

项目5 学生成绩管理系统 V2. X

学习目标

- 熟练掌握函数的定义和调用。
- 理解函数的传值调用和传地址调用。
- 理解函数的嵌套调用。
- 理解全局变量和局部变量。
- 理解变量的存储类别 auto 和 static。

学生成绩管理系统 V2. X 功能描述

学生成绩管理系统 V2. X: 采用模块化设计, 使用函数实现学生成绩的输入输出、更新、统计以及排序等功能。

5.1 学生成绩管理系统界面设计

学生成绩管理系统 V1. X 已基本实现对学生成绩的输入、输出、修改和统计等一系列操作, 但是所有的程序功能只由一个主函数(main 函数)完成, 系统结构不够清晰, 不符合“模块化”程序设计思想, 使阅读和维护程序变得困难。学生成绩管理系统 V2. X 在 V1. 0 的设计基础上, 利用函数实现学生成绩的输入、输出、修改和统计, 而且增加系统界面的设计, 使系统更加美观、规范, 功能更完善, 操作更方便。

5.1.1 函数的定义

C 语言要求, 在程序中用到的函数和变量一样, 先定义后使用。

1. 函数定义的一般形式

类型说明符 函数名 (形参类型 形参 1, 形参类型 形参 2, ..., 形参类型 形参 n)

```
{  
    声明部分  
    执行语句  
    return 语句  
}
```

函数是 C 语言程序的基本单位, 一个函数由函数首部和函数体两部分组成:

- (1) 函数首部: 由类型说明符、函数名和形参列表三部分构成。
- (2) 函数体: 是用 {} 括起来的语句序列, 主要包括声明部分和执行语句。

2. 定义函数 4 步骤

- (1) 函数名: 编程者自定义, 遵循标识符命名规则。

(2) 形参列表: 根据函数功能需求, 分析将对哪些数据进行操作, 确定形参类型和个数, 定义于圆括号中。当为多个形参时, 应分别指明形参类型, 形参之间用逗号隔开。

(3) 函数体：用一对{}括起来的若干语句,包括变量声明语句和执行语句;如果函数有返回值,通常由 return 语句带回一个返回值。

(4) 类型说明符：取决于函数返回值的类型,原则上要求与 return 后面表达式的类型一致。

例如,定义函数,实现从两个整数中找最大数的操作。

```
int max(int x,int y)                /* 函数首部 */
{
    int z;
    z=x>y?x:y;
    return z;                       /* 将 z 的最大值带回主调函数 */
}
```

说明：

(1) 函数名：max。

(2) 形参列表：该函数功能是从两个整数中找最大值,故需定义两个形参,且均为整型。故形参列表设计为“int x,int y”。

(3) 函数体：

```
{
    int z;
    z=x>y?x:y;
    return z;                       /* 将 z 的最大值带回主调函数 */
}
```

需要先定义一个整型变量 z,用于存放从 x、y 形参变量中找出的大数。

从 x、y 形参变量中找大数的操作可以使用 if…else 语句,也可以使用条件运算符。

用 return z; 语句带返回值到主调函数(即使用 max 函数)中。

(4) 函数类型：由于返回值是从 x、y 中找到的最大数 z,是整型,故函数类型也是整型 int。

5.1.2 函数的调用

1. 函数调用的一般形式

函数名(实参 1,实参 2,⋯,实参 n);

例如,为了使用已经定义好的 max 函数,实现从两个整数中找出大数的操作,在 main 函数中编写以下语句：

```
int main()
{
    int a,b;                        //用于存放两个整数
    int c;
    a=12;
    b=34;
    c=max(a,b);                    /* 函数调用 */
}
```

```
printf("max=%d\n",c);
return 0;
}
```

说明:

(1) `int a,b`;定义两个整型变量,用于存放两个整型数据。`int c`;定义整型变量 `c`,用于存放 `a,b` 中找出的大数。

(2) `a=12`;和 `b=34`;把 12 赋值给 `a` 变量,把 34 赋值给 `b` 变量。

(3) `c=max(a,b)`;是函数调用语句。`max` 是被调用函数名,圆括号内 `a,b` 是已经被赋值的实参变量(C 语言规定,实参必须有确切的值)。`c=max(a,b)`;表示调用 `max` 函数后,找出 `a,b` 中的大数,赋值给 `c` 变量。

