f12=f10+f11=144

图 6.8　递推算法思路

（2）递归算法

从未知递推已知的过程。

那么这里的 1 年，其实就是 12 个月后，也就是第 12 个斐波那契数。

实现思路：

F(1)=1

F(2)=1

F(i)=F(i-1)+F(i-2)　　i>=3

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 fib。

（2）在项目的 main.c 文件中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）递推算法代码

通过已知的第 1 个月和第 2 个月都为 1 开始，从第 3 个月开始由前两个月的和相加推到未知的第 12 个月，求得最终结果。

```
int main()
{
    int m =12;              //一年 12 个月
    int f1,f2;              //前两月
    int sum=1;              //最终结果

    f1=1;
    f2=1;
    while(m>=3)
    {
        sum=f1+f2;
        f1=f2;
        f2=sum;
        m=m-1;
    }
    printf("一年共有 %d 对兔子",sum);
    return 0;
}
```

（2）递归算法代码

定义一个 fib2 函数实现求解兔子繁殖的过程。

首先在 main 函数中，调用 fib2(12)，也就是直接从要求解的值（未知）开始；然后在 fib2() 中不断地递归调用自己，最终实现从未知开始递归到已知，程序结束，最终求得结果。

```
int main()
{
```

```
    int temp=1;
    temp=fib2(12);     //一年 12 个月
    printf("一年后 有 %d 对兔子",temp);
    return 0;

}

//递归算法实现
int fib2(int n)
{
    if(n>=3)
    {
        return fib2(n-1)+fib2(n-2);
    }
    else
    {
        return 1;
    }
}
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

利用“递推算法”“递归算法”来求解 1 年后兔子繁殖多少对的问题。

按制定方案进行任务实现，在正确的情况下，任务实现的效果如图 6.9 所示（以递归算法为例）。

