**计算导论与程序设计**

**实验指导书**

**山东大学**

**计算机科学与技术学院**

**计算机系统教研室**

**计算导论与程序设计教研组**

**2022年9月修订**

# 目 录

[第1章 基础模块 1](#_Toc84326596)

[1.1 实验1 编程环境及标准数学函数 1](#_Toc84326597)

[1.1.1 实验目的 1](#_Toc84326598)

[1.1.2 实验任务 1](#_Toc84326599)

[1.1.3 实验要求 1](#_Toc84326600)

[1.1.4 安装Dev-Cpp（Dev-C++）集成开发环境 2](#_Toc84326601)

[1.1.5 选择32/64位编译环境 7](#_Toc84326602)

[1.1.6 在Dev-Cpp中进行C程序开发 8](#_Toc84326603)

[1.1.7 打开（重新装入）已保存的文件 13](#_Toc84326604)

[1.1.8 编程计算下述题目，并输出结果 15](#_Toc84326605)

[1.2 实验2 标准输入输出及数据的机内表示 16](#_Toc84326606)

[1.2.1 实验目的 16](#_Toc84326607)

[1.2.2 实验任务与要求 16](#_Toc84326608)

[1.3 实验3 循环、分支、枚举 19](#_Toc84326609)

[1.3.1 实验目的 19](#_Toc84326610)

[1.3.2 实验任务与要求 19](#_Toc84326611)

[1.4 实验4 进制之间的转换 20](#_Toc84326612)

[1.4.1 实验目的 20](#_Toc84326613)

[1.4.2 实验任务与要求 20](#_Toc84326614)

[1.5 实验5 数组、枚举法求解0/1背包问题算法 20](#_Toc84326615)

[1.5.1 实验目的 20](#_Toc84326616)

[1.5.2 实验任务与要求 21](#_Toc84326617)

[1.6 实验6 字符数组与字符串 23](#_Toc84326618)

[1.6.1 实验目的 23](#_Toc84326619)

[1.6.2 实验任务与要求 23](#_Toc84326620)

[第2章 提高模块 24](#_Toc84326621)

[2.1 实验7 递推 24](#_Toc84326622)

[2.1.1 实验目的 24](#_Toc84326623)

[2.1.2 实验任务与要求 24](#_Toc84326624)

[2.2 实验8 编译预处理 25](#_Toc84326625)

[2.2.1 实验目的 25](#_Toc84326626)

[2.2.2 实验任务与要求 25](#_Toc84326627)

[2.3 实验9 排序、查找 27](#_Toc84326628)

[2.3.1 实验目的 27](#_Toc84326629)

[2.3.2 实验任务与要求 27](#_Toc84326630)

[2.4 实验10 随机数与蒙特卡罗算法 27](#_Toc84326631)

[2.4.1 实验目的 27](#_Toc84326632)

[2.4.2 实验任务与要求 27](#_Toc84326633)

[2.5 实验11 函数与变量的作用域 30](#_Toc84326634)

[2.5.1 实验目的 30](#_Toc84326635)

[2.5.2 实验任务与要求 30](#_Toc84326636)

[2.6 实验12 递归算法与函数的递归调用 31](#_Toc84326637)

[2.6.1 实验目的 31](#_Toc84326638)

[2.6.2 实验任务与要求 31](#_Toc84326639)

[2.7 实验13 结构与链表、贪心法求解0/1背包问题 31](#_Toc84326640)

[2.7.1 实验目的 31](#_Toc84326641)

[2.7.2 实验任务与要求 32](#_Toc84326642)

[2.8 实验14 文件与命令行参数 33](#_Toc84326643)

[2.8.1 实验目的 33](#_Toc84326644)

[2.8.2 实验任务与要求 33](#_Toc84326645)

[第3章 拓展模块 34](#_Toc84326646)

[3.1 实验15 学生成绩管理系统 34](#_Toc84326647)

[3.1.1 实验目的 34](#_Toc84326648)

[3.1.2 实验任务与要求 34](#_Toc84326649)

[3.2 实验16 五子棋（Bonus，选做） 35](#_Toc84326650)

[3.2.1 实验目的 35](#_Toc84326651)

[3.2.2 实验任务与要求 35](#_Toc84326652)

[附录1 实验报告格式 36](#_Toc84326653)

[附录2 实验报告与源代码文件命名与提交 37](#_Toc84326654)

[附录3 在Dev-Cpp中IDE中使用工程（项目） 38](#_Toc84326655)

[附录4 练习题 47](#_Toc84326656)

# 第1章 基础模块

基础模块实验的主要目的是让学生熟悉一种编程环境，熟练掌握C语言的基础知识，并灵活利用这些知识解决一些简单的实际问题。（**注：所有实验题目都要求用C语言编程，不能用C++或其它语言实现**）。

## 1.1 实验1 编程环境及标准数学函数

### 1.1.1 实验目的

熟练掌握一种C语言集成开发环境IDE（如Dev-cpp）的使用方法，以及在该环境下对C/C++应用程序的编程、运行及调试过程；

熟悉C程序的基本结构；

熟悉C语言常用的数学函数的使用；

### 1.1.2 实验任务

在一种C语言集成开发环境IDE（如Dev-cpp）中，编写C语言编程，并运行、调试你所编写的C程序。

在编写C程序时，使用C提供的标准数学函数，计算给定的题目，并输出结果。

### 1.1.3 实验要求

1、下载安装Dev-Cpp（Dev-C++）集成开发环境（IDE）；或使用其他C语言开发环境；

2、编程熟悉C程序的基本框架，以及C语言提供的标准数学函数，并按一定的格式输出程序的执行结果；

3、编程计算下述题目，并输出结果。

(1)

(2)

(3)

(4)

(5)

(6)

(7)

(8) y= 3/4+103除以3的余数

(9) 对于3.8及-3.8，计算不大于它们的整数

(10) 对于3.2及-3.2，计算不小于它们的整数

(11) y=3.2除以-3.0的余数，y= -3.2除以-3.0的余数

5、按附录中提供的实验报告格式与要去，撰写并提交实验报告。

实验报告格式见附录1，要求见附录2。

### 1.1.4 安装Dev-Cpp（Dev-C++）集成开发环境

**1、下载Dev-Cpp（如果已下载，可跳过该步骤）**

从<https://sourceforge.net/projects/dev-cpp/>或<https://bloodshed-dev-c.en.softonic.com/>，或其它网站下载Dec-Cpp安装程序，如5.11版的Dev-Cpp 5.11 TDM-GCC 4.9.2 Setup.exe；

**2、安装Dec-Cpp**

**Step 1：**鼠标左键双击安装程序Dev-Cpp 5.11 TDM-GCC 4.9.2 Setup.exe，要求你选择安装过程提示信息所使用的语言，界面如图1-1所示。

可以使用默认语言English，也可以选择中文，然后左键点击“OK”按钮；

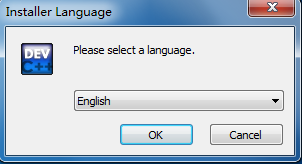


图1-1 安装语言选择界面

**Step 2：**在如图1-2所示的界面中，可以阅读了解软件的许可证，然后点击“I Agree”按钮；

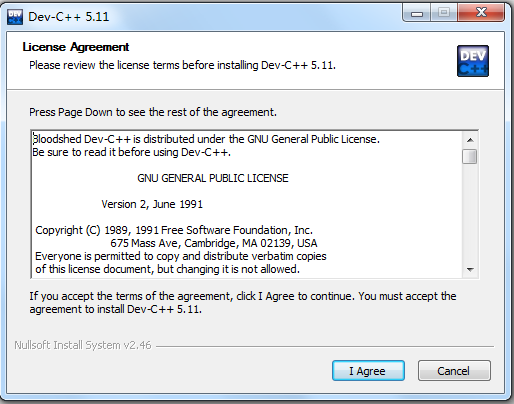


图1-2 License Agreement

**Step 3**：在如图1-3所示的界面中选择要安装的组件，安装默认的组件即可，直接点击“Next”按钮即可；

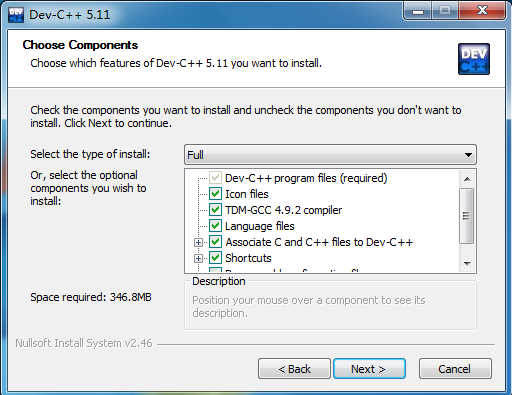


图1-3 Choose Component

**Step 4**：在如图1-4所示的界面中，选择Dev的安装位置；

安装程序默认的是将Dev安装在C:盘，建议安装到C盘以外的分区，如D:盘中自己设定的某一个目录（文件夹）中；

例如，可将文本框中的”C:\Program Files (x86)\Dev-Cpp”修改为”D:\Program Files (x86)\Dev-Cpp”，然后点击“Install”按钮，系统会开始安装；