2. 函数的调用过程

1) 形参与实参

函数定义时,圆括号中的参数称为形式参数,简称形参;而函数调用时,圆括号中的参数称为实际参数,简称实参。C 语言规定,实参与形参个数相等,类型一致。即,实参与形参个数和类型要一一对应,保持一致;并且要保证实参具有确定值。

例如,定义 `max` 函数时的参数 `x,y`,都是形参,调用 `max` 函数时使用的参数 `a,b` 是实参,`a,b` 有确切的值。

如果定义的函数没有参数,即不需要形参,称为无参函数。调用无参函数时,也不需要实参。

2) 传值调用机制

当函数调用时,流程到被调用函数中去,系统给形参变量分配临时存储空间,实参传值给形参;执行完被调函数的函数体语句,流程返回主调函数,形参的存储空间被撤销。实参向形参传值是单向的,即传值调用是单向的。

例如,当执行 `c=max(a,b)`;语句时,流程转到被调函数 `max` 中,系统给形参 `a,b` 各分配 4B 存储空间,`a` 传值给 `x`,`b` 传值给 `y`,此时,`x` 值为 12,`y` 值为 34。在 `max` 函数中通过 `z=x>y? x: y`;语句找到 `x,y` 中的大数赋值给 `z` 变量。

当被调函数 `max` 执行完毕,流程返回主调函数,`return z`;语句带回返回值 `z`,赋值给 `c`,此时 `c` 的值为 34。`max` 函数调用结束,形参 `x,y` 的存储空间被撤销。

3) 函数返回值

如果被调函数有返回值要带回主调函数,必须包含 `return` 语句,例如 `max` 函数中的 `return z`;。如果被调函数没有返回值,可以省略 `return` 语句,函数体最后的右括号“`}`”也会将流程返回主调函数。

C 语言规定,原则上函数类型要求与函数返回值类型一致,即根据函数返回值类型确定函数类型;如果函数没有返回值,函数类型设置为 `void`(空类型)。如果函数类型与函数返回值类型不一致,函数类型决定返回值类型,即返回值类型会被强制转换为函数类型。应该避免发生这种强制类型转换。

【示例 5-1】 定义自定义函数,找两个整数的最大数。

```
#include<stdio.h>
int max(int x,int y)           //函数定义,x,y 为形参,函数类型 int 与返回值 z 类型一致
```

```
{
    int z;
    z=x>y?x:y;
    return z;                                /* return 语句,带返回值 z 到主调 main 函数 */
}
int main()
{
    int a,b,c;
    a=12; b=34;
    c=max(a,b);                             //函数调用语句,函数调用结束,返回值赋值给 c 变量
    printf("max=%d\n",c);
    return 0;
}
```

运行结果：



说明：

(1) 执行过程如下：

① 从主函数 main 开始执行,声明变量 a、b、c,并给 a、b 赋初值 12、34,如图 5.1(a)所示。

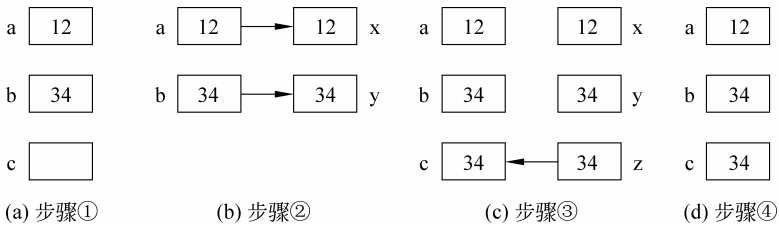


图 5.1 实参、形参数传递示意图

② 调用 max 函数。当函数调用时,流程到被调函数 max 中,声明形参变量 x、y;系统给形参 x、y 分配临时存储空间,实参 a、b 传值给对应的形参 x、y,如图 5.1(b)所示。

③ 在 max 函数中,声明临时变量 z,用于存放找出的最大值 34。当函数调用结束,流程返回函数调用处,return z;语句带回的返回值 34 赋值给 c 变量,如图 5.1(c)所示。