```
一年后 有 144 对兔子Press any key to continue
```

图 6.9　兔子繁殖任务的运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量取名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

利用“递推算法”“递归算法”来求解 1 年后兔子繁殖多少对的问题。

● **技术层面**

“递推算法”和“递归算法”。

● **课程思政**

通过本任务的学习，同学们掌握了“斐波那契数列”求解的相关知识，以及递推算法和递归算法的含义及应用。

同时，希望同学们养成透过表面发现本质的习惯，努力把自己培养成一个有好奇心，并勤于思考的人。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

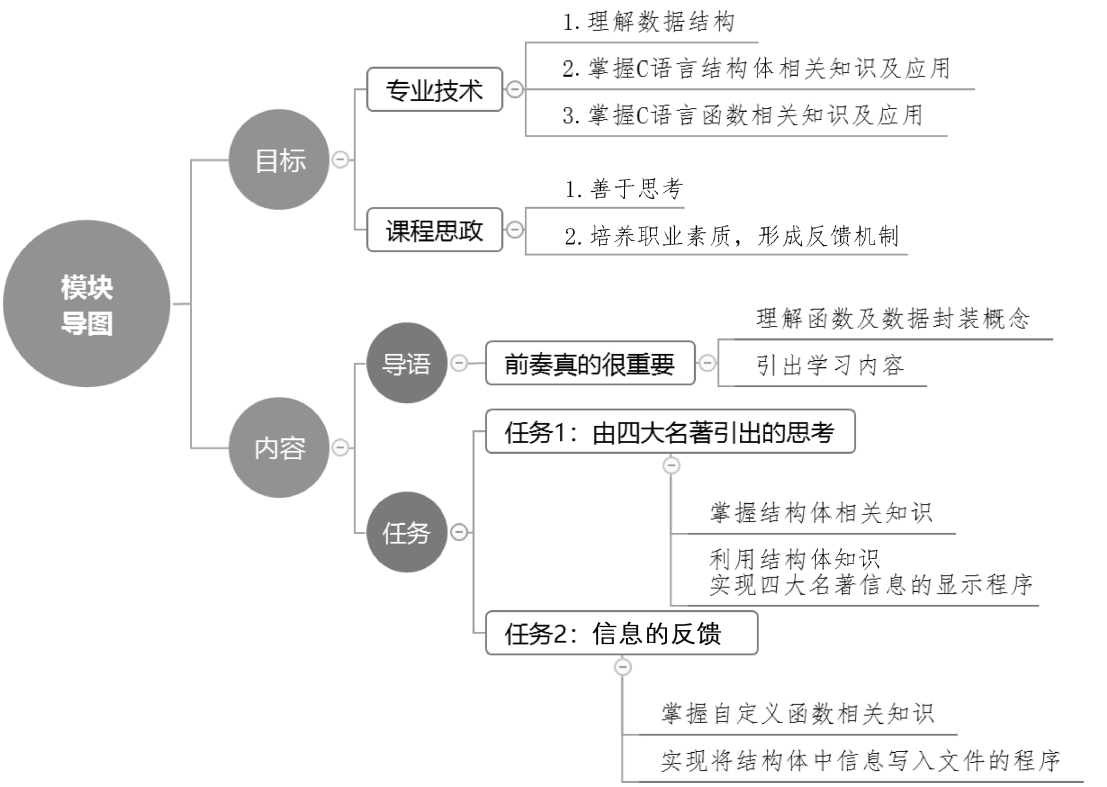
什么是函数呢

下一回：模块 7　C 语言程序中的函数及结构体应用

（开启函数和结构体）

模块 7

C 语言程序中的函数及结构体应用



项目导语：前奏真的很重要

1. 生活中的前奏

在日常生活中，我们经常会提到“前奏”的问题，如购买新房一样，先要按具体功能对其划分出相应的空间，然后做好规划与设计，待装修好就可以入住了，如图 7.1 所示。

入住新房的前奏

以功能进行规划与设计

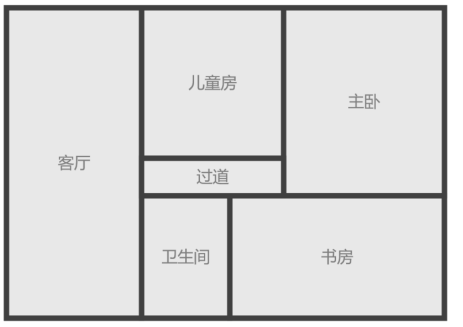


图 7.1　房屋功能的划分

新房装修好后，还要找一些收纳包，将我们的物品打好包进行搬运。到达新家后，我们就可以从包中把物品取出，实现安全可靠地搬运，如图 7.2 所示。

好高兴，搬家了

到新家后，从包中把物品拿出来

图 7.2　打包示意

2. 程序中的前奏

（1）函数

在程序中，如果将软件的所有程序代码都写在一个 main()中，大家想象一下，这样是不是就如同没有按功能进行划分的房屋。

所以，为了让程序更有层次感与可读性，在 C 语言程序中提供了函数。

函数就是把程序以功能为单位，划分成若干个功能块，方便程序之间的调用，如图 7.3 所示。以功能为单位划分为 3 个函数，可在主函数 main 中进行调用。

在后续的面向对象的程序开发语言中，函数被称为方法，所以这个前奏的学习非常重要。

（2）结构体

C 语言中的结构体实现对不同数据的表示与存储，相当于搬家过程中的打开包→装东西→封好包，如图 7.3 所示。

如定义一个 friendmodle 的结构体，它可以同时存放一个人的姓名、性别、电话、QQ 号、年龄等数据，如图 7.4 所示。

函数/方法

函数在面向对象中的新名称叫方法。

```
#include <stdio.h>

int main()
{
    // 调用函数
    openbag();
    inputdata();
    closebag();
    return 0;
}
// 打开包
void openbag()
{
    // 这里实现打开包的功能
}
// 装东西
void inputdata()
{
    // 这里实现装东西的功能
}
// 封好包
void closebag()
{
    // 这里实现关闭包的功能
}
```

图 7.3　函数功能划分

结构体/封装

结构体是面向对象中封装的前奏。