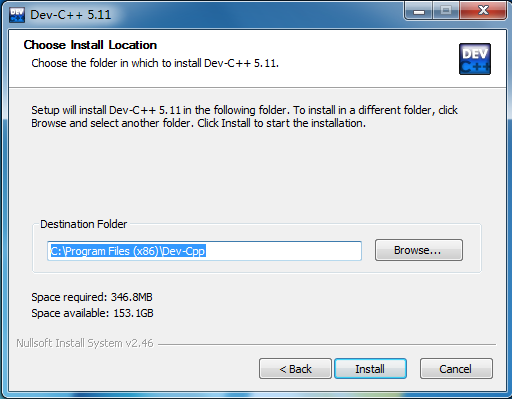


图1-4 选择Dev的安装位置

Step 5：安装完毕界面如图1-5所示。

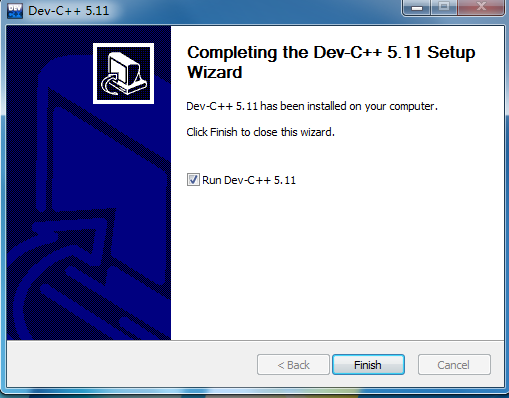


图1-5 安装结束

**3、运行Dec-Cpp**

在图1-5中点击“Finish”可以启动运行Dev-Cpp，由于这是Dev-Cpp的第一次启动运行，需要设置IDE菜单及提示信息所使用的语言，如图1-6所示；

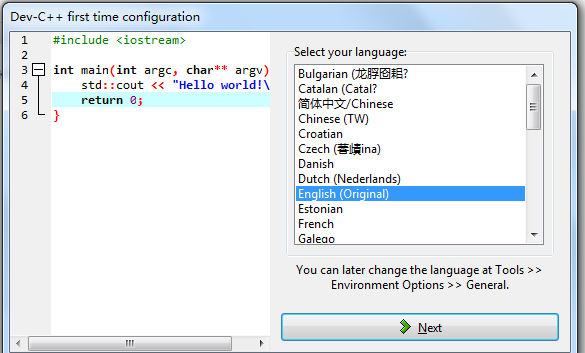


图1-6 选择IDE提示信息所使用的语言

如果使用默认的语言“English”，其IDE如图1-7所示，以后在编程、调试等过程中的提示信息也为英语。

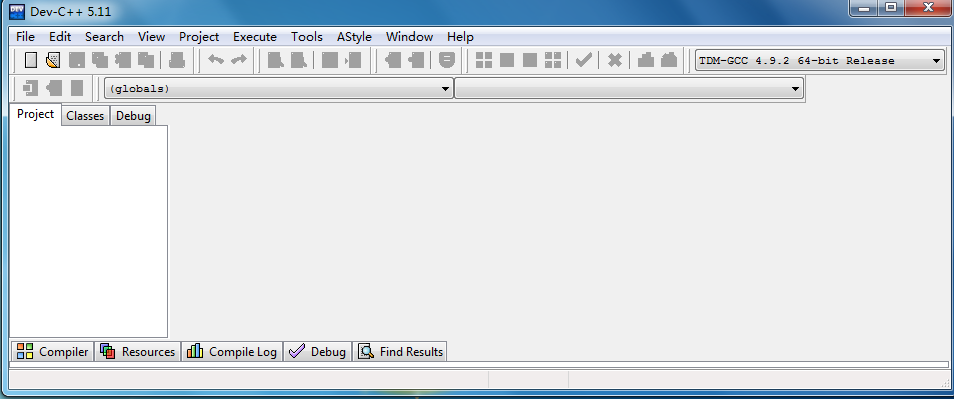


图1-7 英语界面的IDE

如果在如图1-8所示的界面中选择“简体中文/Chinese”，其IDE如图1-9所示。以后IDE的提示信息皆为中文。

你可以选择自己喜好的语言。

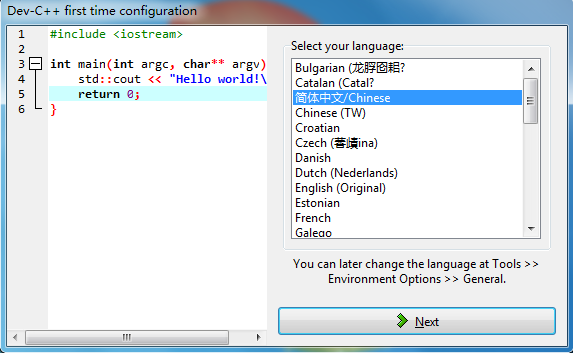


图1-8 选择简体中文作为IDE的提示信息

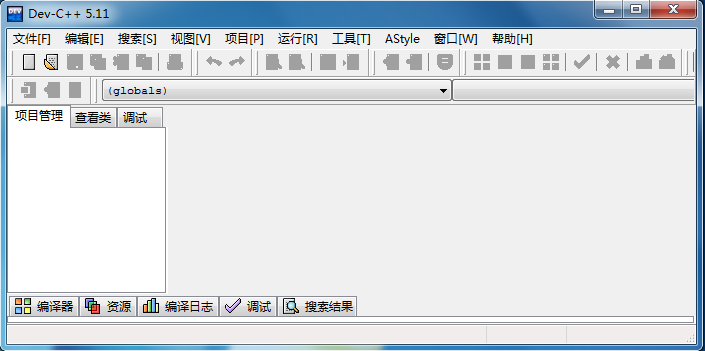


图1-9 中文界面的IDE

以后也可以在“工具(Tools)/环境选项(Environment Options)”中对上述设置的语言进行更改。

以后需要运行Dev-Cpp，可以左键双击桌面的Dev-C++图标启动之，如图1-10所示。



图1-10 Dev-C++图标

### 1.1.5 选择32/64位编译环境

可以根据需要选择32位还是64位编译环境。如图1-11所示。

实验时选择32/64编译环境均可，若希望变量的表示范围较大些，可选择64位编译环境。

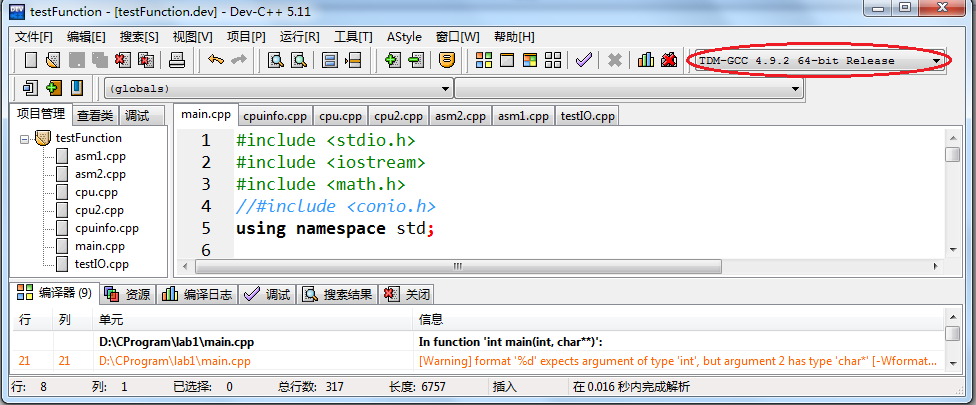


图1-11 32/64为编译环境

### 1.1.6 在Dev-Cpp中进行C程序开发

1、鼠标左键双桌面上的Dev-C++图标（如图1-10所示），打开Dev-Cpp集成开发环境，如图1-12所示。

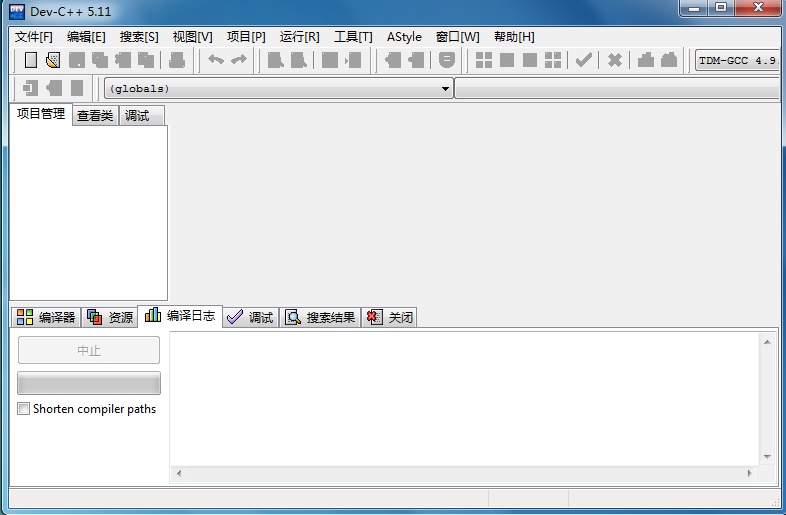


图1-12 Dev-Cpp IDE调试界面

2、创建C程序源代码文件

在编写一个C程序之前，需要建立一个源代码文件来保存要编写的C程序。

可以通过Dev-Cpp IDE菜单、图标，或者快捷键三种途径创建一个源代码文件。

（1）在Dev-Cpp IDE界面中，选择“文件/新建/源代码”，或利用快捷键ctrl-N，新建一个C程序源文件。如图1-13所示。

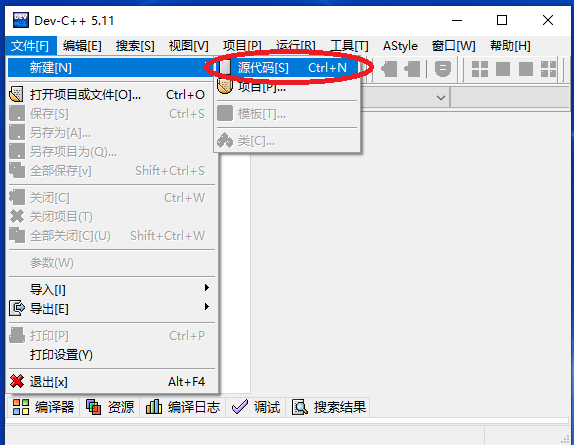


图1-13 利用菜单或快捷键创建源代码文件

（2）在Dev-Cpp IDE界面中，鼠标左键点击“新建”图标，随后点击菜单“源代码”，或利用快捷键ctrl-N，新建一个C程序源文件。如图1-14所示。

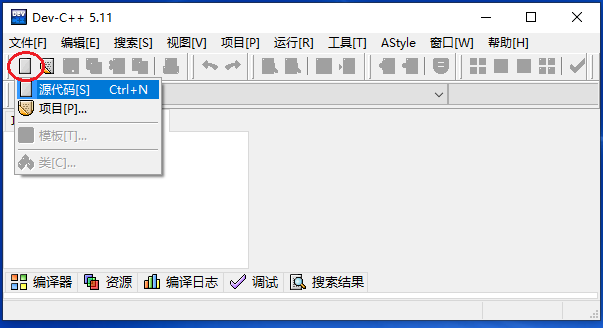


图1-14 利用“新建”图标创建源代码文件

在上面的两种方法中，点击菜单“源代码”或按下快捷键ctrl-N，新建了一个文件名为“未命名1”的C程序源文件，对应的界面如图1-15所示，在里面可以输入你要编写的源程序。