④ 形参的存储空间撤销,如图 5.1(d)所示。

⑤ 主函数 main 继续执行,输出最大值。

读者可以单步执行示例 5-1,查看程序流程走向和 x、y 形参,z 变量,c 变量,数值变化。

(2)程序由一个或多个函数组成,每个函数实现一个特定的功能。示例 5-1 由 main 函数和 max 函数组成,max 函数实现求最大值功能。

(3) 不管 main 函数处在程序中的什么位置,C 语言程序的执行总是从 main 函数开始,到 main 函数结束。

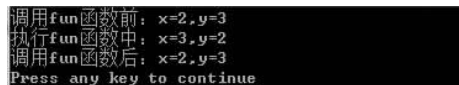
【示例 5-2】 没有返回值的自定义函数的定义与调用。

```

#include<stdio.h>
void fun(int x,int y);                                //fun 函数声明语句
int main()
{
    int x=2,y=3;
    printf("调用 fun 函数前:");
    printf("x=%d,y=%d\n",x,y);                      /* 输出 x、y 的初始值 */
    fun(x,y);                                         /* 调用 fun 函数 */
    printf("调用 fun 函数后:");
    printf("x=%d,y=%d\n",x,y);                      /* 输出调用后的 x、y 值 */
    return 0;
}
/* 定义 fun 函数,实现两个参数值的交换 */
void fun(int x,int y)                                //x、y 为形参,本函数没有返回值,为 void 类型
{
    int t;
    t=x;
    x=y;
    y=t;
    printf("执行 fun 函数中:");
    printf("x=%d,y=%d\n",x,y);                      /* 输出交换后的值 */
}

```

运行结果:



```

调用fun函数前: x=2,y=3
执行fun函数中: x=3,y=2
调用fun函数后: x=2,y=3
Press any key to continue

```

说明:

(1) fun 函数实现两个参数值的交换并输出,由于没有返回值,函数类型为 void;虽然函数体中省略了 return 语句,但是函数体中的右大括号“}”也有返回调用处的作用。

(2) 如果函数定义在下,函数调用在上,在函数调用语句之前,需要函数声明语句。因为 C 语言编译是自上而下的,函数声明语句能避免系统不识别被调函数。

函数声明语句格式:

函数类型 函数名(形参列表);

形象化理解: 函数声明语句就是函数首部+“;”。例如:

```
void fun(int x,int y);
```

(3) fun 函数中形参 x 和 y 虽然借助 t 变量进行了交换,但是 main 函数中对应的实参 x 和 y 的值并没有相应发生改变。即函数调用时,实参传值给形参,函数调用结束,形参的存储空间撤销,形参没有改变实参值。在此,再次验证了传值调用是单向的。

5.1.3 学生成绩管理系统界面设计

1. 明确功能

采用模块化设计,使用函数实现对学生成绩管理系统中操作功能选择界面的设计,使学生成绩管理系统更加规范、美观,从而提高学生成绩管理系统的可读性和可操作性。

2. 分析问题

学生成绩管理系统界面设计模块如图 5.2 所示。

输入:无输入。

任务:显示学生成绩管理系统的操作选择界面。

输出:输出学生成绩管理系统的操作选择界面。

3. 逻辑设计

此项目主要实现以上两个设计模块中操作功能选择界面的显示输出,其伪代码为

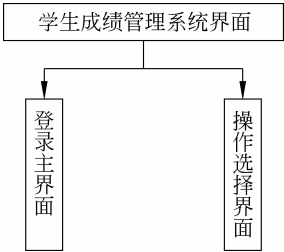
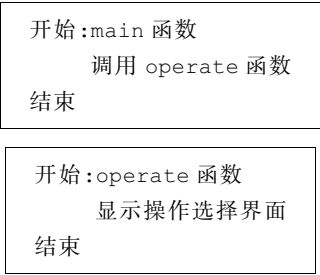


图 5.2 学生成绩管理系统界面模块



4. 编写代码

```
#include<stdio.h>

void operate(); //函数声明

int main()
{
    operate(); //调用 operate 函数,没有实参 /
    return 0;
}

/* 输出显示操作选择界面 */
void operate() //无参函数,无返回值
{
    printf("\t 欢迎进入学生成绩管理系统 V2.1\n");
    printf("===== \n");
    printf("\t\t 请你选择操作类型:\n");
    printf("\t\t 1 输入各学生成绩\n");
    printf("\t\t 2 显示各学生成绩\n");
    printf("\t\t 3 按要求修改学生成绩\n");
    printf("\t\t 4 统计总成绩、平均成绩\n");
    printf("\t\t 5 统计最高分\n");
}
```