```
struct friendmodle
{
    char name[50];
    char sex[4];
    char tel[20];
    char qq[25];
     int age;
};
```

姓名

性别

电话

QQ号

年龄

图 7.4　程序结构体

来吧，正式开启程序的“前奏”

为了具体介绍结构体和函数的知识，本模块共设计以下两个任务。

任务 1：由四大名著引出的思考；

任务 2：信息的反馈。

任务 1　由四大名著引出的思考

目标描述

任务描述
● 编写程序实现 展示我国四大名著的信息，包含（本任务）：书名、作者、单价。

● 技术层面

掌握结构体的含义及应用。

● 课程思政

善于思考。

学习活动 1　接领任务

领任务单

● 任务确认

编写 C 语言程序，实现展示我国四大名著的信息。

具体要求如下：

（1）程序最终能正确展示书本的信息（书名、作者、单价）；

（2）掌握 C 语言代码的使用规范（变量取名及注释说明）；

（3）程序能正确运行，并具有可扩展性。

● 确认签字

学习活动 2　分析任务

编写 C 语言程序，实现展示我国四大名著（见图 7.5）的信息。

每本书包含如下 3 个信息（本任务），例如，

书名	作者	定价
《红楼梦》	曹雪芹	38.6 元
《三国演义》	罗贯中	54.3 元
《水浒传》	施耐庵	33.6 元
《西游记》	吴承恩	49.4 元

使一本书中包含三个不同类型的数据

分析：

（1）我国四大名著的信息。

结论：4 本书的信息=>数据集合。

（2）每本书包含的信息。

书名　　作者　　定价

↓　　↓　　↓

字符型　字符型　浮点型

如何实现这个数据的集合，并保存数据。

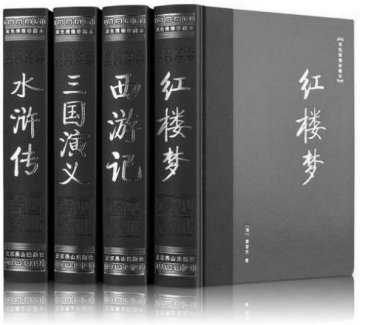


图 7.5　四大名著

利用数组能实现吗

这里使用数组实现不了。

因为，数组只能实现同一数据类型的数据集合。

那怎么办呢？

使用结构体可以实现。

知识学习：C 语言的结构体

知识学习

学习笔记

知识回顾：

整型：表示一个整数；

浮点型：表示一个小数；

字符型：表示一个字符；

数组：存储一组具有相同类型的数据集合。

遇到的问题：

在解决实际问题时，我们需要将以上不同数据类型结合起来，共同表示某一个对象的信息。

如表示一个学生的信息：学号（字符型）、姓名（字符型）、年龄（整型）等；

表示一本书籍的信息：书名（字符型）、作者（字符型）、定价（浮点型）等；

这种情况，我们就需要一个新的数据类型，它能实现使用不同类型共同表示一个对象整体信息，这就是结构体。

结构体特点：

结构体可以实现将不同数据类型构成一个整体来表示现实中某一个对象的信息，如表示一个学生的信息等。

C 语言使用结构体变量进一步加强了表示数据的能力。

```
struct friendmodle
{
    char name[50];
    char sex[4];
    char tel[20];
    char qq[25];
     int  age;
};
```

姓名

性别

电话

QQ号

年龄

1．结构体的定义

结构体只有先定义好，才可使用。

结构体定义的语法：

```
struct 结构体名
```

```
{
    成员变量;
};
```

说明：

（1）结构体使用 struct 定义；

（2）结构体名遵循变量定义规则；

（3）{}中定义可表示对象的信息。

2．结构体定义举例

```
//定义书本信息结构体
struct book
{
    char title[50];          //书名，一个字符串
    char author[50];         //作者，一个字符串
    float value;             //定价，一个浮点数
};
```

说明：

（1）定义了一个叫 book 的结构体。

（2）book 结构体包含三个成员：title、author、value，分别表示书名、作者、定价信息。

3．结构体的使用

定义好结构体后，相当于创建了一种新的数据类型，就可以像 int 之类的数据类型一样，创建结构体变量，从而实现对结构体的应用。

struct book book1;　　//定义一个 book 结构体变量 book1

定义好结构体变量后，就可以通过结构体变量访问成员了，如

book1.value=65.7;

4．结构体数组的使用

定义好结构体后，相当于创建了一种新的数据类型，也可以结构体来定义一个结构体数组。

结构体数组的定义：

```
struct  结构体名  数组名[大小];
```

说明：

（1）结构体数组的定义和传统的数组定义完全一样；

（2）只是在定义结构体数组时，必须加上 struct。

5．结构体数组举例

如以定义好的 book 结构体为例，来定义结构体数组，如

