
SCJJ-ME

Program C

VC++ 6.0

DING YIHANG

DING YIHANG & JIA CHENG FEN & GONG YANG

2019-2020

目录

实验一：上机操作初步(2 学时).....	2
实验二：简单的 C 程序设计(2 学时).....	6
实验三：选择结构程序设计(2 学时).....	8
实验四：循环结构程序设计(2 学时).....	14
实验五：函数(2 学时).....	19
实验六：数组(2 学时).....	22
实验七：数组与排序(2 学时).....	26
综合测试(2 学时).....	29

实验一：上机操作初步(2 学时)

一、实验方式：上机

二、实验目的：

- 1、熟悉 VC++语言的上机环境及上机操作过程。
- 2、了解如何编辑、编译、连接和运行一个 C 程序。
- 3、初步了解 C 程序的特点。

三、实验内容及步骤：

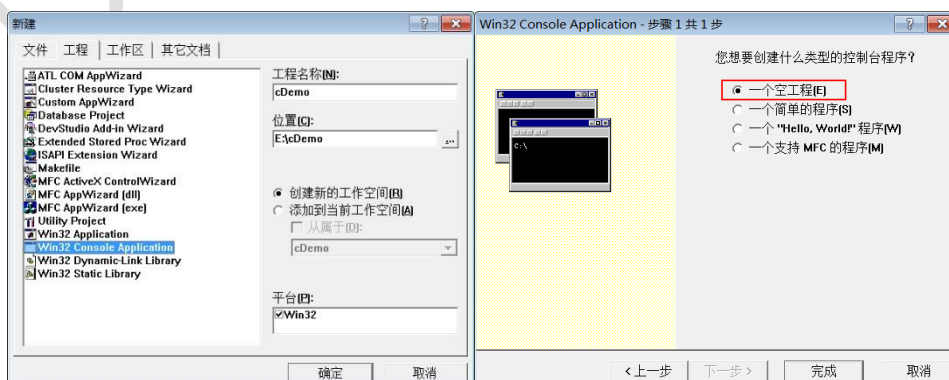
1、软件介绍：VC6.0

微软原版的 VC6.0 已经不容易找到，网上提供的都是经过第三方修改的版本，删除了一些使用不到的功能，增强了兼容性。这里我们使用 VC6.0 完整绿色版，它能够支持一般的 C/C++ 应用程序开发以及计算机二级考试。

在 VC6.0 下运行 C 语言程序，C-Free 支持单个源文件的编译和链接，但是在 VC6.0 下，必须先创建工程（Project），然后再添加源文件。一个真正的软件，往往需要多个源文件和多种资源，例如图片、视频、控件等，通常是把它们放到一个文件夹下，进行有效的管理。你可以把工程理解为这样的一个文件夹，IDE 通过工程来管理这些文件。工程有不同的类型，例如开发“黑窗口”的控制台程序，需要创建 Win32 Console Application 工程；开发带界面的 GUI 程序，需要创建 Win32 Application 工程。

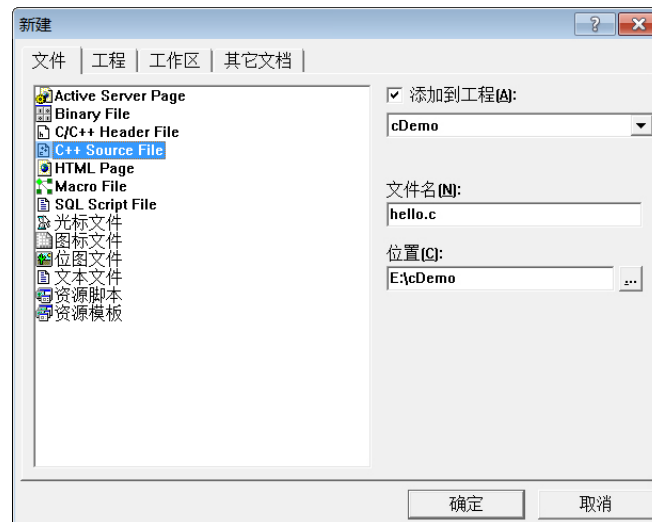
1) 新建 Win32 Console Application 工程

打开 VC6.0，在菜单栏中选择“文件 -> 新建”，或者 Ctrl+N，弹出下面的对话框，切换到“工程”选项卡，选择“Win32 Console Application”，填写工程名称和路径，点击“确定”，会弹出一个对话框询问类型，这里选择“一个空工程”。



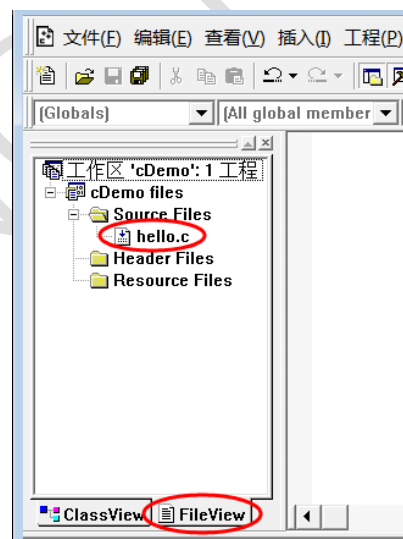
2) 新建 C 源文件

再次新建，在菜单栏中选择“文件 -> 新建”，或者 Ctrl+N，弹出下面的对话框，切换到“文件”选项卡，选择“C++ Source File”，填写文件名“xxx.c”，点击确定完成。该步骤是向刚才创建的工程添加源文件。