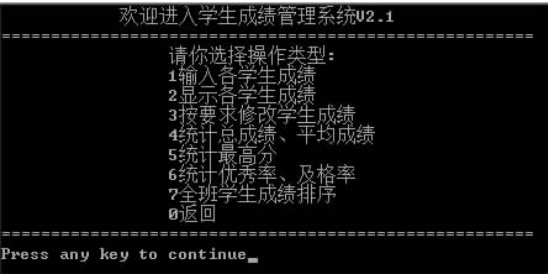
```
printf("\t\t 6 统计优秀率、及格率\n");
printf("\t\t 7 全班学生成绩排序\n");
printf("\t\t 0 返回\n");
printf("=====\n");
}
```

说明：

- (1) 程序由 main 函数和 operate 自定义函数组成,main 是主调函数,operate 是被调函数。
- (2) operate 函数输出操作选择界面,为无参函数,无返回值,故函数类型为 void。
- (3) 调用 operate 函数语句为 operate();,该函数为无参函数,调用时没有实参。
- (4) 输出函数中使用转义字符'\t',表示横向跳到下一制表位,一个制表位占 8 个字符位。'\t\t'表示跳过两个制表位,控制从 17 列开始输出操作界面。

5. 运行、调试程序

输入程序,保存文件后运行。如果出错,检查程序,修改错误,并重新运行。
运行结果：



5.1.4 实训任务

【任务 5-1】 判断给定的 3 个整数能否构成三角形。

功能描述：根据输入的 3 条边的值,判断“任意两条边的和是否大于第三条边”,得出结论。

提示：

- (1) 设计函数 triangle,根据 3 个整型参数 a、b、c 的值判断能否构成三角形,即： $a+b>c$ & $b+c>a$ & $c+a>b$,如果条件成立,返回值为 1,否则返回值为 0。
- (2) main 函数调用 triangle 函数,根据返回值为 1 或 0 显示能否构成三角形的判断结果。

【任务 5-2】 学生成绩管理系统登录主界面设计。

功能描述：使用学生成绩管理系统时,输完密码后,进入如下的登录主界面：

```
~~~~~
                        欢迎使用学生成绩管理系统 V2.1
~~~~~

                        请你选择操作类型：

=====
```

=====

提示：

(1) 设计输出以上信息的函数 inland, 为无参函数(此模块将在学生成绩管理系统 V2.1 设计中使用)。

(2) main 函数调用 inland 函数, 显示系统登录主界面。

【任务 5-3】 银行 ATM 机功能显示界面设计。

功能描述：使用 ATM 机取款时, 当输完密码后, 进入如下功能界面：

中国建设银行 ATM 机

```
-----
                        1. 查询
                        2. 取款
                        3. 存款
                        4. 退卡
                        5. 返回
-----
```

提示：

(1) 设计输出以上信息的函数 show(无参函数)。

(2) main 函数调用以上 show 函数, 显示 ATM 机功能界面。

【任务 5-4】 学生量化管理系统 V2.0 界面设计。

提示：

(1) 学生量化管理系统界面也包括系统登录主界面和操作功能选择界面, 分别定义两个函数实现。

(2) main 函数中依次调用以上定义的两个函数。

5.2 学生成绩管理系统 V2.0

5.2.1 函数的嵌套调用

C 语言的函数定义是互相平行、各自独立的, 不能嵌套定义。但函数可以嵌套调用, 也就是除了在主函数中可以调用自定义函数外, 在被调用的自定义函数中, 还可以再调用其他函数, 称为函数的嵌套调用。

【示例 5-3】 函数的嵌套调用。

```
#include<stdio.h>
int f2(int m);           //f2 函数声明语句
int f1(int n)            //f1 函数定义, n 是形参
{   int y;
    if (n>5)
    {
```

```
        y=f2(n);                //在 f1 函数中调用 f2 函数,n 是实参
    }
    else
    {
        y=2 * n;
    }
    return y;                    //返回值 y
}
int f2(int m)                   //f2 函数定义,m 是形参
{
    int x;
    x=2 * m;
    return x;                  //返回值 x
}
int main()
{
    int a;
    scanf("%d",&a);
    printf("%d\n",f1(a));      //调用 f1 函数,a 是实参,已经输入值
    return 0;                 //返回结果直接输出
}
```