```
struct book bookList[4];
//可以存放 4 本书的结构体数组
struct book bookList[300];
//可以存放 300 本书的结构体数组
```

定义好结构体数组就可以通过数组元素来访问结构体的成员了，如

strcpy(**bookList[2].title**,"水浒传");

strcpy(**bookList[2].author**,"施耐庵");
bookList[2].value=33.6;
这里的 strcpy 是个函数。
这个函数的功能就是字符串赋值函数，也就是将“水浒传”这个字符串，赋值给 bookList[2].title 成员。

学习活动 3　制定方案

实现本任务方案

● **实现思路**

通过对本任务的分析及相关知识学习，制定方案如下：

方法一：采用结构体变量实现

（1）定义一个结构体，包含书名、作者和定价；
（2）在程序中定义结构体变量；
（3）分别给结构体变量成员赋上书本的 3 个信息值；
（4）显示输出结构体变量值，以实现本任务要求。

方法二：采用结构体数组实现

（1）定义一个结构体，包含书名、作者和定价；
（2）在程序中定义结构体数组；
（3）给结构体数组成员赋上书本的 3 个信息值；
（4）循环显示输出结构体数组元素的值，以实现本任务要求。

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 showbookinfo。
（2）分别在项目的 main()中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● **实现代码**

采用两种方式实现任务。

方法一：使用结构体变量方式实现参考代码。

```
//定义书本信息结构体
struct book
{
    char title[50];          //书名，一个字符串
    char author[50];         //作者，一个字符串
    float value;             //定价，一个浮点数
};
```

```
int main()
{
      //定义 4 个结构体变量
     struct book book1,book2,book3,book4;
     //对结构体变量赋值
     //第 1 本书的信息
     strcpy(book1.title,"红楼梦");
     strcpy(book1.author,"曹雪芹");
     book1.value=38.6;
     //第 2 本书的信息
     strcpy(book2.title,"三国演义");
     strcpy(book2.author,"罗贯中");
     book2.value=54.3;
     //第 3 本书的信息
     strcpy(book3.title,"水浒传");
     strcpy(book3.author,"施耐庵");
     book3.value=33.6;
     //第 4 本书的信息
     strcpy(book4.title,"西游记");
     strcpy(book4.author,"吴承恩");
     book4.value=49.4;
     //显示输出
     printf("中国四大名著信息列表\n");
     printf("书名:%s\t 作者:%s\t 定价:%.1f\n",book1.title, book1.author, book1.value);
     printf("书名:%s\t 作者:%s\t 定价:%.1f\n",book2.title, book2.author, book2.value);
     printf("书名:%s\t 作者:%s\t 定价:%.1f\n",book3.title, book3.author, book3.value);
     printf("书名:%s\t 作者:%s\t 定价:%.1f\n",book4.title, book4.author, book4.value);

}
```

说明：

（1）定义了一个结构体 book，包含书本的 3 个信息成员；

（2）在 main()中定义了 4 个 book 结构体变量；

（3）分别给这 4 个结构体变量赋上四大名著的信息；

（4）显示 4 个结构体变量的成员值，从而实现本任务要求。

但如果不止 4 本书，该怎么办？

可以使用结构体数组实现。

方法二：使用结构体数组方式实现参考代码。

```
//定义书本信息结构体
struct book
{
     char title[50];          //书名，一个字符串
     char author[50];         //作者，一个字符串
     float value;             //定价，一个浮点数
};

int main()
```

```
{
    //定义 1 个结构体数组
    struct book bookList[4];

    //数组赋值
    strcpy(bookList[0].title,"红楼梦");
    strcpy(bookList[0].author,"曹雪芹");
    bookList[0].value=38.6;

    strcpy(bookList[1].title,"三国演义");
    strcpy(bookList[1].author,"罗贯中");
    bookList[1].value=54.3;

    strcpy(bookList[2].title,"水浒传");
    strcpy(bookList[2].author,"施耐庵");
    bookList[2].value=33.6;

    strcpy(bookList[3].title,"西游记");
    strcpy(bookList[3].author,"吴承恩");
    bookList[3].value=49.4;

    //显示输出
    printf("中国四大名著信息列表\n");

    int i;
    for(i=0; i<4; i++)
    {
        printf("第%d 本书本信息,书名:%s\t 作者:%s\t 定价:%.1f\n",
                i+1, bookList[i].title,bookList[i].author,bookList[i].value);
    }
}
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