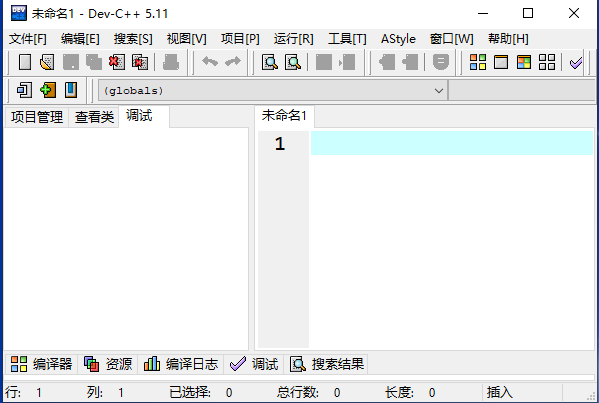


图1-15 源代码输入/调试界面

3、输入编辑源程序

例如我们要在Dev-Cpp中计算，并输出计算结果，程序大致如图1-16所示。

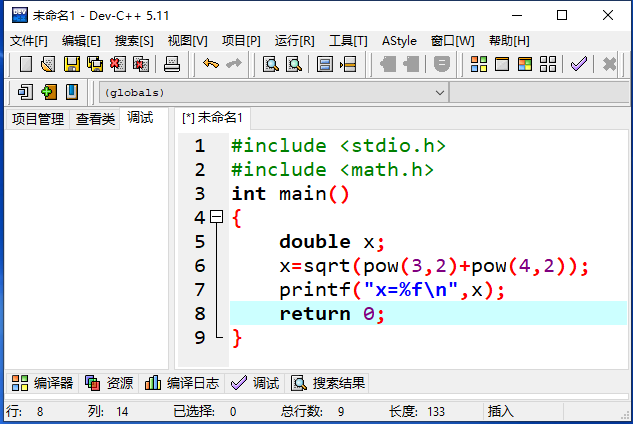


图1-16 输入编辑C源程序

4、保存源代码程序

程序输入结束后，需要将文件保存到磁盘或SSD（固态硬盘）中，其后可以编译执行之。

保存文件的两种途径：

（1）选择菜单“文件/保存”，如图1-17所示。

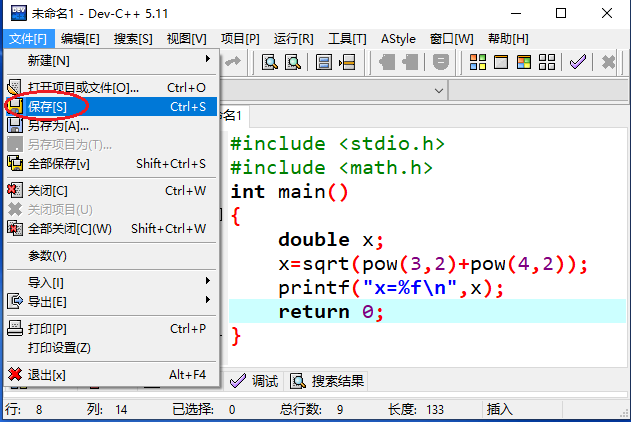


图1-17 利用菜单保存源文件

（2）鼠标左键点击保存图标，如图1-18所示。

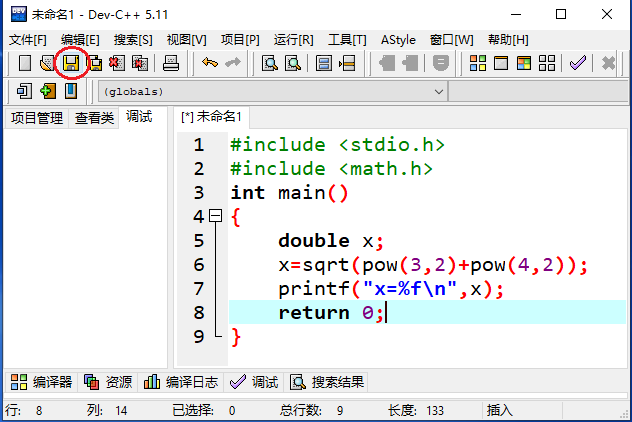


图1-18 通过点击“保存”图标保存源文件

在其后出现的如图1-19所示的对换框中，在红色椭圆标注的两处位置，自己选择或指定源文件要保存到的目录（文件夹），以及文件名，然后点击右下方绿色方框所示的“保存”按钮。

这里假设文件命名为testFunction.cpp，保存到文件夹D:\DevCpp中。

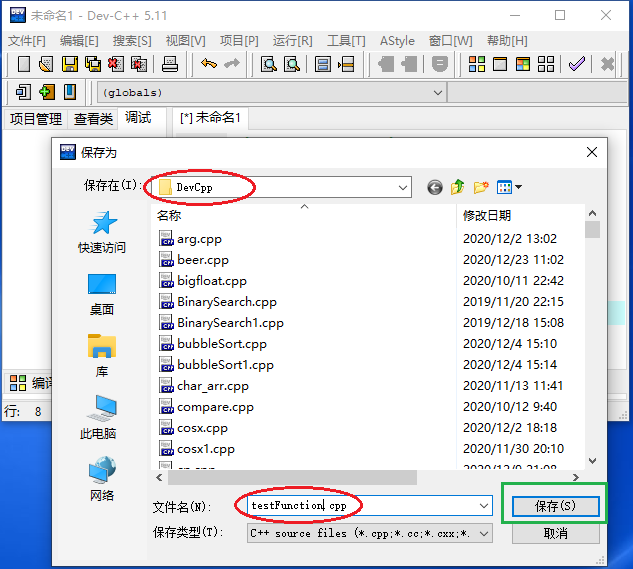


图1-19 源文件保存对话框

**注1：**关于文件的命名。尽管文件的名字只要操作系统能够识别即可，没有什么特殊的限制，但按照软件工程的思想及编程规范，应该为文件起一个有意义的名字。如本例中的testFunction，表示这个实验是要你熟悉测试C的标准函数的使用。

**注2：**尽管可以将你新建的源文件保存到默认的目录中（如我的文档），但建议将文件保存到C盘之外的其它盘上，如在D盘上建立目录（文件夹），如CProgram，在该目录下再建立一个子目录，如lab1，表示实验1的内容，然后将你的文件保存到D:\CProgram\lab1中。

**注3：**关于保存源程序时，文件的扩展名（后缀）问题

一般地，C程序扩展名为.c，C++扩展名多为.cpp。

当文件扩展名为.c时，系统使用gcc编译器将源程序编译链接成可执行程序；而当文件扩展名为.ccp时，系统使用g++编译器。由于C语言是C++语言的一个子集，因此，使用编译器g++也可以正常处理扩展名为.c的C程序。

我们实验使用C而不是C++编程，原则上保存源程序时应使用扩展名.c，而不是.cpp。

但在C语言中，注释语句只能用/\*…\*/，而C++还可以使用//…，为方便起见，可以将文件保存为.cpp类型（但只能用C语言编程，不能用C++，只是注释可以用//）。

5、编译执行程序

可以通过菜单或按钮编译执行你所编写的C程序。

（1）菜单选择“运行/编译执行”，或按快捷键F11，可编译执行你所编辑的源程序，如图1-20所示。

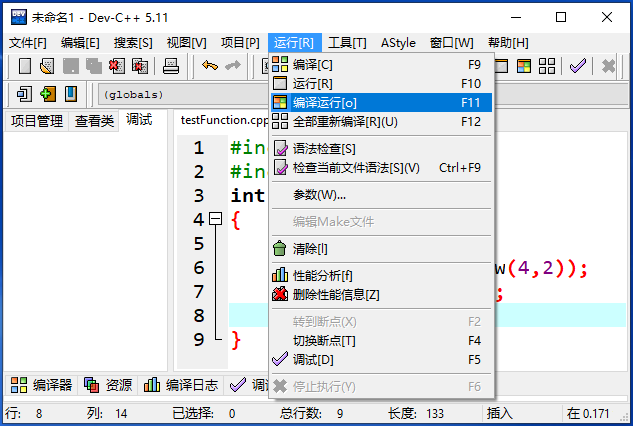


图1-20 通过菜单编译执行源程序

（2）鼠标左键点击红色椭圆标注的“编译运行”图标，编译执行源程序。如图1-21所示。

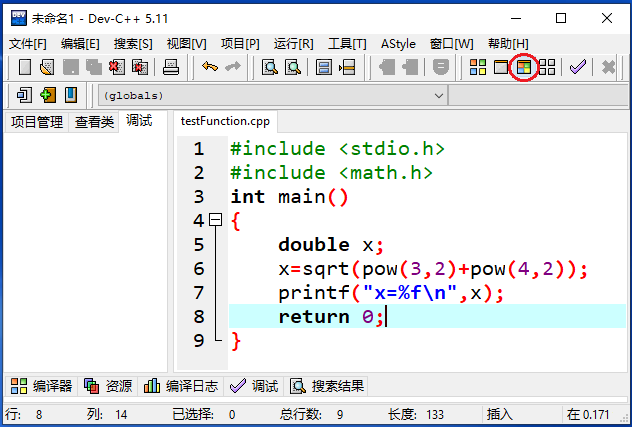


图1-21 通过“编译运行”图标编译执行源程序

如果程序没有语法错误正常执行，执行结果如图1-22所示。

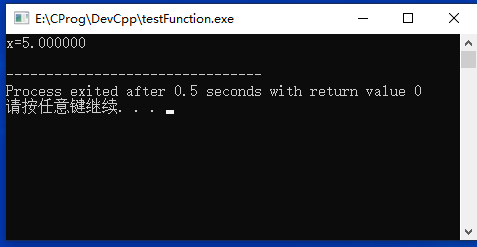


图1-22 程序执行结果

### 1.1.7 打开（重新装入）已保存的文件

选择菜单“文件/打开项目或文件”，或鼠标左键点击“打开项目或文件”图标，打开、查看或调试已经保存的源文件。分别如图1-23与1-24所示。

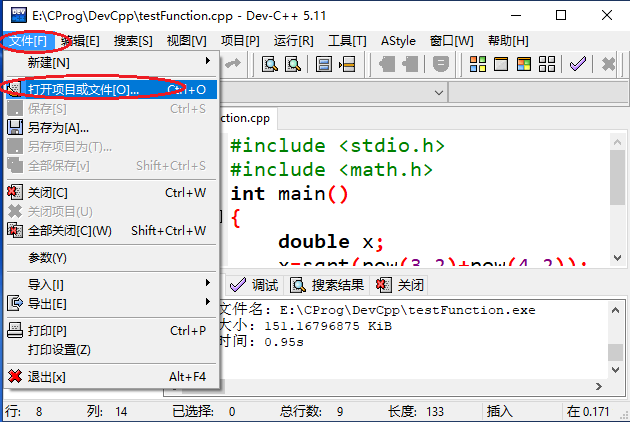


图1-23 利用菜单打开文件

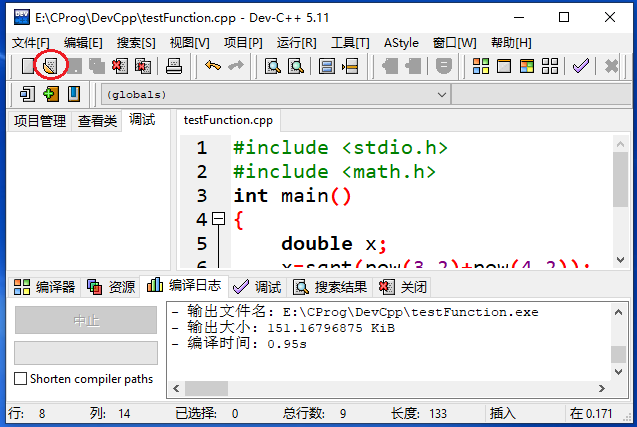


图1-24 通过点击图标打开文件

### 1.1.8 编程计算下述题目，并输出结果

可以用C++的cout输出结果，最好用C的标准输出printf()输出。

(1)

(2)

(3)

(4)

(5)

(6)

(7)

(8) y= 3/4+103除以3的余数

(9) 利用C标准数学函数计算不大于3.8、-3.8的最大整数；

(10) 利用C标准数学函数计算不小于3.2、-3.2的最小整数；

示例程序：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* 程序名：lab1.c \*/

/\* 主要功能： 利用C语言标准函数计算， \*/

/\* 并输出结果 \*/

/\* 作者：xxx \*/

/\* 编制时间：2022年10月x日 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <math.h>

int main()

{

float y;

y=1+(1/(1+(1/(1+1/5.0))));

printf(“f=%f, %e, %g\n”,y,y,y);

return 0;

}

## 1.2 实验2 标准输入输出及数据的机内表示

### 1.2.1 实验目的

熟练掌握C语言标准输入函数scanf()与标准输出函数printf()的使用，以及机数据的机内存储形式。

### 1.2.2 实验任务与要求

1、C语言的标准输入函数scanf()，分别输入short, int, long long long类型的几个数据，然后利用printf()输出（十进制数、无符号数、八进制、十六进制格式），根据输入、输出结果理解C语言整型符号数输入输出的形式，以及数据的机内存储格式；

如：

定义一个int（或short、long，long long、unsigned）型变量，利用scanf()输入-1到该变量中，然后利用printf()输出该变量值机内存储的16进制形式（机器数的16进制）。根据输出的机器数的十六进制形式，加深对补码概念的理解；