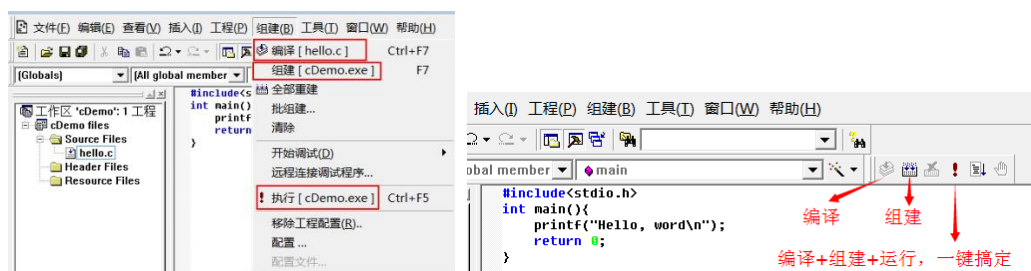
3) 编写 C 语言代码


在工作空间中可以看到刚才创建的工程和源文件，如下图所示，双击 xxx.c，进入编辑界面，就可以书写代码了。



4) 编译并运行代码

你可以在“组建”菜单中找到编译、组建和运行的功能（左图），或者使用快捷键（右图）。



保存编写好的源代码，点击运行按钮  或 Ctrl+F5，如果程序正确，可以看到运行结果，如下图所示：

```
C语言程序设计实验课！
Press any key to continue
```

注意：编译生成的 .exe 文件在工程目录下的 Debug 文件夹内。以上面的工程为例，路径为 E:\cDemo，打开看到有一个 Debug 文件夹，进入可以看到 cDemo.exe。

5) 工程文件说明

进入工程目录，会看到很多其他文件 .dsp .dsw .opt .plg，它们是 VC6.0 创建的，用来支持当前工程，不属于 C 语言的范围。你需要记住这几种：源程序文件扩展名：.c，目标文件扩展名：.obj，可执行文件扩展名：.exe，工程文件：.dsp。

源程序“.c”：源程序文本。源程序不能直接在计算机上执行。目标程序“.obj”：源程序经过“编译程序”编译所得到的二进制代码称为目标程序。可执行程序“.exe”：可在操作系统下独立执行的程序称为可执行程序。打开之前保存的文件是打开.dsp 格式的文件。

一个工程文件实际上可以放多个源代码，但是这是属于 C 语言后期针对大型项目才这样做。我们前期几十行的代码，请确保一个工程文件只放一个源代码!! 如果不想每次重新开工程文件，小技巧就是使用/* */注释掉之前的代码即可。

2、题目练习

1、让我们写一个更加华丽的 hello world!，输出如下信息：

```
*****
```

```
Hello World!
```

```
*****
```

```
#include <stdio.h>
int main(void)           //主函数的标准写法
{
    printf("*****\nHello  World! \n*****\n");
    system("pause");      // 如果删除，在debug文件中运行.exe程序会一闪而过。
    return 0;
}
```

2、从键盘输入两个整数，计算并输出两个整数的和与积。

```
#include <stdio.h>
int main(void)
{
    int a, b;
    scanf("%d%d", &a, &b);
    printf("和=%d,积=%d\n",a+b, a*b);
    return 0;
}
```

// 注：如果是除法 a/b ，注意先要进行分母 b 是否为 0 用 if 语句进行讨论，然后进行注意类型的强制转换 $(float)a/b$

3、从键盘输入一个角度的弧度值 x ，计算该角度的余弦值，将计算结果输出到屏幕。

```
#include <stdio.h>
#include <math.h>
int main(void)
{
    double x, a;
    printf("请输入一个角度的弧度值:");
    scanf("%lf",&a);
    x=cos(a);
    printf("余弦值=%lf",x);
    return 0;
}
```

4、从键盘上输入两个整数，交换这两个整数。

```
#include <stdio.h>
int main(void)
{
    int a,b,c;
    printf ("请输入两个数:");
    scanf ("%d%d", &a, &b);
    printf ("起始数:%d %d",a,b);
    c=a,a=b,b=c;
    printf ("交换后:%d %d\n",a,b);
    return 0;
}
```

实验二：简单的 C 程序设计(2 学时)

一、实验方式：上机

二、实验目的：

- 1、掌握 C 语言的数据类型。
- 2、学会使用 C 语言的运算符及表达式。
- 3、掌握不同数据类型的输入输出方法。

三、实验内容及其步骤：

- 1、输入 r1、r2，求出圆形垫片面积。（要求自行编写程序并上机运行，并且在程序中引入宏常量）
- 2、从键盘输入一个 3 位整数，并分离三位的个位、十位和百位数字。（要求输入正确的程序并且上机查看运行结果）

```
#include <stdio.h>
int main(void)
{
    int x, a, b, c;
    printf("请输入一个三位整数: ");
    scanf("%d", &x);
    a = x/100;
    b = x%100/10;
    c = x%10;
    printf("百位是%d, 十位是%d, 个位是%d\n", a, b, c);
    return 0;
}
```

3、熟悉强制类型转换和自增，输入正确的程序并且上机查看运行结果。

/*3.1 熟悉强制类型转换*/

```
#include <stdio.h>
int main(void)
{
    float a = 3.1415926;
    printf("a = %f\n", a);
    printf("a = %d\n", (int)a);
    return 0;
}
```