编写 C 语言程序，使用两种方法实现展示我国四大名著的信息，包括（本任务）书名、作者、定价。

按制定的方案进行任务实现，在正确的情况下，方法一实现的效果如图 7.6 所示。

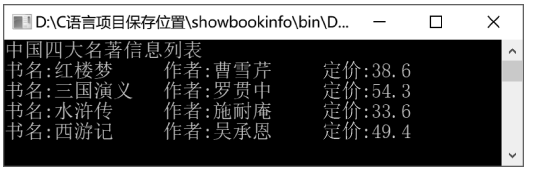


图 7.6　方法一的运行效果

方法二实现的效果如图 7.7 所示。

图 7.7　方法二的运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	开发环境安装与置换情况					
3	掌握知识的情况					
4	程序运行情况					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

编写 C 语言程序，展示我国四大名著的信息，包含（本任务）书名、作者、定价。

● **技术层面**

结构体的定义、结构体变量的应用、结构体数组的应用。

● **课程思政**

通过本任务的学习，同学们掌握了 C 语言结构体的相关知识，同时也希望同学们能有更多的思考。例如，

从《西游记》中领会到团队协作的重要性；

从《水浒传》中领会到英雄本色，在未来的职业江湖中始终有一把“交椅”是属于你的，这把“交椅”的质量如何，那就得看你的努力程度了。

从《三国演义》中领会到用人之道，想要取得成功，就要学会用人、管人、能留住人。

从《红楼梦》中学到认识自我，做一个自律、自省、自知、自强的人。

只有这样才能做到心中有数，遇到困难时披荆斩棘，做自己的主人。

● **教学拓展**

同学们掌握了结构体的应用，试着对本任务进行优化，完成书本的详细信息显示。

书本信息包括书名、作者、出版社、ISBN、定价。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

讲讲“函数”这个前奏

下一回：信息的反馈

（自定义函数）

任务 2　信息的反馈

目标描述

任务描述
● **编写程序实现** 编写程序将结构体中的数据保存到文件中，并对操作进行反馈（保存结果是成功或失败）。 ● **技术层面** 掌握函数的定义与应用。 掌握函数的返回、传参和调用。 ● **课程思政** 培养职业素质，形成反馈机制。

学习活动 1　接领任务

领任务单
● 任务确认 编写 C 语言程序，使用其自定函数的方式，实现将结构体中的数据保存到文件中，并反馈保存结果（成功或失败）。 具体要求如下： （1）程序能正确将结构体中的数据写到文件中； （2）写数据到文件的功能，单独以自定义函数实现； （3）写数据成功与否要有良好的反馈信息； （4）掌握 C 语言代码的使用规范（变量取名及注释说明）； （5）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

使用 C 语言自定函数的方式，实现将结构体中的数据保存到文件中，并反馈保存的结果。

要完成本任务，首先要了解函数及自定义函数的含义。

知识学习：C 语言的函数

知识学习

每个 C 语言程序都至少有一个函数，即主函数 main，代表程序的入口。函数有很多称呼，如方法、子程序等。

函数可分为两类。

（1）内置函数

标准库提供了大量的程序可以调用的函数，如：

strcat()用来连接两个字符串；

strcpy()用来给字符串赋值。

（2）自定义函数

用户自己编写，用于实现某个特定功能的程序。

1．函数的定义

```
return_type function_name( parameter list )
{
    body of the function
    return  数据
}
```

学习笔记

说明：

（1）return_type：代表函数运行后返回的数据类型。

有数据返回：代表函数执行完后，返回来的数据是什么类型，即调用方使用对应的数据类型来接收。

无数据返回：函数执行完后，没有数据返回，这里使用特定关键字 void 表示。

（2）function_name：代表函数的名称。

通过函数的名称可进行访问与执行。

函数必须有名称，需要遵循变量名取名的规则。

同一程序中不能有同名的函数名，不能使用 C 语言的特定关键字作为函数名。

（3）parameter list：代表函数的参数。

它是向函数内部输入数据的接口。

参数是有数据类型的，调用函数时，一定注意数据类型、参数顺序、参数数量要一一对应）。

函数可以没有参数，也就是运行该函数时，不需要向函数输入数据。

（4）body of the function：代表函数主体。

它是函数具体完成什么功能的程序代码。

（5）return 数据：代表函数返回数据。

如果这个函数不使用 void 表示，则执行完了，必须返回数据，使用 return 语句则返回数据。

如果这个函数使用 void 表示，说明不用返回数据，则函数中不需要使用 return 语句。

2．函数定义举例

```
/* 函数返回两个数中较大的那个数 */
int max(int num1, int num2)
{
    /* 局部变量声明 */
    int result;
    if (num1 > num2)
    {
        result = num1;
    }
    else
    {
        result = num2;
    }
    //返回结果
    return result;
}
```