定义一个int型变量，输入int型的最小值（最小负数），根据输出的机器数的十六进制形式，加深对补码概念的理解；

定义一个int型变量，输入int型的最大值（最大正数），根据输出的机器数的十六进制形式，加深对补码概念的理解

定义一个long long型变量，分别将一个int型的正整数与负整数赋给该long long型变量，输出该long long型变量的机器数的十六进制形式，理解符号位扩展的思想。

定义一个long long型变量，输入一个long long型的值给该long long型变量，然后将其赋给一个int型变量，根据该int型变量的输出结果，理解8字节数据转换到到4字节空间时，数据截取的思想与实现。注：在给long long型变数赋值时，分别输入一个在int型数据表示范围之内的数，及一个超出int表示范围的值，然后将该long long型变量的值赋给int型变量，根据int型变量的输出结果解释数据截取的思想。

2、分别用scanf()输入unsigned short, unsigned int, unsigned long unsigned long long类型的几个数据，然后利用printf()输出（十进制数、无符号数、八进制、十六进制格式），根据输入、输出结果理解C语言整型无符号数输入输出的格式，以及数据的机内存储形式；

3、分别用scanf()输入float, double类型的几个数据，然后利用printf()输出（一般实数形式，指数形式，十六进制格式等），根据输入、输出结果理解C语言浮点数输入输出的格式，以及数据的机内存储形式；

参考IEEE 754标准中规定的浮点数的机器数存储格式；

在IEEE 754标准中，float型的存储格式：共占用4字节。其中最高位是符号位，紧随其后的8位是阶码（指数，采用移码表示，注：移码采用的是127+真值，不是128加真值），最后的23位是尾数的小数部分，采用原码表示；

其中，阶码的真值是机器数阶码减去127（），尾数的真值是23位尾数+1（采用规格化形式，尾数的小数点的左边隐含存储一位1，因此尾数的真值是1+23位尾数表示的小数部分，即尾数的真值是1.xxx…xxx的形式）

在IEEE 754标准中，double型的存储格式：最高位是符号位，其后的11位是阶码（指数），最后的52位是尾数；（共占用8字节）

其中，指数的真值是机器数阶码减去1023（）），尾数的真值是52位尾数+1（采用规格化形式，1隐含存储在小数点的左边）

例如输入float型数据的几个边界值，分析它们的机内码形式：

float f; 输入 f=0.0, f= -0.0, f=1.0, f= -1.0, f=1e128（+）, f=-1e128（-）；

再输入几个一般的值进行测试分析；

提示：

**printf(“%X”)中的格式”%X”的作用是按照十六进制形式输出一个整数的机器数；**

**printf(“%o”)中的格式”%o”的作用是按照八进制形式输出一个整数的机器数；**

**如果采用格式”%X”输出浮点数，只输出浮点数整数部分的机器数；**

因此，若以十六进制形式输出一个单精度的浮点数（float）的机器数，需要让系统将该单精度浮点数所占用的四个字节中的内容理解为一个四字节的整数（int，或unsigned）。

同理，若以十六进制形式输出一个双精度的浮点数（double）的机器数，需要让系统将该双精度浮点数所占用的八个字节中的内容理解为一个八字节的整数（long long，或unsigned long long）。

输出单精度浮点数的十六进制形式代码片段：

float f= - 1.0; //4字节存储

unsigned \*fp; //指向4字节的无符号数指针；

fp=(unsigned \*)&f; //将4字节的浮点数理解为4字节的无符号数

printf(“f=%f, 0x%08X\n”, f, \*fp);

输出双精度浮点数的十六进制形式代码片段：

double d= - 1.0; //8字节存储

unsigned long long \*dp; //指向8字节的无符号数指针；

dp=(unsigned long long \*)&d; //将8字节的浮点数理解为8字节的无符号数

printf(“d= %f, 0x%016llX\n”, d,\*dp);

测试用例：

对于单精度浮点数：

-1.0的机器数：0xBF800000

+1.0的机器数：0x3F800000

-1e128的机器数：0xFF800000 （-）

+1e128的机器数：0x7F800000 （+）

- 0.0的机器数：0x80000000

+0.0的机器数：0x00000000

12.6的机器数：0x4149999A

对于双精度浮点数：

-1.0的机器数：0xBFF0000000000000

+1.0的机器数：0x3FF0000000000000

-1e1024的机器数：0xFFF0000000000000 （-）

+1e1024的机器数：0x7FF0000000000000 （+）

- 0.0的机器数：0x8000000000000000

+0.0的机器数：0x0000000000000000

12.6的机器数：0x4029333333333333

对于单精度浮点数，-1.0的机器数为0xBF800000，对应的二进制形式为：

1011 1111 1000 0000 … 0000；

按照IEEE 754标准解释，将上述二进制数划分成3段：1 01111111 000…0。

最高位是浮点数的符号位；该例中符号位是1，表示该浮点数是一个负数；

中间的8位01111111是阶码部分（指数），是指数对应的移码形式（注：这里的移码是[x]移=127+x，与标准的移码定义[x]移=128+x稍有区别），该例中指数的移码是127，因此指数的真值是127-127=0；

最后的23位是尾数的小数部分（这里是0000…0000），根据约定，尾数的小数点左边是1（尾数的整数部分是1），因此尾数的真值是1+0.00...000=1.00…00，因此，机器数0xBF800000对应的浮点数的真值为：

同理，机器数0x80000000对应的浮点数真值是，由于阶码是最小值-127，被认为是单精度浮点数真值;（机器零）

机器数0x7F800000对应的浮点数真值是，其中的阶码是最大值128，被认为是单精度浮点数的；

同理可以分析、理解任一浮点数的机内存储格式。

关于浮点数的0.0与无穷大的几点注记：

注1：指数部分采用移码表示，这里移码的定义：

对于单精度浮点数，指数部分是一个字节，[x]移码=127+x，不是128+x；

对于双精度浮点数，指数部分是11位，[x]移码=1023+x；

注2：指数部分的机器数（移码）的范围是：0~255，因此其真值的范围是：-127~128；

对于一个规格化形式的单精度浮点数，当指数取其最小值 -127时（移码形式是00000000），认为该浮点数的值非常接近0（+0或-0），就认为该浮点数的值是0.0或-0.0（只是被机器当做零，因此有时称为机器零）；

对于一个规格化形式的单精度浮点数，当指数取其最大值 128时（移码形式是11111111），认为该浮点数的值非常大，就认为该浮点数的值是或。

4、用scanf()输入字符型数据，然后利用printf()输出（字符，十进制、十六进制等），根据输入、输出结果理解C语言字符输入输出的格式，以及数据的机内存储形式，理解系统对非数值型数据的编码原理，如字符的ASCII码。

## 1.3 实验3 循环、分支、枚举

### 1.3.1 实验目的

熟练掌握C语言循环与分支语句的使用，掌握利用枚（穷）举法解决问题的思想。

### 1.3.2 实验任务与要求

1、计算的近似值;

当最后一项的绝对值小于等于时，忽略掉以后的项。

提示：为提高程序的执行效率，不要使用这种方法（n为自然数）；可以考虑：一个数n，连续乘以-1，n的符号会正负相间；

测试用例：当最后一项的绝对值小于等于时，，大约循环50万次；（有的环境输出）

ps. 当最后一项的绝对值小于等于时，忽略掉以后的项，则

，大约循环500万次；

当最后一项的绝对值小于等于时，忽略掉以后的项，则

，大约循环5000万次；

当最后一项的绝对值小于等于时，忽略掉以后的项，则

，大约循环5亿次；

2、函数sin(x)的泰勒级数展开式如下

编程计算sin(x)的值，精度要求。将你的计算结果与C库函数的结果进行比较，思考C库函数对三角函数的实现方法。

提示：当x比较大，如x=100000， 且精度要求很高时，与会产生溢出，如何解决？

3、喝不完的啤酒。啤酒2元1瓶，2个空瓶换1瓶，4个瓶盖换1瓶，问10元钱最多可以喝几瓶？（不允许赊账）

测试用例：喝瓶酒15瓶，剩余空瓶1个，瓶盖3个，钱0元；

思考：如果允许赊账，最多可以喝多少瓶啤酒？（最多喝20瓶，空瓶、瓶盖、及钱数无剩余）

4、今有物，不知其数。三三数之，剩二；五五数之，剩三；七七数之，剩二。问：物几何？（注：根据题意，给出多个解中的最小值即可）

5、输出乘法口诀表，格式如下：

1\*1=1

1\*2=2 2\*2=4

1\*3=3 2\*3=6 3\*3=9

…

1\*9=9 2\*9=18 3\*9=27 4\*9=36 5\*9=45 … 9\*9=81

要求：

（1）利用循环结构；

（2）不允许直接利用类似于语句printf(“1\*3=3 2\*3=6 3\*3=9\n”)输出。

（3）位置对齐可在printf中使用”\t”；

## 1.4 实验4 进制之间的转换

### 1.4.1 实验目的

熟练掌握数组的定义与使用，以及各进制之间的转换方法。

### 1.4.2 实验任务与要求

编程实现下列数据之间的进制转换，并输出结果；

1、十进制转换成2进制、8进制及16进制；

2、2进制、8进制，16进制转换成十进制；

3、2进制与8进制，16进制之间的转换；

4、转换方法要求利用传统的进制转换方法，不能利用*itoa()*系列函数进行自动转换。

要求：

将数制之间的转换方法利用程序实现，不允许直接利用printf()的“%d”、”%o”及”%x”等格式由机器自动转换。，

测试用例：

键盘输入十进制整数：235，输出对应的二进制，八进制及十六进制数；

23510=111010112=3538=EB16

键盘输入十进制小数：0.23579，输出对应的二进制数；

0.2357910=0.0011 1100 0101 1100 1100 11002

键盘输入二进制整数：10101101，输对应的十进制数；(173)

键盘输入二进制小数：0.1110011，输出对应的十进制数；(0.898438)

键盘输入二进制整数：11101101，输对应的4进制，8进制及16进制数；4进制：3231，8进制：355，16进制：0xED

键盘输入16进制整数12Def，输对应的2进制，4进制，8进制数；

16进制12Def对应的二进制：10010110111101111，4进制：1023233233, 8进制：226757

## 1.5 实验5 数组、枚举法求解0/1背包问题算法

### 1.5.1 实验目的

熟练掌握C语言数组在循环结构中的使用，掌握枚举法解决优化问题的思想与方法。

### 1.5.2 实验任务与要求

1、编程实现矩阵的加法、乘法及转置运算。

加法：；

减法：；

乘法：；

转置：将矩阵转置成；

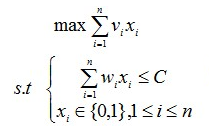
2、利用枚举法解决0/1背包问题。

问题描述

给定种物品和一背包。物品的重量（体积）是，其价值为，背包的容量为C。问：应如何选择装入背包的物品，使得装入背包中物品的总价值最大。

形式化描述

给定要求找一元向量 X=，即：

****

请分别利用枚举法（穷举法）解决上述问题。

枚举法（穷举法）肯定能找到最优解。

提示：

**【枚举法】：**

变量定义：

设表示n种物品的重量， 表示n种物品的价值；

表示问题的最优解，若=1，表示将第i种物品装入背包，最优解对应的背包重量为Weight， 价值为Value；初始化Weight=0， Value=0；

C表示背包的容量（所能容纳的重量）；

Y表示目前n种物品的一种装法（解法），按该装法，背包中物品的重量为CurWeight ，背包中物品的价值为CurValue ；

算法描述（伪代码）

for (i=0 to -1)