/*3.2 熟悉自增自减运算符*/

```
#include <stdio.h>
int main(void)
{
    int m, n, i = 8, j = 10;
    m = ++i;
    n = j++;
    printf("%d, %d, %d, %d", m, n, i, j);
    return 0;
}
```


实验三：选择结构程序设计(2 学时)

一、实验方式：上机

二、实验目的：

- 1、熟练掌握 if 语句和 switch 语句。
- 2、练习熟悉条件运算表达式 ? :。

三、实验内容及其步骤：

- 1、读入 3 个分别表示箱子长、宽、高的整数值，判断并输出该箱子是立方体还是长方体。

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

```
int main(void)
```

```
{
```

```
    int a, b, c;
```

```
    printf("请依次输入箱子的长、宽、高：");
```

```
    scanf("%d%d%d", &a, &b, &c);
```

```
    if (a == b && b == c)
```

```
    {
```

```
        printf("该箱子为立方体！");
```

```
    }
```

```
    else
```

```
    {
```

```
        printf("该箱子为长方体！");
```

```
    }
```

```
    return 0;
```

```
}
```

- 2、有一函数， $y = \begin{cases} x & (x < 1) \\ 2x - 1 & (1 \leq x < 10) \\ 3x - 11 & (x \geq 10) \end{cases}$ ，编写程序，输入实数 x 值，输出 y 值（结果保留小数点后 2 位数字）。

注意：这个题牵扯到实数判断相等的问题，要自己定义一个非常小的值。

```

#include<stdio.h>
#include<math.h>
#define EPS 1E-6
int main(void)
{

    float x, y;
    printf("请输入x的值: ");
    scanf("%f", &x);
    if (x < 1)
    {
        y = x;
        printf("函数 y=%.2f", y);
    }
    else if (x > 1 || fabs(x - 1.0) <= EPS && x < 10) //实数比较相等的时候一定要留心!
    {
        y = 2 * x - 1;
        printf("函数 y=%.2f", y);
    }
    else
    {
        y = 3 * x - 11;
        printf("函数 y=%.2f", y);
    }

    return 0;
}

```

3、输入月份，屏幕输出相应的季节。春季 3、4、5 月份，夏季 6、7、8 月份，秋季 9、10、11 月份，冬季 12、1、2 月份。

```

#include<stdio.h>
int main(void)
{
    int month;
    printf("请输入月份: ");
    scanf("%d", &month);
    switch (month)
    {
        case 3:
        case 4:
        case 5:
            printf("此时是春季! ");
            break;
        case 6:
        case 7:

```

```

case 8:
    printf("此时是夏季! ");
    break;
case 9:
case 10:
case 11:
    printf("此时是秋季! ");
    break;
case 12:
case 1:
case 2:
    printf("此时是冬季! ");
    printf("哈哈! ");
    //case后可以任意数量的语句，是少数几个不需要花括号的地方
    break;
default:
    printf("请输入正确的月份数! ");
    //default 不是必须的，当没有default时，所有 case 都匹配失败，就什么都不执行。
    //default放到最后不需要break;，但如果在前面必须break
}

return 0;
}

```

注意：当输入一个数据为5.9时候，最终输出为春季！这是为什么？这是scanf()的问题，P90。由于函数scanf()不进行参数类型匹配检查。因此当参数地址表中的变量类型与格式字符不一致时，只是导致数据不能正确读入，但编译器不提示任何出错信息，即当用户输入错误的数据，例如5.9对于scanf("%d")的%d来说，小数点是个非法字符，scanf()就认为输入数据结束，导致后面的9没有录入，实际上switch接受的只是小数点前面的5而已，符合switch的数据要求。

4、输入三个各不相等整数 x, y, z, 请把这三个数由小到大输出。实例中为了节省纸张，使用了条件表达式的三重嵌套，请先练习熟悉下例条件表达式，然后改用 if 语句重新改写程序。

```

#include<stdio.h>
int main(void)
{
    int x, y, z, min, mid, max;
    printf("请输入三个各不相同的整数: ");
    scanf("%d%d%d", &x, &y, &z);
    max = (x > y) ? ((x > z) ? x : z) : ((y > z) ? y : z);
    min = (x < y) ? ((x < z) ? x : z) : ((y < z) ? y : z);
    mid = x + y + z - max - min;
    printf("这三个数从小到大依次为: %d, %d, %d", min, mid, max);
    return 0;
}

```

注意：平时写程序不建议用三重条件表达式嵌套，不便于阅读。

/*第四题 if 语句改写，分享几个同学们写的不错的想法*/

//方法1，实际上这属于人工排序，不建议用，显得很low的。

```
#include<stdio.h>
int main(void)
{
    int a, b, c;
    printf("输入各不相同的三个整数: ");
    scanf("%d%d%d", &a, &b, &c);
    if (a > b && b > c)
    {
        printf("从小到大依次为: %d, %d, %d", c, b, a);
    }
    if (c > b && b > a)
    {
        printf("从小到大依次为: %d, %d, %d", a, b, c);
    }
    if (c > a && a > b)
    {
        printf("从小到大依次为: %d, %d, %d", b, a, c);
    }
    if (b > a && a > c)
    {
        printf("从小到大依次为: %d, %d, %d", c, a, b);
    }
    if (a > c && c > b)
    {
        printf("从小到大依次为: %d, %d, %d", b, c, a);
    }
    if (b > c && c > a)
    {
        printf("从小到大依次为: %d, %d, %d", a, c, b);
    }
    return 0;
}
```