说明：

（1）函数名：max。

（2）参数：2 个整型数据。

（3）返回结果是一个整型值。

```
/* 产生一个 100 以内的随机数 */
int   getSomeData( )
{
      int result =0;
      //获取一个随机数种子
      srand((unsigned) time(NULL));
      //用 rand()产生 100 以内随机数
      result = rand()%100;
      //返回
      return result;
}
```

说明：

（1）函数名：getSomeData。

（2）参数：无。

（3）返回结果是一个整型值。

```
/* 显示菜单 */
void showmenu( )
{
       printf("菜单选择：\n");
       printf("1. 添加数据\n");
       printf("2. 修改数据\n");
       printf("3. 删除数据\n");
       printf("0. 退出系统\n");
 }
```

说明：

（1）函数名：**showmenu**。

（2）参数：无。

（3）返回结果：无。

3．函数的调用

函数定义好后，只有调用时，函数才会被执行。

根据函数的定义进行调用，代码如下：

```
/* 函数返回两个数中较大的那个数 */
int max(int num1, int num2)
{
      /* 局部变量声明 */
      int result;
      if (num1 > num2)
      {
            result = num1;
      }
      else
      {
```

```
            result = num2;
        }
        //返回结果
        return result;
    }
    //主函数中调用自定义的函数
    int main ()
    {
        int a = 100;
        int b = 200;
        //调用函数
        int ret = max(a, b);
        printf( "较大的数是  %d\n", ret );
        return 0;
    }
```

说明：

（1）自定义函数 int max(int num1, int num2)，调用时要求传入两个整型数据，函数运行后要返回一个整型的数据。

（2）在主函数 main 中调用 int ret = max(a, b)。

先定义 a 和 b 两个变量，并赋值，然后调用 max 自定义函数，将返回结果保存到 ret 变量中。

（3）将 100 和 200 两个整数传入 max()中，并计算出最大值返回，也就是比较两个数中的最大数。

```
    /*  显示菜单  */
    void showmenu( )
    {
        printf("菜单选择：\n");
        printf("1.  添加数据\n");
        printf("2.  修改数据\n");
        printf("3.  删除数据\n");
        printf("0.  退出系统\n");
     }
    /*  产生一个 100 以内的随机数  */
    int    getSomeData( )
    {
        int result =0;
        //获取一个随机数种子
        srand((unsigned) time(NULL));
        //用 rand()产生 100 以内随机数
        result = rand()%100;
        //返回
        return result;
    }
    int main ()
    {
```

```
    //调用函数
    showmenu( );
    int num1= getSomeData( );

    printf( "随机数为: %d\n", num1);

    return 0;
}
```

说明：

（1）实现了两个自定义函数，即 showmenu 和 getSomeData。

（2）在主函数 main 中进行调用。

其中，showmenu()没有参数和返回，所以直接调用；

getSomeData()没有参数，但有返回，所以调用时用 num1 变量来接收返回的整型数据。

特别说明：

函数调用时“()”不能少。

学习活动 3　制定方案

实现本任务方案

● **实现思路**

通过对本任务的分析及相关知识学习，制定方案如下：

（1）定义一个描述书本信息的结构体；

（2）在 main()之前（上方）创建一个新的函数 saveData()。

接收参数为书本信息结构体。

函数功能：将接收参数传进来的结构体中的书本信息保存到文件中。

函数返回：返回一个整型数据（1 为成功，0 为失败）。

（3）在 main()中实现对书本结构体数据的赋值，并调用 saveData()完成本任务。

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 FunDemo。

（2）在 main.c 文件中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● **实现代码**

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 FunDemo。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()之前创建描述书本信息的结构体代码，其代码如下。

```
//定义书本信息结构体
struct book
{
    char title[50];          //书名，一个字符串
    char author[50];         //作者，一个字符串
    float value;             //定价，一个浮点数
};
```

（4）编写 saveData()。

在定义好的结构体下方创建该函数。

接收参数为书本信息结构体；

函数功能：将接收参数传进来的结构体中的书本信息保存到文件中；

函数返回：返回一个整型数据（1 为成功，0 为失败），参考代码如下：

```
/*   将结构体的数据保存到文件中   */
int   saveData(struct book Datas)
{
    int jieguo=1;
    FILE *fp=fopen("test.txt","a");
    if(!fp)
    {
        jieguo=0;
    }
    fprintf(fp,"%s %s %f \n",Datas.title, Datas.author, Datas.value);
    fclose(fp);
    // 函数返回
    return jieguo;
}
```