{

将*i*对应的二进制数按位存放到数组Y中；

尝试按照装包方案y进行装包，则背包中物品的重量为CurWeight, 价值为CurValue；

如果CurWeight>C，则丢弃该方案，继续尝试下一种方案；否则，若该方案优于以前的方案，即如果CurValue>Value，则暂存该方案：即X=Y，Value=CurValue， Weight= CurWeight。 继续循环；

}

输出最优解X和对应的物品的相关信息， 背包中物品的价值Value，背包中物品的重量Weight；

测试用例：

背包容量C=150，7种物品，其重量W=(10,40,30,50,35,40,30)，价值V=(35,30,6,50,40,10,25) (W与V均为浮点数)

枚举法得到的最优解：装入物品1,2,4,5，背包重量135，最大价值155；

## 1.6 实验6 字符数组与字符串

### 1.6.1 实验目的

熟练掌握字符数组、字符串、循环语句及条件语句的使用。

### 1.6.2 实验任务与要求

1、字符串的输入输出。

（1）利用scanf(“%s”,…)输入字符串（字符数组），然后利用printf(“%s”,…)输出，根据输入、输出结果理解C语言字符串的输入输出的格式；

（2）输入一个含有空格的字符串（如将”hello world”作为一个字符串）赋值给字符串变量，然后输出该字符串；

提示：利用格式scanf(”%[^\n]”,str)输入；

注意输入缓冲区的数据残留；如scanf(”%[^\n]”,str)语句会造成输入缓冲中残留回车换行符。

（3）逐个输出字符串中每个字符的的编码（ASCII码），检测字符串是否以’\0’作为结束符。

（4）利用循环结构连续调用scanf(”%[^\n]”,str) 输入多个字符串，根据str的输出结果，观察是否在输入一个字符串后，后续的scanf(”%[^\n]”,str)中，会将缓冲区中残留的回车换行符读入到str中。

根据str的输出结果理解输入缓冲区残留的问题，并给出相应的解决方案，以消除数据残留所带来的副作用。

2、密码输入

设定一个密码，如：const char \*password=”onlyyou”;

键盘输入一个字符序列，以回车键结束输入。如果输入的字符序列与password中保存的密码**恰好相同**，则输出”valid user”，程序退出；否则输出”invalid user”，并提示用户重新输入密码，允许用户最多尝试5次密码输入。

要求：

（1）在输入密码字符序列的过程中，每输入一个字符，显示星号”\*”，不能显示所输入密码字符的内容。

（2）允许退格：若某一个或几个字符输入错误，允许利用退格键，删除一个或多个已输入的字符，将光标退回到用户希望的位置，以重新输入正确的字符；

注：光标每退回一格，应删除光标所在位置的字符，并显示空格。

（3）若5次均输入错误，则输出“5次密码输入错误，锁定用户账户”等字样。

提示：printf("\b"); //光标退后一格

3、键盘输入一个英文单词，输出其对应的复数形式

规则如下：

（a）以辅音字母y结尾，则将y变成i，再加es；

（b）以s,x,ch,sh结尾，加es；

（c）以元音o结尾，加es；

（d）其它情况，加s；

# 第2章 提高模块

提高模块实验的主要目的是在学生具备一定的C语言编程能力后，运用所学知识编程解决一些较为复杂的实际问题。

## 2.1 实验7 递推

### 2.1.1 实验目的

熟练掌握C语言数组在循环结构中的使用，掌握递推问题的设计与实现方法。

### 2.1.2 实验任务与要求

1、编程求解青蛙跳台阶的问题：青蛙跳台阶问题：有n个台阶，每次青蛙可以选择跳1个台阶、2个台阶或者3个台阶，青蛙爬到第n个台阶，共有多少种走法？（假设）

测试用例：

|  |  |  |
| --- | --- | --- |
| 台阶n | 走法 | 备注 |
| 1 | 1 | 跳1个台阶，表示为1 |
| 2 | 2 | 两个1步，或一个2步，表示为11或2 |
| 3 | 4 | 四种走法：111，12，21，3 |
| 4 | 7 |  |
| 8 | 81 |  |
| 10 | 274 |  |
| 20 | 121415 |  |
| 50 | 10562230626642 |  |
| 60 | 4680045560037375 |  |
| 70 | 2073693258389777176 |  |
| 80 | 14947546285095418801 |  |
| 81 | ？？？ | 思考：为什么给出的假设？ |

2、中国政府对新冠疫情施行严格防控，成效明显。如果放任自流，假设1位新冠患者每天会感染10人。若不考虑死亡、治愈、隔离等其它情况，理论上在第一天有两位患者，则第n天会有多少新冠患者？

测试用例：

|  |  |  |
| --- | --- | --- |
| 天数n | 患者数 | 备注 |
| 1 | 2 |  |
| 2 | 22 | 嗯，可控 |
| 10 | 4715895382 | 哇，传染速度如此之快 |
| 20 | 11637716454571782886 | 佩戴口罩，定点丢弃，及时更换 |
| 21 | 17334416558032302050 | 少出门，少聚集，勤洗手，勤通风 |
| 22 | ??? | 思考：为什么第22天没有给出结果？ |

## 2.2 实验8 编译预处理

### 2.2.1 实验目的

熟练掌握C语言编译预处理中宏定义、包含以及条件编译的使用。

### 2.2.2 实验任务与要求

1、宏定义（#define）

（1）首先利用宏定义的值，如 #define PI 3.141592，然后定义另一个带参数的宏，以半径r作为宏的参数，分别计算圆的周长、面积及球的体积，其中第二个宏定义中的使用已经定义的宏PI。

程序中输入两个数，如a,b，(a+b)作为圆及球的半径，利用你定义的宏分别计算半径为(a+b)的圆的周长、面积及球的体积，测试定义宏时小括号()的作用，进一步理解宏展开的过程与方法。

（2）定义一个带参数的宏，在宏定义中判断该参数所代表的数据是否为整数，并输出相应的信息。

定义与使用宏方式如下：

#define INTEGER(x) …. //判定x是否为整数，并输出相应信息。

int main()

{

INTEGER(8); // 屏幕输出：8 is an integer.

INTEGER(10.3); // 屏幕输出：10.3 is not an integer.

return 0;

}

提示：宏“#” 用于替换文本，如：

#define HI(m) printf(“Hi,” #m “!\n”)，则

HI(there); //将上述宏中的#m替换成“there”，输出“Hi, there!”

（3）自己定义其它你想使用的宏，并编程测试；

如：#define PRd(s,a) printf("%s %d\n",s,a)

编写代码如：int b=-20; PRd(“b=”,b)；

2、包含（#include）

在同一个工程中创建一个包含main()的源文件，再创建另一个源文件，如labxx.h（其中的xx是该实验的编号），将所有涉及的头文件在labxx.h中声明，然后在包含main()函数的文件中利用#include “labxx.h”将涉及到的头文件包括进来，以测试编译预处理#include的作用。

**3、（选做）**条件编译（#ifdef ….#else …..#endif，或 #ifndef ….#else …..#endif）

（1）为防止MAX\_SIZE重复定义，造成混乱，可使用如下如下定义方法：

#ifndef MAX\_SIZE

#define MAX\_SIZE 1024

#endif

（2）条件编译代码示例

下述程序片段保证系统为整型变量a分配4个字节的内存空间；

关于宏CPU16与CPU32定义的位置：

1. 可在在编译程序gcc中用参数 –DCPU16，或–DCPU32对CPU16或CPU32进行定义；
2. 在程序中的合适位置利用#define CPU16，或#define CPU32定义；

我们实验采用第二种方法定义，如下：

//#define CPU16

#define CPU32

#ifdef CPU16 //相当于#if defined (CPU16)

long a=3;

#else

int a=5;

#endif

或：

#if definded (CPU8) || (CPU16)

long a=3;

#else

int a=5;

#endif

如果防止出现a未定义的错误，可采用如下方式：

#ifdef CPU16

long a=3;

#elif defined (CPU32)

int a=4;

#else

int a=5;

#endif

或：

#if definded (CPU8) || (CPU16)

long a=3;

#elif defined (CPU32)

int a=4;

#else

int a=5;

#endif

还有其它形式的条件编译，如：

//-----------------------------

**#ifndef** CPU16 //相当于 #if **!defined** (CPU16)

….

#else

…..

#endif；

//-----------------------------

#define test 1

#if 0 //if !1 //if !test //注意与#ifdef或#ifndef的不同

….

#else

…….

#endif

//------------------

自己设想一种条件编译的场景，熟悉条件编译的使用；

## 2.3 实验9 排序、查找

### 2.3.1 实验目的

掌握冒泡排序、线性查找与折半查找的思想与程序实现。

### 2.3.2 实验任务与要求

1、给定一组整数（包括正整数与负整数），利用线性法查找某一元素，如果找到，给出该元素所在的位置，否则，给出找不到的信息。

2、给定一组整数（包括正整数与负整数），利用冒泡法将数据按升序排列。

3、在上述排序后的数据中，实现折半查找。

## 2.4 实验10 随机数与蒙特卡罗算法

### 2.4.1 实验目的

理解伪随机数的产生方法以及产生随机数时种子的含义。掌握随机数的使用，与蒙特卡罗算法的设计与实现方法。

### 2.4.2 实验任务与要求

1、编程利用线性同余法产生n个随机数，理解随机数产生方法，以及随机数种子的作用与含义；

线性同余法是一个经典的伪随机数发生器，算法如下：

a）将X(0)作为随机数的种子；

b）根据递推公式X(n+1) = (a\*X(n)+b) mod M，依次产生一个随机数序列

X(1),X(2)，…

其中，M的作用是产生0到M-1之间的伪随机数，因此M的大小直接影响伪随机数序列的周期长短，故M尽可能取比较大的值；

a和b的值越大，产生的伪随机数也越均匀

如果a和M如果互质，产生随机数效果更好。

例如：a.b可定义为int型；

数组X可定义为int型，但要防止a,b取值太大，造成X[i]溢出；

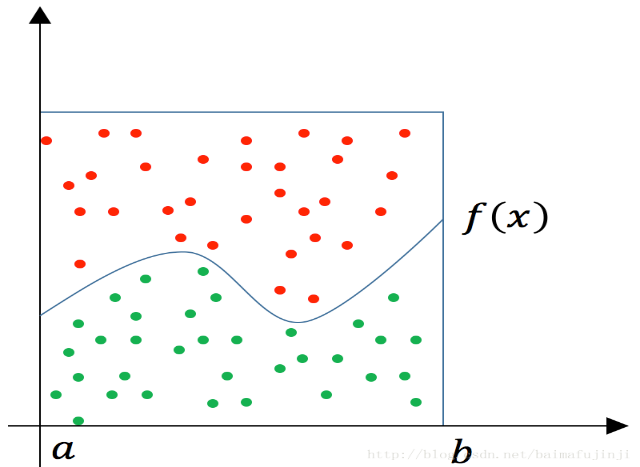
M可取随机数的最大值RAND\_MAX(0x7FF)；

种子X(0)可依次取1,2,3,…,或取当前时间作为种子，测试你生成的随机数；

2、利用蒙特卡罗法计算定积分与的近似值。

（1）计算定积分的近似值

函数f(x)从a到b的定积分，就是求f(x)曲线下方的面积，如下图所示。



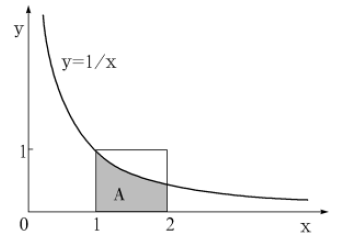
可以用一个比较容易计算面积的矩型包含函数的积分区间，然后随机地向这个矩形框里面投点，假设落入矩形区域内的点数为M，落在函数f(x)下方的点数为N，则面积之比即为点数之比。