运行结果：

```
8
16
Press any key to continue_
```

说明：

(1) 此程序包含 main 函数和自定义函数 f1、f2。3 个函数是彼此平行的，形象化地理解：main 函数和 f1、f2 函数三者间是“兄弟关系”，不是“父子关系”。

(2) 程序执行过程如图 5.3 所示。

5.2.2 数组元素作函数实参

数组元素实际上就是该类型的一个简单变量。例如，有一个整型数组 `int a[10];`，则其中的数组元素 `a[0]`，`a[1]`，`a[2]`， $\cdots$ ，`a[9]` 都是整型变量，因此，数组元素作函数实参时，等价于简单变量作实参，是传值调用。

【示例 5-4】 数组元素作函数实参。

```
#include<stdio.h>
void swap1(int x,int y)                /* 自定义函数 swap1,实现两个数交换 */
{
    int t;
    t=x;
```

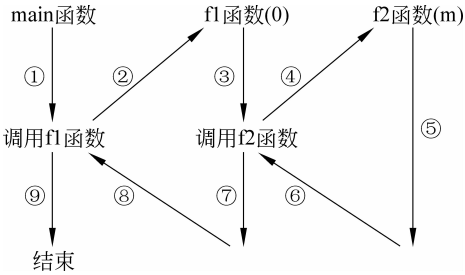


图 5.3 函数的嵌套调用

```
x=y;
y=t; /* 交换 x 和 y 的值 */
printf(" 形参值:x=%d,y=%d\n",x,y);
}
int main()
{
    int a[]={3,5,2}; /* 定义数组 a,a[1]值为 5,a[2]值为 2 */
    printf("交换之前: a[1]=%d,a[2]=%d\n",a[1],a[2]);
    /* 输出交换前的 a[1]和 a[2]的值 */
    swap1(a[1],a[2]); /* 调用交换函数 */
    printf("交换之后: a[1]=%d,a[2]=%d\n",a[1],a[2]);
    /* 输出交换后的 a[1]和 a[2]的值 */
    return 0;
}
```

运行结果:

```
交换之前: a[1]=5,a[2]=2
形参值: x=2,y=5
交换之后: a[1]=5,a[2]=2
Press any key to continue
```

说明:

- (1) 从 main 函数开始执行,先输出交换之前的 $a[1]=5$ 和 $a[2]=2$ 。
- (2) 函数调用 `swap1(a[1],a[2]);` 语句,实参 $a[1]$ 、 $a[2]$ 是数组元素,等价于简单变量;实参 $a[1]$ 、 $a[2]$ 传值给形参 x 、 y ,此时 $x=5$ 、 $y=2$,如图 5.4(a) 所示。
- (3) `swap1` 函数利用 t 将 x 和 y 的值互换, x 的值为 2、 y 的值为 5,如图 5.4(b) 所示。

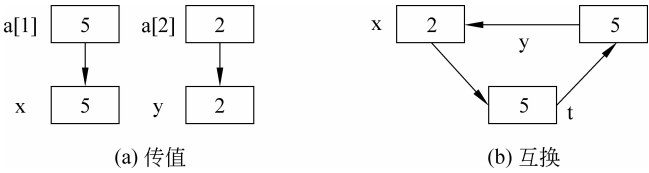


图 5.4 swap1 函数调用与执行过程示意图

- (4) 在 `swap1` 函数中输出形参值: $x=2$ 、 $y=5$ 。`swap1` 函数没有返回值,函数类型是 `void`,流程返回主调 `main` 函数。
- (5) `main` 函数继续执行,输出交换之后的 $a[1]=5$ 和 $a[2]=2$ 。
- (6) 比较以上 4 个示例可以看出,被调函数没有返回值时,可以没有 `return` 语句,函数类型是 `void`,一般在被调函数中完成输出操作;被调函数如果有返回值,使用 `return` 语句带回返回值,函数类型由 `return` 语句后的表达式类型决定,一般在主调函数输出结果。

5.2.3 数组名作函数实参

C 语言规定: 数组名代表数组的首地址。以数组名作函数实参,要求形参必须是同类型的数组。当函数调用时,将实参数组首地址传给形参数组,二者共用存储空间,这种传递称为传址调用。