（5）编写 main()，实现调用。

在完成以上操作后，编写 main()中的代码，实现对自定义函数 saveData()的调用，以完成本任务。

```
//调用函数
int main()
{
    //定义 1 个结构体变量
    struct book book1 ;
    strcpy(book1.title,"红楼梦");
    strcpy(book1.author,"曹雪芹");
    book1.value=38.6;
    // 调用函数
    int   test=saveData(book1);
    //判断显示结果
    if(test==1)
    {
        printf("保存成功！");
    }
```

```
    else
    {
        printf("保存失败！");
    }
}
```

实现本任务后的完整代码如下。

```
//定义书本信息结构体
struct  book
{
    char title[50];          //书名，一个字符串
    char author[50];         //作者，一个字符串
    float value;             //定价，一个浮点数
};
/*  将结构体的数据保存到文件中  */
int  saveData(struct book Datas)
{
    int jieguo=1;
    FILE *fp=fopen("test.txt","a");
    if(!fp)
    {
        jieguo=0;
    }
    fprintf(fp,"%s %s %f \n",Datas.title, Datas.author, Datas.value);
    fclose(fp);
    //  函数返回
    return jieguo;
}

//调用函数
int main()
{
    //定义 1 个结构体变量
    struct book book1 ;
    strcpy(book1.title,"红楼梦");
    strcpy(book1.author,"曹雪芹");
    book1.value=38.6;
    //  调用函数
    int  test=saveData(book1);
    //判断显示结果
    if(test==1)
    {
        printf("保存成功！");
    }
    else
    {
        printf("保存失败！");
    }
}
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

编写 C 语言程序，先将书本信息保存到结构体变量中，然后作为参数传入写文件函数，从而实现将书本的信息保存到文件中。

按制定方案进行任务实现，在正确的情况下，实现的效果如图 7.8 所示。

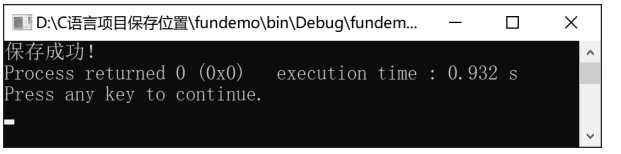


图 7.8　任务运行的效果

文件中写入数据的效果如图 7.9 所示。

图 7.9　数据写入文件的效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量取名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

将书本信息保存到结构体变量中，并作为参数传入写文件函数，从而实现将书本的信息保存到文件中。

写文件的过程是单独以一个函数实现的，该函数写完数据后，给出反馈，保存其结果（成功或失败）。

● **技术层面**

函数的定义与应用、函数的返回、传参和调用。

● **课程思政**

通过本任务的学习，同学们应掌握自定义函数的定义、返回、传参、调用等知识，并要进行加强练习。了解函数运行完有返回值的情况。

日常生活中，尤其是在未来的职场中，反馈机制将是一项非常重要的职业素质。恰当的信息反馈，在人际沟通的双向互动中发挥着十分重要的作用。

例如，领导或同事给你安排一件事务时，你是能做，还是不能做？或者完成事务的时间等，都必须及时进行反馈。如果给你安排的事务，根本完成不了，而又不反馈，那结果将会非常严重。

所以，同学们在好好理解函数的同时，也要使自己成为一个具有良好职业素质的人。

● **教学拓展**

同学们详细掌握了函数的应用，试着对之前实现的程序进行优化调整。

（1）改造“打怪”游戏的程序。

模块 4 中实现的“打怪”游戏程序，当时怪物的生命值是固定的 10000，现在编写一个自定义函数，随机产生一个 10000～100000 的数，让生命值调用这个函数，使怪物的生命值变得更加神秘。

（2）改造 BMI 程序。

之前的 BMI 程序所有代码都是在 main()中编写完成的，现在学了自定义函数后，将 BMI 区间判断单独编写一个自定义函数，将显示结论与建议也单独编写函数，并进行调用。

这样就可以让程序代码变得更加“模块化”了。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

没事，函数搞明白就行

老师，上面写文件的代码没怎么搞明白哦

下一回：模块 8　C 语言程序中的文件操作应用

（详讲文件操作）

模块 8

C 语言程序中的文件操作应用

项目导语：“日出而作，日入而息”的规律

1.“日出而作，日入而息”

本模块主要介绍 C 语言中的文件操作。它与“日出而作，日入而息”有什么关系呢？

日出而作，日入而息。
凿井而饮，耕田而食。
帝力于我何有哉！

——先秦时期《击壤歌》（见图 8.1）

图 8.1　先秦时期《击壤歌》

这首先秦时期的《击壤歌》展现了农耕时代上古先民的幸福生活场景，诠释出原始的自由安闲和自给自足的简单快乐。

国泰民安——大国自豪

安居乐业——自力更生

2. 程序中的“日出而作，日入而息”

有人可能会说，你讲文件操作跟“日出而作，日入而息”有什么关系呢？

哈哈，这就是规律。人是如此，那程序也差不到哪儿去。

接下来，我们说说程序这个主题吧。

程序在运行时，需要数据来支撑，或者程序在运行时，会产生数据，如何把数据写入文件中，以长期保存呢？如何把文件中的数据读取出来，服务于程序运行呢？

程序中“日入而息”：向文件中写入数据，保存数据

程序中“日出而作”：从文件中读取数据，支撑程序运行

开启文件操作的学习

为了详细介绍文件操作的知识，本模块共设计以下 3 个任务：

任务 1：找个地方休息一下；

任务 2：起床了都出来露个脸；

任务 3：与结构体和函数一起玩玩。

任务 1　找个地方休息一下

目标描述

任务描述
● 编写程序实现 将保存四大名著信息的结构体数据，保存到一个文本文件中。 写文件的功能，单独以函数实现。 ● 技术层面 掌握将数据写入文本文件的操作的流程。 掌握将数据写入文本文件的相关知识。 ● 课程思政 传承勤劳美德。

学习活动 1　接领任务

领任务单
● 任务确认 编写 C 语言程序，实现将结构体中的数据写入本地磁盘的文件中。 具体要求如下： （1）程序最终能正确将结构体中的信息写入文件中； （2）将数据写到文件中的功能单独定义函数实现； （3）掌握 C 语言代码的使用规范（变量命名及注释说明）； （4）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

编写 C 语言程序，将保存四大名著信息的结构体数据写入一个文本文件中，实现写文件的功能。

知识学习：C 语言写文件操作

知识学习

学习笔记

文件操作指 C 语言对磁盘文件的内容进行相应的操作（写入/读取）。

前提：

对文件进行写入/读取前，要打开文件，也就是确定操作的对象。

（1）打开文件的语法

打开文件，即确定操作的文件，打开文件的语法如下：