设矩形面积为A，f(x)下方的面积为B，则有，即，由此可计算定积分的近似值。投点的数量越来越多，其近似值也越接近真实值。

**利用上述思想编程计算的近似值。**

由于，因此求得的值为的近似值。

定积分 图像如下图所示。



提示：取个随机点，限定，则这个点都在所标矩形的范围内；

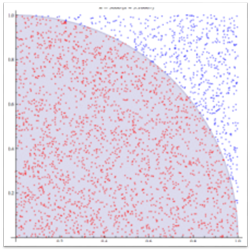
矩形的面积为A=(2-1)\*(1-0)，阴影部分的面积设为S；

若这个点中有个落入阴影内，则，即为所给定积分的近似值。

随机数的产生于使用方法参见教材第12章：蒙特卡洛法。

（2）计算的近似值（选做）

问题描述：在一个边长为1的正方形中划定一个半径为1的1/4圆区域（扇形），如下图所示：



向这个正方形区域内随机投点，设落在整个正方形内的点数是C，落在1/4圆（扇形）内的点数是D，则点数之比即为面积之比，即 则有。

根据概率理论，投点的数量越来越多，其近似值也越接近真实值。

## 2.5 实验11 函数与变量的作用域

### 2.5.1 实验目的

掌握函数的定义、声明与调用过程；

掌握在一个工程中多个文件之间协作编程的思想。

关于在Dev-Cpp中涉及到工程（项目）的概念及工程文件的使用，请参阅附录3的有关内容。

### 2.5.2 实验任务与要求

要求：

（1）**主函数与其它自定义函数在同一个工程的不同文件中实现；**

（2）在实现其它自定义函数的文件中，定义一个或多个全局变量，在main()函数中访问；或在main()函数所在的文件中，定义几个全局变量，在其它函数中访问这些变量，掌握extern对外部变量的声明与使用方法。

（3）同时熟悉利用static，register，auto对变量进行声明与使用的方法。

1、函数实现，模拟rand()与srand()的功能。

注：在C语言标准头文件stdlib.h中，有涉及到随机数的函数及宏定义，如下所示：

#define RAND\_MAX 0x7fff

int rand(void);

void srand(unsigned int \_Seed);

其中，函数rand()返回一个0~ RAND\_MAX(0x7FF)之间的随机整数；函数srand()设置随机数的种子。

在一个程序中，多次调用rand()，会得到一个随机数序列，不妨设为(a,b,c,d,e,…)，若重新运行该程序，得到的随机数序列是相同的，仍然是(a,b,c,d,e,…)，原因是每次重新运行程序，函数rand()使用的初始种子是相同的（回顾实验7中种子x(0）的作用)。

如果利用srand((unsigned) time(NULL)，或srand((unsigned) time(0))，将当前时间作为随机数的种子，由于每次运行时时间是不同的，则每次运行程序产生的随机数是不同的。

编写函数int Rand(void)及void Srand(unsigned int Seed)，要求如下：

1. 利用自定义的函数Srand()设置自定义函数 Rand()的种子；
2. 每次调用函数Rand()，返回一个随机数；但在同一个程序中，多次调用函数Rand()，返回不同的随机数。

例如：int a,b,c;

a=Rand(); b=Rand(); c=Rand();

a,b,c的值不同；

要求将int Rand(void)及void Srand(unsigned int Seed)在一个文件中，主函数main()在同一个工程中的不同文件中。

2、密码学中经常涉及到两个大整数的运算；请编写实现两个30位十进制整数的加法运算的函数。

主函数中输入两个30位的十进制整数，调用你编写的函数，将这两数做加法运算，并输出运算结果。

3、函数指针作为函数的参数。

函数int process(int x, int y, 函数指针pf)，将函数指针作为函数的参数；

在函数process()中通过形参pf（函数指针）分别调用下述三个函数，完成表达式x+y，及x\*y的计算。

（1）函数int add(int x.int y) 实现x+y的功能；

（2）函数int power(int x.int y) 计算；

（3）函数int multi(int x, int y) 计算x\*y；

## 2.6 实验12 递归算法与函数的递归调用

### 2.6.1 实验目的

掌握利用递推与递归求解问题的思想与实现方法；

掌握在一个工程中多个文件之间协作编程的思想。

### 2.6.2 实验任务与要求

实验任务：编程实现下列各题：

1、函数实现。分别利用**递推法**与**递归法**求斐波那契序列第n项的值，计算一个比较大的n，如50，比较两种方法的执行效率（给出每种方法所需时间）。

斐波那契序列的定义：，n为自然数

2、利用递归实现。在一组有n个元素的有序数据中折半查找关键字key，如果找到，返回key所在的位置，否则返回-1。

3、利用递归法找出一个数组中的最大元素。

4、小明有一堆硬币若干枚，其中一枚是假币，外观上这枚假币与真币无法区分，但假币比真币轻（每枚真币的重量都相同）。目前小明手上的工具只有一台天平。

（1）假设硬币有枚（n为正整数）。在查找假币的过程中，若要求平均比较次数尽量小（不能超过），请编程帮小明找出其中的假币，输出假币的编号，并分析你编写的程序的时间复杂度。

（2）假设硬币有n枚（n为正整数），请设计不限定比较次数的查找假币程序，时间复杂度不超过O(n)。

**BONUS**：如果无法知道假币是否比真币轻还是比真币重，如何解决上述问题？

## 2.7 实验13 结构与链表、贪心法求解0/1背包问题

### 2.7.1 实验目的

掌握结构类型、结构指针与链表的相关操作。

### 2.7.2 实验任务与要求

1、给定n种物品，基于结构体与结构数组，利用贪心法解决0/1背包问题。

贪心策略：优先选择单位重量价值最大的物品装入背包；

定义每件物品的结构：

struct GOOD

{

int No; ///物品编号

float weight; //物品重量

foat value; //物品价值

float pw; //物品单位重量的价值； pw=value/weight;

}

//提示：

函数原型：void greedy(struct GOOD goods[], int n); // goods[]为n种物品的结构数组；

将n种物品的结构数组goods[]按照物品结构中的pw进行降序排列；

从数组goods[]的第一个元素开始，顺序尝试将数组元素对应的物品装入背包，并依次输出装入背包的物品的相关信息；

当尝试到将某个元素对应的物品装入背包后，背包中物品的总重量大于背包容量，则装包方案即为该元素之前的数组元素对应的所有物品；

最后输出装入背包中物品的总重量与总价值；

测试用例：

背包容量C=150，7种物品，其重量W=(10,40,30,50,35,40,30)，价值V=(35,30,6,50,40,10,25) (W与V均为浮点数)

贪心法得到的解：装入物品1,5,4,7，背包重量125，最大价值150；

比较：利用枚举法得到的最优解：装入物品1,2,4,5，背包重量135，最大价值155；

2、链表的基本操作

定义结构体：

struct data {

int n;

struct data \*next;

}

函数实现：

（1）给定一组无序整数，按给定数据的顺序建立一个单向链表；返回链表的头指针；

（2）将链表中的结点按数据进行排序，构成一个按数据升序排列的有序单向链表；返回有序链表的头指针。

（3）链表的遍历：按升序输出链表中的所有数据；

函数原型：void output(data \*head); //从头指针head开始依次输出链表中的所有数据；

（4）编写链表结点的插入函数。

函数原型：data\* insert(data \*head, int N); //在以head为头指针的升序排列的链表中，将整数N插入到链表的合适位置，使插入后的链表仍为升序排列；返回链表的头指针；

（5）实现链表结点的删除函数：在链表中删除一个结点；

函数原型：data\* remove(data \*head, int N); //在以head为头指针的升序排列的链表中，删除整数N对应的结点，返回链表的头指针；

3、（Bonus，选做）利用链表实现多项式的加减运算。

定义结构体：

struct item {

int a;

int n;

struct data \*next;

}

每个结果体表示一个多项式的一项，成员a表示该项的系数，成员n表示该项的幂次。

将一个多项式利用链表表示，然后定义两个多项式，实现多项式的加减运算，结果也采用链表表示。

## 2.8 实验14 文件与命令行参数

### 2.8.1 实验目的

掌握文件的相关操作及C语言命令行参数的使用。

### 2.8.2 实验任务与要求

1、创建一个文件，为该文件预留128K的空间。

2、在程序的命令行输入两个文件的名字，如a.c b.c，将文件a.c的内容复制到文件b.c中。

要求：

（1）如果文件a.c不存在，给出相应提示，程序退出；

（2）如果文件b.c不存在则自动创建之；如果文件b.c已经存在，提示是否覆盖。如果不允许覆盖，则退出，否则覆盖已存在的文件b.c。

# 第3章 拓展模块

拓展模块实验的主要目的是在学生熟练掌握了C语言的基本知识，并具备比较高的编程能力后，综合运用所学知识，编程实现一些复杂的实际问题。

## 3.1 实验15 学生成绩管理系统

### 3.1.1 实验目的

实现一个学生成绩管理系统，以训练学生对C语言的结构类型定义、结构变量的声明方法，文件的打开、读写、关闭等操作，以及C语言综合知识的使用能力。

### 3.1.2 实验任务与要求

利用C语言的结构类型与文件操作，实现一个学生成绩管理系统。

要求：

（1）每个学生的记录包括：学号、姓名、数学成绩、英语成绩、计算导论与程序设计成绩；

（2）功能：菜单提示：成绩录入、修改、添加、删除、查询统计、退出等功能。

如按学号、姓名、各科成绩查询（包括成绩段查询），按分数段统计人数等

（3）每个同学记录可以用结构struct类型实现，录入修改后保存到文件中，以后从文件中读取学生信息并进行相应的操作。

提示：

1、关于菜单

可以使用多级菜单，即程序开始执行，系统进入主菜单供用户选择，用户根据菜单选择一个功能后，可根据需要进入下一级菜单。

选择一个操作后，应该回到本级菜单供用户选择，用户选择返回上级菜单后可返回上级菜单

只有主菜单中可以根据用户的选择退出程序；

2、关于对学生结构的写入与读取

对于学生的信息以结构为单位写入到文件，读取时也是以结构为单位。

输入学生信息时，可以将每个学生的信息存储到一个结构中，然后使用fwrite()将学生的结构写入文件中。

读取文件中的学生信息时，使用fread()将学生的结构从文件中读出，然后访问结构成员变量对学生信息进行解析。

## 3.2 实验16 五子棋（Bonus，选做）

### 3.2.1 实验目的

实现一个五子棋游戏，以训练学生对C语言综合知识以及相关算法的设计与实现能力。

### 3.2.2 实验任务与要求

双方分别使用黑白两色的棋子，下在棋盘直线与横线的交叉点上，先形成五子连线者获胜。

注：横向五子连线、纵向五子连线、斜向五子连线均视为五子连线。

要求：完成下列功能之一即可。

（1）两人对弈

（2）人机对弈（人先手，或人后手均可）

（3）机器模拟两人对弈

# 附录1 实验报告格式

山东大学 计算机科学与技术 学院

计算导论与程序设计 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号： | 姓名： | | 班级： |
| 实验题目： | | | |
| 实验学时： | | 实验日期： | |
| 实验目的： | | | |
| 硬件环境：  （所使用的机器的型号、CPU型号等硬件信息） | | | |
| 软件环境：  （所使用的操作系统，编程环境IDE的名称，版本等软件信息） | | | |
| 实验内容与实验步骤： | | | |
| 结论分析与体会： | | | |