//方法2，声明6个整型变量，x, y, z, min, mid, max，然后用if的各种嵌套完全改写模板，就不在此列出了，同样也是代码比较多。

//方法3，只声明了4个整型变量，注意这个题比较的顺序是，a依次跟b、c比较，将最小的数放到最前面，最后b在跟c比较，这种结构不光针对于三个不同的整数，任意的三个整数都可以。这种排序方法具体查看数组后面章节内容。交换排序法，选择排序的基础。

```
#include <stdio.h>
int main(void)
{
    int a, b, c, t;    //定义4个基本整型变量a、b、c、t
    printf("Please input a, b, c: \n");
    scanf("%d%d%d", &a, &b, &c);
    if (a > b)    /*如果a大于b,借助中间变量t实现a与b值的互换*/
    {
        t = a;
        a = b;
        b = t;
    }
    if (a > c)    /*如果a大于c,借助中间变量t实现a与c值的互换*/
    {
        t = a;
        a = c;
        c = t;
    }
    if (b > c)    /*如果b大于c,借助中间变量t实现b与c值的互换*/
    {
        t = b;
        b = c;
        c = t;
    }
    printf("The order of the number is:\n");
    printf("%d,%d,%d", a, b, c);    /*输出函数顺序输出a、b、c的值*/
    return 0;
}
```

//方法 4，只声明了 4 个整型变量，注意这个题比较的顺序是，a 跟 b 比较，b 跟 c 比较，将最大的数放到最后面，最后 a 在跟 b 比较，调整前面的顺序，这种结构不光针对于三个不同的整数，任意的三个整数都可以。冒泡排序。

```
#include<stdio.h>
int main(void)
{
    int a, b, c, t;
    printf("Input three number: \n");
    scanf("%d%d%d", &a, &b, &c);
    if (a > b)
    {
        t = a;
        a = b;
        b = t;
    }
    if (b > c)
    {
        t = b;
        b = c;
        c = t;
    }
    if (a > b)
    {
        t = a;
        a = b;
        b = t;
    }

    printf("%d, %d, %d\n", a, b, c);
    return 0;
}
```

注意：后期学了数组，直接用数组实现排序，具体看实验七。

实验四：循环结构程序设计(2 学时)

一、实验方式：上机

二、实验目的：

- 1、熟悉 continue 和 break 语句。
- 2、掌握循环语句及其嵌套。
- 3、掌握标志变量的用法。
- 4、掌握循环结构的实现的常用算法—穷举。

三、实验内容及其步骤：

- 1、读入 5 个正整数并且输出他们，当输入的数据为负数的时候，程序终止。

```
#include<stdio.h>
int main(void)
{
    int i, n;
    for (i = 1; i <= 5; i++)
    {
        printf("请输入第%d 个数: ", i);
        scanf("%d", &n);
        if (n < 0)
        {
            break; //将 break 换成 continue 看看有什么效果
        }
        printf("n=%d\n", n);
        printf("\n"); //再一次换行
    }
    printf("程序运行结束! \n");
    return 0;
}
```

2、我国古代数学家张丘建在《算经》一书中曾提出过著名的“百钱买百鸡”问题，该问题叙述如下：鸡翁一，值钱五；鸡母一，值钱三；鸡雏三，值钱一；百钱买百鸡，则翁、母、雏各几何。思路：公鸡一个五块钱，母鸡一个三块钱，小鸡三个一块钱，现在要用一百块钱买一百只鸡，问公鸡、母鸡、小鸡各多少只？可将该问题抽象成方程式组。设公鸡 x 只，母鸡 y 只，小鸡 z 只，得到以下方程式组：

$$A: 5x+3y+1/3z = 100$$

$$B: x+y+z = 100$$

$$C: 0 \leq x \leq 100$$

$$D: 0 \leq y \leq 100$$

$$E: 0 \leq z \leq 100$$

/*暴力穷举法*/

#include<stdio.h>

int main(void)

{

int i, j, k;

printf("百元买百鸡的问题所有可能的解如下: \n");

for (i = 0; i <= 100; i++)

{

for (j = 0; j <= 100; j++)

{

for (k = 0; k <= 100; k++)

{

if (5 * i + 3 * j + k / 3 == 100 && k % 3 == 0 && i + j + k == 100)

{

printf("公鸡%2d 只, 母鸡%2d 只, 小鸡%2d 只\n", i, j, k);

}

}

}

}

return 0;

}

3、根据题目 2 的解法，求解 $3x+5y+7z=100$ 的所有非负整数解，编写程序。

```
#include <stdio.h>
int main(void)
{
    int x, y, z, sum;
    for (x = 0; x <= 33; x++)
    {
        for (y = 0; y <= 20; y++)
        {
            for (z = 0; z <= 14; z++)
            {
                sum = 3 * x + 5 * y + 7 * z;
                if (sum == 100)
                {
                    printf("解为: x=%-2d y=%-2d z=%-2d\n", x, y, z);
                    printf("\n");
                }
            }
        }
    }
    return 0;
}
```