```
FILE   *fp = fopen("文件路径" , "操作文件类型");
```

说明：

① 文件路径。

这里可以写单独的文件名，如“1.txt”即表示当前程序运行路径中的文件，若存在则返回这个文件的指针。若不存在则返回 NULL。

也可以写一个文件的绝对路径，如“C:\aaa\ccc\a.txt”，计算机中这个路径是否存在该文件，若存在则返回文件指针，若不存在则返回 NULL。

② 操作文件类型。

操作文件类型指对文件进行什么类型的操作，具体类型如下：

类　型	说　明
r	只读。以读的方式打开文本文件，只能读数据，如果文件不存在则出错
w	覆盖。以写的方式打开文本文件，能向文件中写入数据，若文件不存在则新建。反之，则从文件起始位置写，原内容将被覆盖
a	追加。为在文件后面添加数据而打开文本文件，追加数据。若文件不存在，则新建；反之，则在原文件后追加数据
r+	可读可写。为读和写而打开文本文件，读时，从头开始；写时，新数据只覆盖所占的空间，其后不变
w+	建立一个新文件进行写操作，随后可以从头开始读。如果文件存在，则原内容全部消失

（2）关闭文件

对文件操作结束后，一定要关闭文件。

关闭文件使用:fclose()，其代码如下：

```
FILE   *fp = fopen("文件路径" , "操作文件类型");
...
...
fclose(fp);   //fp 打开文件时定义的文件对象
```

说明：

fclose(fp)：其中的 fp 是指打开文件时的变量名。关闭这个变量，意味着关闭这个变量对应的文件，从而实现关闭文件的作用。

（3）对文件进行写操作函数

实现对打开的文件进行写操作的函数如下：