附录：核心程序代码片段、调试过程、测试用例及测试结果（运行结果截图等）

# 附录2 实验报告与源代码文件命名与提交

一、需要提交的材料

实验报告+源代码

二、实验报告及源代码文件的命名规则

1、每个实验分别单独建立一个实验报告文件，实验报告格式见附录1；

**实验报告命名规则：**学号 姓名 实验x 实验名称

如：2021xxxx 小明 实验1 编程环境及标准数学函数

2、**实验代码命名规则：**学号 姓名 实验x 实验名称（源代码）

如：2021xxxx 小明 实验1 编程环境及标准数学函数（源代码）

注：源代码包括头文件、C源程序等，解压后能够直接编译运行。

不需要提交编译后的可执行程序（.exe文件）

3、将实验报告与源代码压缩成一个包

**压缩包命名规则：**学号 姓名 实验x 实验名称

如：2021xxxx 小明 实验1 编程环境及标准数学函数

三、实验报告的提交

每个实验提交一个压缩包，压缩包中包括实验报告与实验代码。（见二）

助教老师按班级将同学的文件归类，分别保存到以“班级/实验“命名的目录

中。如：1班/实验1

四、提交方式

以班为单位提交；

五、对于上述问题，助教可根据具体情况做出相应的调整。

# 附录3 在Dev-Cpp中IDE中使用工程（项目）

工程（project），在Dev中称为项目。对于2.4节实验10（函数与变量的作用域），需要使用工程。其后的实验，也建议使用工程的概念，对软件进行有效的管理。

**一、在工程中编程与运行**

**1、新建一个工程**

在菜单中选择“文件/新建/项目”，如图1所示，也可以通过点击“新建”图标创建一个新的工程。随即弹出“新项目”窗口，如图2所示。

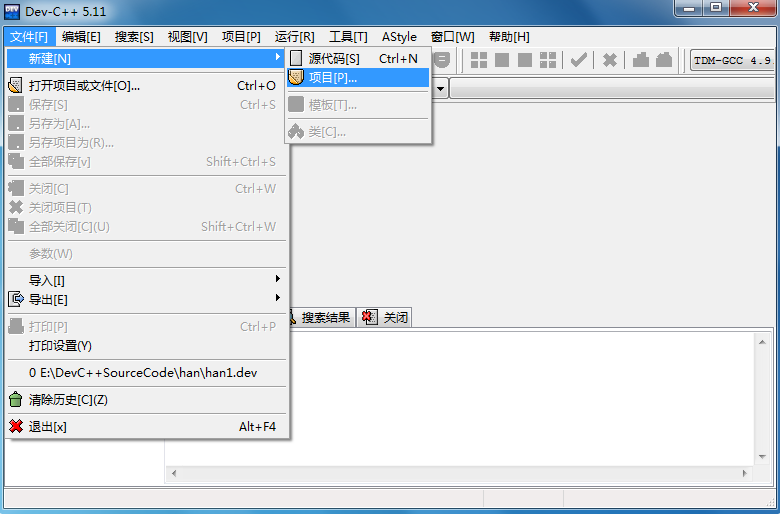


图1 新建一个工程（项目）

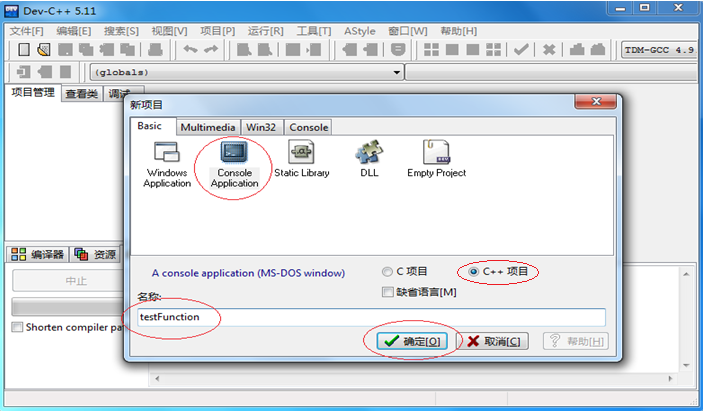


图2 新项目（New Project）窗口

在图2中，选择“Console Application”，选择“C项目”或C++项目，在名称文本框中输入工程（项目）名称，点击确定按钮。

**注1：**我们采用C语言而不是C++语言编程，因此，原则上我们应该选择C项目，而不是C++项目。在1.1.6中我们提到，C语言是C++的一个子集，使用C++的编译器可以编译C程序。因此，选择C++项目不影响我们使用C语言编程。

**注2：**尽管对工程的命名没有什么限制，但按惯例应该根据实验内容或其它需求起一个有意义的名字。

在随后弹出的窗口中询问你要将该工程保存到什么位置，如图3所示。

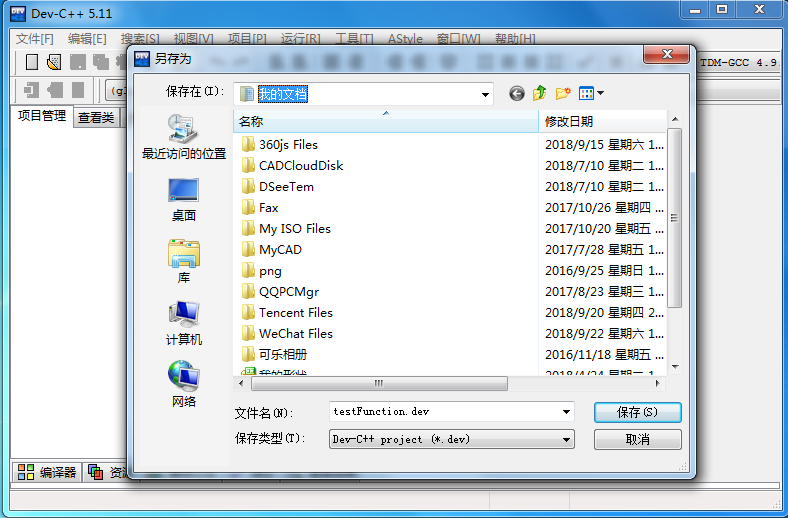


图3 新建工程保存位置

尽管可以将你新建的工程保存在默认的目录中（如我的文档），但仍然建议在C盘之外的其它盘上，如D盘，建立目录，如CProgram，在该目录下再建立一个子目录，如lab1，表示实验1的内容，然后将你的工程文件保存到D:\CProgram\lab1中，如图4所示。

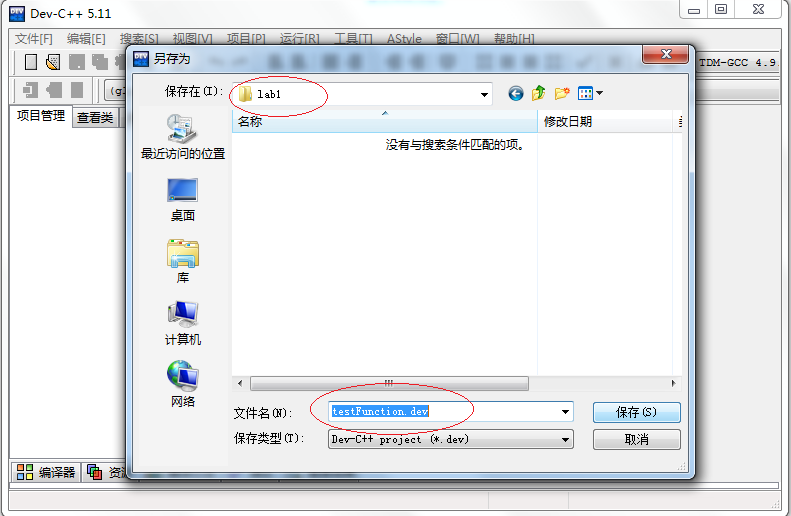


图4 将工程文件保存到D:\CProgram\lab1中

随后弹出如图5所示的窗口，你可以在该窗口中输入编辑你的C程序。

注：对于一个新建的工程，默认C程序的主函数mian()所在的文件为main.cpp。

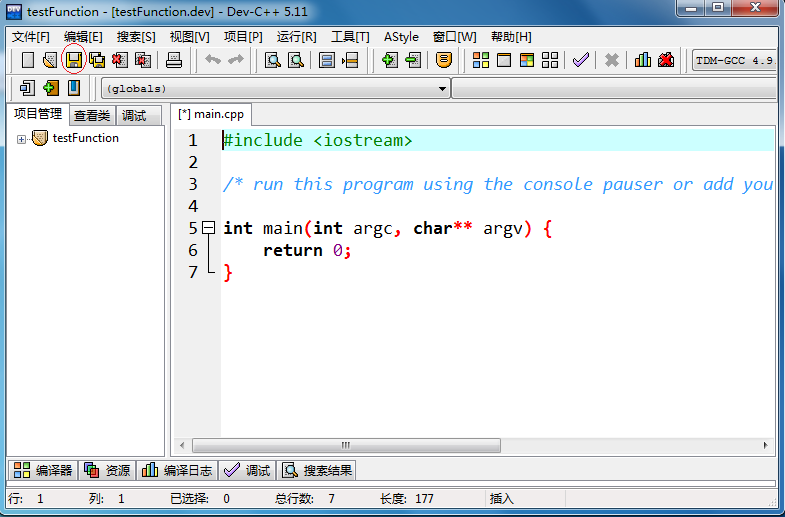


图5 C程序编辑窗口

点击如图5中“编辑”菜单下面的图标，可保存该文件，如图6所示。（也可以通过菜单“文件/保存”完成上述操作）

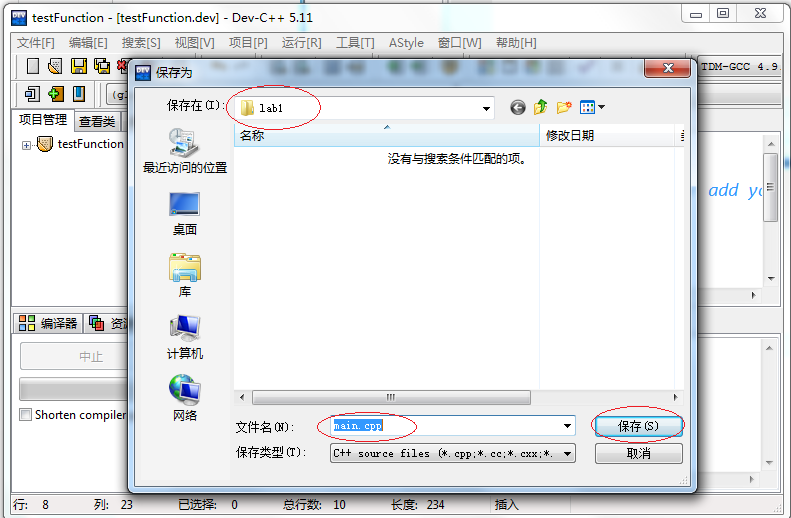


图6 保存文件

虽然可以采用默认文件名（main.c或main.cpp），建议自己根据实验内容或其它需求取其它的名字，点击保存按钮。

此时D:\CProgram\lab1目录的内容如图7所示，一个是我们创建的工程文件，另一个该工程中的C程序主函数所在的文件。

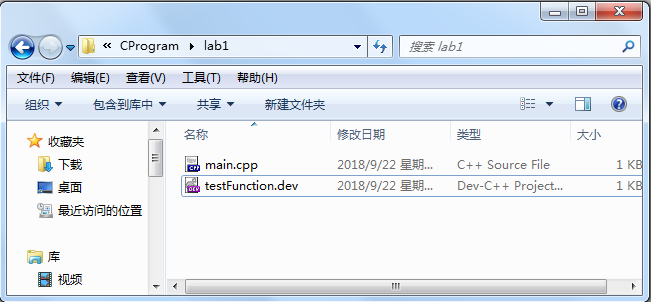


图7 D:\CProgram\lab1中的内容

需要注意的是，每一个工程中的主函数所在的文件（简称主文件），根据你所选择的项目类别均默认命名为main.c（C项目）或main.cpp（C++项目），如果将所有的项目保存到同一个目录（文件夹）中，其所对应的主文件会导致文件重名。因此，应该根据需要对主文件进行命名。