4、计算并输出给定整数 m 以内所有的素数。

思路 1): 判断一个整数 m 是否是素数，只需把 m 被 $2 \sim m-1$ 之间的每一个整数去除，如果都不能被整除，那么 m 就是一个素数。

思路 2): 另外判断方法还可以简化。 m 不必被 $2 \sim m-1$ 之间的每一个整数去除，只需被 $2 \sim \sqrt{m}$ 之间的每一个整数去除就可以了。如果 m 不能被 $2 \sim \sqrt{m}$ 间任一整数整除， m 必定是素数。例如判别 17 是否是素数，只需使 17 被 $2 \sim 4$ 之间的每一个整数去除，由于都不能整除，可以判定 17 是素数。原因：因为如果 m 能被 $2 \sim m-1$ 之间任一整数整除，其二因子必定有一个小于或等于 \sqrt{m} ，另一个大于或等于 \sqrt{m} 。例如 16 能被 2、4、8 整除， $16=2*8$ ，2 小于 4，8 大于 4， $16=4*4$ ， $4=\sqrt{16}$ ，因此只需判定在 $2 \sim 4$ 之间有无因子即可。

```

#include<stdio.h>
#include<math.h>
#include<stdlib.h> //调用了 exit(0);函数

int main(void)
{
    int m, i, j;
    int flag = 1; //定义一个标志变量，如不理解看书 P85、P131
    printf("请输入一个正整数: ");
    scanf("%d", &m);
    if (m == 0 || m == 1)
    {
        printf("%d 不是素数", m);
        exit(0); //调用此函数，需要将 stdlib.h 标准库函数包含进来
    }
    //C11 标准可以写成 for (int i = 2; i <= m; i++), 而上机软件 C89 标准不能这样写
    for (i = 2; i <= m; i++)
    {
        for (j = 2; j <= sqrt(i); j++)
        {
            if (i % j == 0)
            {
                flag = 0;
                break;
                //如果能整除，就跳出内层循环，不打印，进入外层循环重新测试其他数
            }
            else
            {
                flag = 1;
                //如果内层循环结束后都不能整除，那么进入外层循环的下一步，打印。
            }
        }
        if (flag == 1) //或者简化一点写成 if(flag)
        {
            printf("%d 是素数\n", i);
        }
    }

    return 0;
}

```

5、用 do-while 玩猜数游戏 1-100，先由计算机想一个数请用户猜，如果用户猜大了，计算机给出提示“big”，如果用户猜小了，计算机给出提示“small”，直到用户猜对了，计算机给出提示“right”并结束程序。

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

int main(void)
{
    int magic, guess, i = 0;
    srand(time(NULL));
    magic = rand() % 100 + 1;
    do {
        printf("请猜一个 1-100 的整数: ");
        scanf("%d", &guess);
        i++;           //对猜的次数计数
        if (guess > magic)
        {
            printf("big!\n");
        }
        else if (guess < magic)
        {
            printf("small!\n");
        }
        else
        {
            printf("right!\n");
        }
    } while (guess != magic);

    printf("你猜了%d 次，终于猜对了! ", i);

    return 0;
}
```

实验五：函数(2 学时)

一、实验方式：上机

二、实验目的：

- 1、掌握函数的定义、函数声明、函数调用和函数嵌套。
- 2、掌握通过参数在函数间传递数据的方法。
- 3、掌握全局变量和局部变量。
- 4、掌握常用算法—递归。

三、实验内容及其步骤：

- 1、编写程序实现 $1 \sim n$ 的阶乘然后每项阶乘进行求和，用两个函数分别实现阶乘和求和的效果。

```
#include<stdio.h>
long Factorial(int n);           //函数声明
long Sum(int n);                //函数声明
int main(void)
{
    int num;
    printf("从键盘输入 n 的值: ");
    scanf("%d", &num);
    printf("1!+2!+...%d! = %ld\n", num, Sum(num));
    // 调用求和函数 Sum()
    return 0;
}
long Sum(int n)                  //定义函数 Sum，实现累加求和
{
    int i;
    long result = 0;
    for (i = 1; i <= n; i++)
    {
        result += Factorial(i); //嵌套调用求阶乘函数 Factorial()
    }
    return result;
}
long Factorial(int n)            //定义函数 Factorial(), 实现求阶乘
{
    int i;
    long result = 1;
    for (i = 1; i <= n; i++)
    {
```

```

        result *= i;
    }
    return result;
}
//注意:

```

2、编写程序实现对 n 求阶乘，要求定义阶乘的函数，并且使用递归来实现。

```

#include<stdio.h>
long Factorial(int n)
{
    if (n == 0 || n == 1)                //递归的基线条件
    {
        return 1;
    }
    else
    {
        return Factorial(n - 1) * n;
    }
}
int main(void)
{
    int a;
    printf("Input a number: ");
    scanf("%d", &a);
    printf("Factorial (%d) = %ld\n", a, Factorial(a)); //调用自定义函数 Factorial()。
    return 0;
}

```

3、有 5 个人坐在一起，问第 5 个人多少岁，他说比第 4 个人大 2 岁。问第 4 个人多少岁，他说比第 3 个人大 2 岁。问第 3 个人多少岁，他说比第 2 个人大 2 岁。问第 2 个人多少岁，他说比第 1 个人大 2 岁。最后问第 1 个人，他说他是 10 岁

分析：该问题是一个递归问题。要求第 5 个人的年龄，必须先知道第 4 个人的年龄，显然第 4 个人的年龄也是未知的，但可以由第 3 个人的年龄推算出来。而想知道第 3 个人的年龄又必须先知道第 2 个人的年龄，第 2 个人的年龄则取决于第 1 个人的年龄。又已知每个人的年龄都比其前一个人的年龄大 2，因此根据题意，可得到如下几个表达式：

```

age(5)=age(4)+2; age(4)=age(3)+2
age(3)=age(2)+2; age(2)=age(1)+2
age(1)=10

```

自行编写程序，自定义一个年龄的函数，当输入第几个人时求出其对应的年龄，并在主函数中打印输出。

```

#include<stdio.h>                //递归求年龄
int age(int n)
{
    int x;

```

```

    if (n == 1)
        x = 10;
    else
        x = age(n - 1) + 2;
    return x;
}
int main()
{
    int n;
    printf("请输入 n 值: ");
    scanf("%d", &n);
    printf("第%d 个人的年龄为%d\n", n, age(n));
    return 0;
}

```

4、分析程序的运行结果，为什么是这个样子，了解全局变量和局部变量。

```

#include<stdio.h>
int n = 10;                //全局变量
void Func1(void)
{
    int n = 20;            //局部变量
    printf("Func1 n: %d\n", n);
}
void Func2(int n)
{
    printf("Func2 n: %d\n", n);
}
void Func3(void)
{
    printf("Func3 n: %d\n", n);
}
int main(void)
{
    int n = 30;            //局部变量
    Func1();
    Func2(n);
    Func3();
    {
        //代码块
        int n = 40;        //局部变量
        printf("Block n: %d\n", n);
    }
    printf("main n: %d\n", n);
    return 0;
}

```

实验六：数组(2 学时)

一、实验方式：上机

二、实验目的：

- 1、掌握一维数组的定义、赋值、输入和输出的方法。
- 2、掌握一维数组与 for 循环搭配。
- 3、掌握一维数组与函数搭配。
- 4、掌握一维数组与宏常量搭配。
- 5、熟悉查找数组元素—遍历（线性查找）。

三、实验内容及其步骤：

- 1、要求用户录入 5 个同学的成绩，然后反向顺序输出这些成绩。输入 88 91 95 90 60 100，反向输出 100 60 90 95 91 88。

```
#include<stdio.h>
int main(void)
{
    int num[5], i;
    printf("输入 5 个同学的成绩：");
    for (i = 0; i < 5; i++)
    {
        scanf("%d", &num[i]);
    }
    printf("逆序输出这 5 个同学的成绩：");
    for (i = 4; i >= 0; i--)
    {
        printf("%d ", num[i]);
    }
    return 0;
}
```

- 2、在实际开发中，经常需要查询数组中的元素。例如，学校为每位同学分配了一个唯一的学号，现在有一个数组，保存了考研班所有同学的学号信息，有辅导员想知道他分管的学生是否进入了考研班，只需提供要查学生的学号即可，如果辅导员提供的学号和数组中某个学号一致，就说明该生在考研班，否则就不在。不幸的是，C 语言标准库没有提供与数组查询相关的函数，所以我们只能自己写代码。请编程实现这种功能，为简单起见，定义该考研班学生人数有 10 人，每个人的学号 8 位格式 19010101，19 表示年级，01 表示学院，接下依次是班级 01，个人编号 01。该数组是无序数组，用循环遍历数组中每个元素，即，把要查询的值挨个比较一遍。

```
#include<stdio.h>
int main(void)
{
    int id[10] = {
        19030102, 19040201, 19010201,
        19040103, 19010102, 19020102,
        19010101, 19060120, 19031031,
        19020131 };
    int i, num, flag = 1;
    //注意：此处又用到了标志变量，如果不用标志变量，能不能实现同样的效果？试一下
    printf("\n*****待查询 8 位学号*****\n");
    scanf("%d", &num);
    printf("\n*****查询后的结果是*****\n\n");
    for (i = 0; i < 10; i++)
    {
        if (id[i] == num)
        {
            printf("学号%d 在实验班，在第%d 个数组元素！ \n", num, i+1);
            flag = 0;
            break;
        }
    }
    if (flag)
    {
        printf("学号%d 不在实验班！ \n", num);
    }
    return 0;
}
//注意：这个题本质跟打印素数很像。也可以用此法来查找最大值最小值
```


3、编写程序从键盘输入 19 级某个班同学的 C 语言成绩（整数），然后输出他们的平均值（保留 2 位小数）。要求 1：综合运用之前的函数等知识；要求 2：引入宏常量，每个班同学不超过 10 人。

```
#include<stdio.h>
#include<stdlib.h>
#define N 10
//最大的数组长度为 10，如果同学数超出 10，需要改 N 为更大的长度。
void EnterScore(int score[], int n);
void ReadScore(int score[], int n);
float Average(int score[], int n);

int main(void)
{
    int n, score[N];                //内存中留出长度为 N 的内存（4N 个字节）
    printf("请输入 19 级某班的人数为：");
    scanf("%d", &n);
    if (n > N)                      //检查 n 是否超出 N 的范围
    {
        printf("Error, Please check array[N]\n!");
        exit(0);
    }
    printf("\n***** 请开始录入成绩*****\n");
    EnterScore(score, n);
    printf("\n***** 成绩依次排列是*****\n");
    ReadScore(score, n);
    printf("\n***** 最终的平均成绩*****\n");
    printf("%.2f\n", Average(score, n));
    return 0;
}

void EnterScore(int score[], int n)    //录入成绩
{
    int i;
    for (i = 0; i < n; i++)
    {
        printf("请输入第%d 个同学的成绩：", i + 1);
        scanf("%d", &score[i]);
    }
}

void ReadScore(int score[], int n)    //读取成绩
{
    int i;
```

```

    for (i = 0; i < n; i++)
    {
        printf("%d  ", score[i]);           // %d 后有俩空格，为了输出好看
    }
    printf("\n");
}

float Average(int score[], int n)           // 计算平均值（小数）
{
    int i, sum = 0;
    if (n <= 0)                             // 除 0 错误检查
    {
        printf("Error\n");
        exit(0);
    }
    for (i = 0; i < n; i++)
    {
        sum += score[i];
    }
    return (float)sum / n;                  // 强制类型转换
}

```

//注意：上课强调过定义一维数组时候方括号内只能放**常量** `int num[N]`；而实际上这只是针对 **C89** 的标准。在 **C99** 标准中，我们定义一维数组 `int num[i]`是可以在方括号内放**变量**的，这部分知识属于 C99 的变长数组。这样做的好处是不必让程序员来指定长度，因为数组长度过长，导致浪费内存，例如我们定义的 `N=10; int num[N]`；意味着我们划出一块内存给这个数组使用，内存总共 $4 \text{ (字节)} \times 10 \text{ (数组长度)} = 40 \text{ 字节}$ ，如果输入学生的数量为 6，实际上是浪费了 $4 \times 4 = 16 \text{ 个字节}$ 的内存；如果数组长度过短，肯定会导致程序程序出错。在本例中，可以尝试将 `n` 输入 11，超出了我们一开始定义数组内存的大小，运行下会发现程序出错。

实验七：数组与排序(2 学时)

一、实验方式：上机

二、实验目的：

- 1、熟悉经典的排序方法—冒泡排序。
- 2、熟悉选择排序的基础—交换排序。

三、实验内容及其步骤：

1、在实际开发中，有很多场景需要我们将数组元素按照从大到小（或者从小到大）的顺序排列，这样在查阅数据时会更加直观。例如，一个保存了商品单价的数组，排序后更容易看出它们的性价比。对数组元素进行排序的方法有很多种，比如冒泡排序、归并排序、选择排序、插入排序、快速排序等，其中最经典最需要掌握的是「冒泡排序」。整个排序过程就好像气泡不断从水里冒出来，最大的先出来，次大的第二出来，最小的最后出来，所以将这种排序方式称为冒泡排序。

冒泡排序的整体思想是这样的：从数组头部开始，不断比较相邻的两个元素的大小，让较大的元素逐渐往后移动（交换两个元素的值），直到数组的末尾。经过第一轮的比较，就可以找到最大的元素，并将它移动到最后一个位置。第一轮结束后，继续第二轮。仍然从数组头部开始比较，让较大的元素逐渐往后移动，直到数组的倒数第二个元素为止。经过第二轮的比较，就可以找到次大的元素，并将它放到倒数第二个位置。以此类推，进行 $n-1$ （ n 为数组长度）轮排序，每一轮比较的次数比上一轮 -1 次，即，数组长度 $n-1-i$ 当前轮数 i 。

编程实现数组的冒泡排序 `int num[4] = {4, 5, 2, 1}; n=4`

第一轮($i=0$)，比较两个数字 (4, 5)，不交换位置；比较 (5, 2)，交换位置 (2, 5)；比较 (5, 1)，交换位置 (1, 5)；此时，最大元素 5 被移动到了最后一位，但其他元素还是乱的。此时，`int num[4] = {4, 2, 1, 5};` 比较进行了 3 次 ($j=n-1-i$)。

第二轮($i=1$)，再次从头比较。比较 (4, 2)，交换 (2, 4)；比较 (4, 1)，交换 (1, 4)；4 和 5 就不用比较了，因为第一轮已经确立了 5 最大的地位。此时，`int num[4] = {2, 1, 4, 5};` 比较 2 次 ($j=n-1-i$)。

第三轮($i=2$)，再次从头比较。比较 (2, 1)，交换 (1, 2)；2 和 4 就不用比较了，因为第二轮已经确立了 4 的次大地位。此时，`int num[4] = {1, 2, 4, 5};` 比较 1 次 ($j=n-1-i$)。

前三轮确定了 3 个数的位置，因为这里一共有四个数，所以剩下的一个数不需再比较，就是最小的。所以 n 个数，只需排 $n-1$ 轮即可。

编程时候注意，每一轮确定一个正确位置的数，每一轮还存在一个比较次数的问题。而数组的下标是从 0 开始。

```

#include<stdio.h>
int main(void)
{
    int num[4] = { 4, 5, 2, 1 };
    int i, j, temp;
    // n-1 轮排序
    for (i = 0; i < 4-1; i++)
    {
        // 每一轮比较前 n-1-i 个，已经排好的最后 i 个不用比较。
        for (j = 0; j < 4 - 1 - i; j++)
        {
            if (num[j] > num[j+1])
            {
                temp = num[j];
                num[j] = num[j+1];
                num[j+1] = temp;
            }
        }
    }
    //输出排序后的数组
    for (j = 0; j < 4; j++)
    {
        printf("%d ", num[j]); //留有空格
    }
    return 0;
}

```

2、交换排序是选择排序的基础，或者说选择排序是交换排序的升级版，本次上机只熟悉交换排序，详细实施过程见课本 P202。

编程实现对上题中数组元素 `int num[4] = {4, 5, 2, 1}` 的进行交换法从小到大排序。

```

#include<stdio.h>
int main(void)
{
    int num[4] = { 4, 5, 2, 1 };
    int i, j, temp;
    // n-1 轮排序
    for (i = 0; i < 4 - 1; i++)
    {
        for (j = i+1; j < 4 ; j++)
        {
            if (num[i] > num[j])

```

```
        {
            temp = num[i];
            num[i] = num[j];
            num[j] = temp;
        }
    }
}
//输出排序后的数组
for (i = 0; i < 4; i++)
{
    printf("%d ", num[i]);    //留有空格
}
return 0;
}
```

3、自行编写程序，如何分别改写上述两种方法实现 `int num[4] = {4, 5, 2, 1}` 从大到小的排序。
提示：方法有很多。

比如：将 if 括号里面的大于号改成小于号，这个是最简单的方法。

综合测试(2 学时)

一、实验方式：上机

二、实验目的：

1、综合运用 C 语言的基础知识解决一些简单的问题。

三、实验内容及其步骤：

1、上了一上午的课，终于到了午饭时间，大家讨论中午吃什么，最后舍长大人一拍桌子，说：叫外卖！于是开始打电话叫外卖，规则如下：每单少于 20 元（不含）不予以送餐，每单在 20（含）到 30（不含）元之间收取 4 块钱送餐费，每单在 30（含）到 40（不含）元之间收取 2 元送餐费，每单在 40（含）元以上免费送餐，请你计算这顿午餐的全部费用。

编程思路：

- 1) 分阶段收取配送费，选择使用 `switch..case` 或 `if..else..` 语句进行判断；
- 2) 输入菜品个数不定，使用数组的概念，加入 `for` 循环遍历数组读取输入的值。

变量规定：

- 1) 菜品的数量用 `int n`;
- 2) 所有菜品的价格为 `float sum`;
- 3) 所需要付的总价格(包含配送费)为 `float totalsum`;

效果演示：

```
请输入预定餐品数量:3
请输入每个餐品的价格:21.5 11.2 23.5
你此次外卖费(免运费):56.20
```

```
请输入预定餐品数量:2
请输入每个餐品的价格:11 8.9
你此次外卖费小于20元,不予配送,感谢你的支持!
```

2、某个公司采用公用电话传递数据，数据是四位的整数，在传递过程中是加密的，加密规则如下：每位数字都加上 5，然后对 10 取余，来代替该数字；再将第一位和第四位交换，第二位和第三位交换。示例：

```
请输入四位数字:1234
加密后的数字:9876
```

四、实验报告：

班级	姓名	学号	成绩

参考答案 1: (if 写法)

```
#include <stdio.h>
#include <math.h>
#define N 10 //划一块内存出来, 假设外卖的数量不超过10
#define EPS 1E-6 //定义极小值, 用来实数比较相等
int main()
{
    int n, i;
    float sum=0, totalsum, num[N];
    printf("请输入预定餐品数量:");
    scanf("%d", &n);
    printf("请输入每个餐品的价格:");
    for (i = 0; i < n; i++)
    {
        scanf("%f", &num[i]);
        sum += num[i];
    }
    if (fabs(sum-20.0) <= EPS || sum>20 && sum < 30)
    {
        totalsum = sum + 4;
        printf("你此次外卖费(包含运费4元):%.2f\n", totalsum);
    }
    else if (fabs(sum - 30.0) <= EPS || sum >30 && sum < 40)
    {
        totalsum = sum + 2;
        printf("你此次外卖费(包含运费2元):%.2f\n", totalsum);
    }
    else if (fabs(sum - 40.0) <= EPS || sum >40)
    {
        totalsum = sum + 0;
        printf("你此次外卖费(免运费):%.2f\n", totalsum);
    }
    else {
        printf("你此次外卖费小于20元,不予配送,感谢你的支持!");
    }
    return 0;
}
```

参考答案 1: (switch 写法)

有很少同学使用以下这种 switch 和 强制类型转换, 在此展示表扬下。

```
#include <stdio.h>
#define N 10
int main(void)
{
    int n, i;
    float sum = 0, totalsum, num[N];
    printf("请输入预定餐品数量:");
    scanf("%d", &n);
    printf("请输入每个餐品的价格:");
    for (i = 0; i < n; i++)
    {
        scanf("%f", &num[i]);
        sum += num[i];
    }
    switch ((int)sum / 10)           //舍掉小数, 强制转换整数
    {
        case 0:
        case 1:
            printf("你此次外卖费小于20元,不予配送,感谢你的支持!");
            break;
        case 2:
            totalsum = sum + 4;
            printf("你此次外卖费(包含运费4元):%.2f\n", totalsum);
            break;
        case 3:
            totalsum = sum + 2;
            printf("你此次外卖费(包含运费2元):%.2f\n", totalsum);
            break;
        case 4:
            totalsum = sum;
            printf("你此次外卖费(免运费):%.2f\n", totalsum);
            break;
        default:
            totalsum = sum;
            printf("你此次外卖费(免运费):%.2f\n", totalsum);
    }
    return 0;
}
```


参考答案 2:

```
#include <stdio.h>
int main(void)
{
    int a, i, num[4];
    printf("请输入四位数字: ");
    scanf("%d", &a);
    num[0] = a % 10;           //个位
    num[1] = a % 100 / 10;     //十位
    num[2] = a % 1000 / 100;   //百位
    num[3] = a / 1000;         //千位
    for (i = 0; i < 4; i++)
    {
        num[i] += 5;
        num[i] %= 10;
    }
    for (i = 0; i <= 1; i++)    //交换
    {
        int temp;
        temp = num[i];
        num[i] = num[3 - i];
        num[3 - i] = temp;
    }
    printf("加密后的数字: ");
    for (i = 3; i >= 0; i--)
    {
        printf("%d", num[i]);
    }
    printf("\n");
    return 0;
}
```