**2、输入编辑源程序**

例如我们要在Dev-Cpp中计算，并输出计算结果，程序大致如图8所示。

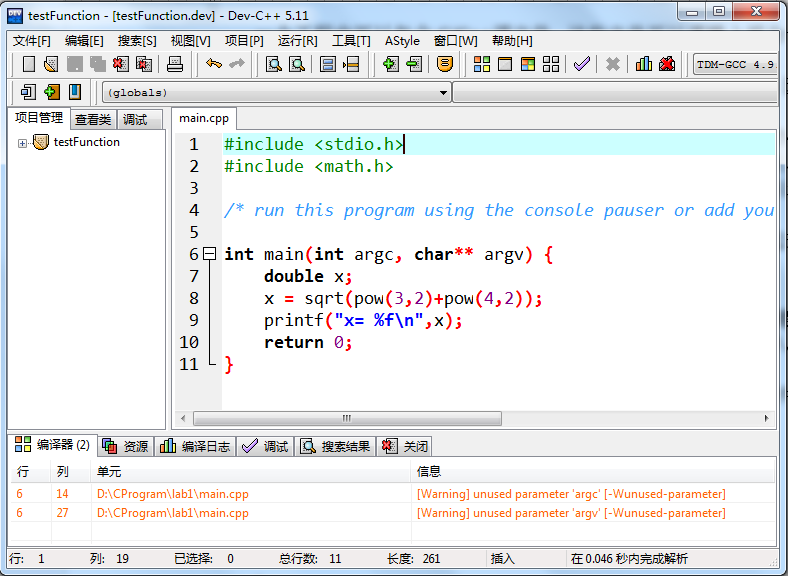


图8 输入编辑C源程序

**3、编译执行程序**

保存上述输入的源文件后，菜单选择“运行/编译执行”，或按快捷键F11，可编译执行你所编辑的源程序，如图9所示。

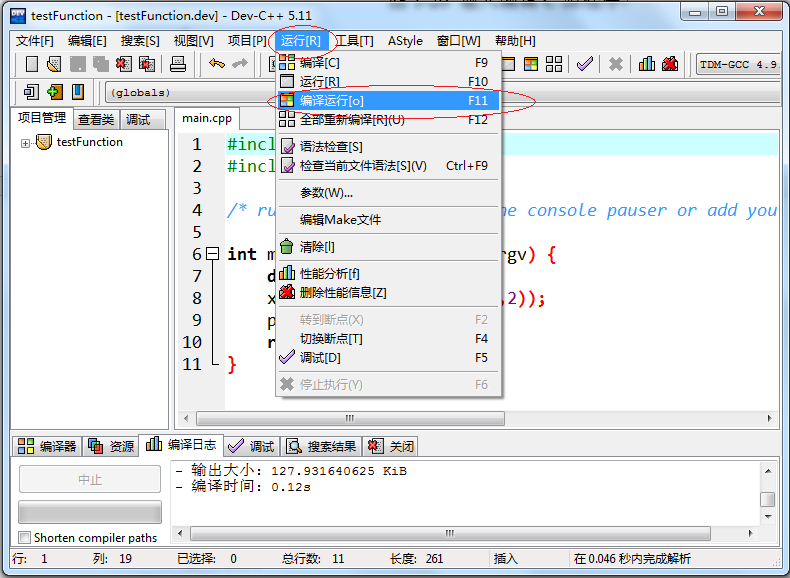


图9 编译执行源程序

如果程序没有语法错误正常执行，执行结果如图10所示。

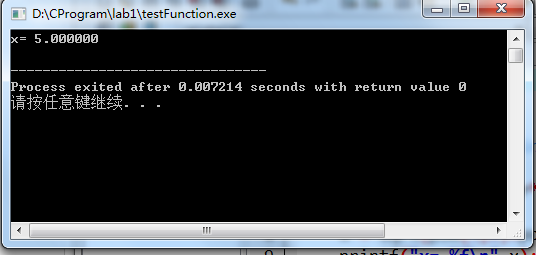


图10 程序执行结果

这时，工作目录D:\CProgram\lab1中的内容如图11所示。

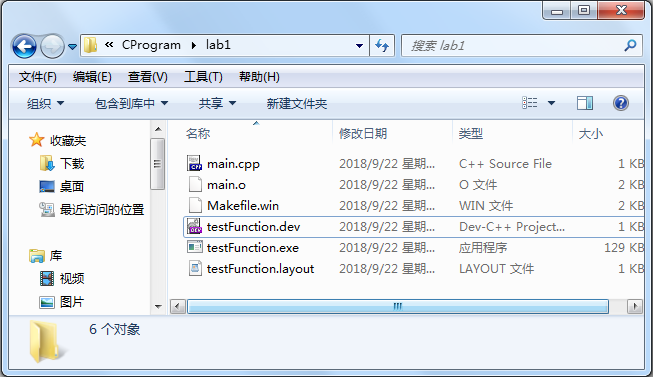


图11 程序编译后工作目录D:\CProgram\lab1中的内容

**注：如果“编译运行”输出的不是理想的结果，可能是同一个工程中有些源文件做了修改后没有及时编译，选择“全部重新编译”，将该工程中的所有文件重新编译一遍，然后选择“运行”。**

**二、 向一个工程添加C源代码文件**

采用工程的目的，就是为了方便管理同一个工程中包含的多个源代码文件。在实验过程中，建议将头文件、函数等不同类型的代码分开，分别存放到不同的源文件中，例如在头文件中对常量、全局变量、结构、函数、类等进行声明，函数与类的实现存放到其它文件中。

可以通过菜单“文件/新建/源代码”，或通过点击“新建按钮/源代码”向一个已存在的工程中添加源文件，如图12和图13所示。

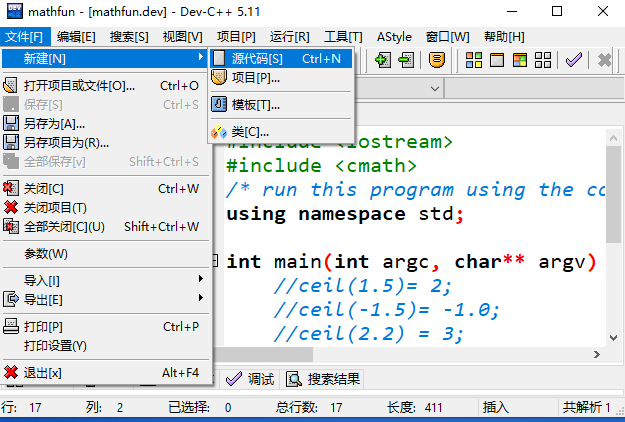


图12 通过菜单向工程中添加源代码文件

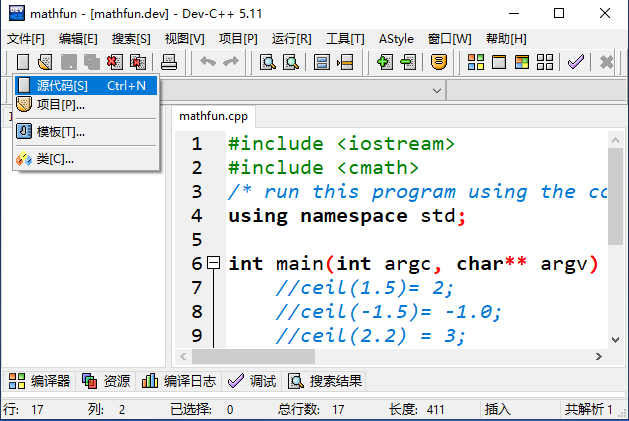
****

图13 通过按钮向工程中添加源代码文件

在随后出现的Confirm对话框中选择“Yes”，如图14所示。

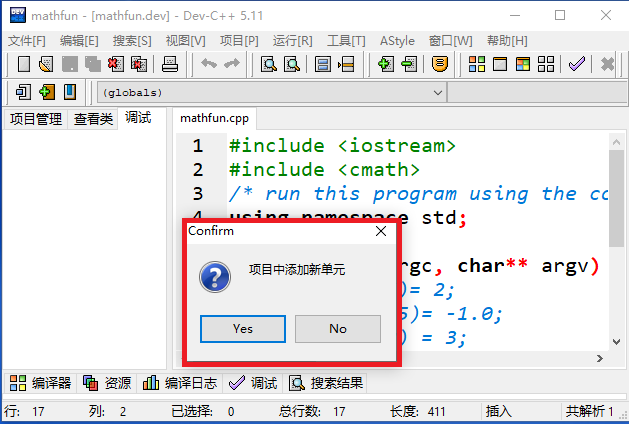


图14 向工程中添加源代码文件的Confirm对话框

然后，会新出现一个名字为“未命名”的文本框，如图15所示。你可以在其中输入C程序，输入结束后保存。

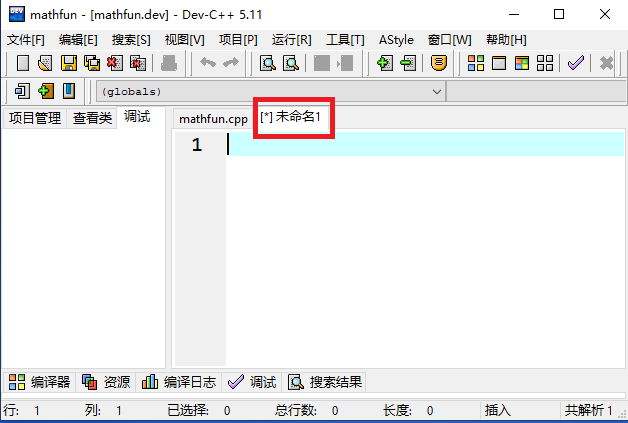


图15 向工程中添加新的源代码文件

**三、 打开已创建的工程**

如果以前已经创建了一个工程，需要重新打开装入到IDE中，在原来的基础上继续调试程序。可以通过菜单“文件/打开项目或文件”，或通过打开图标完成。如图13所示。

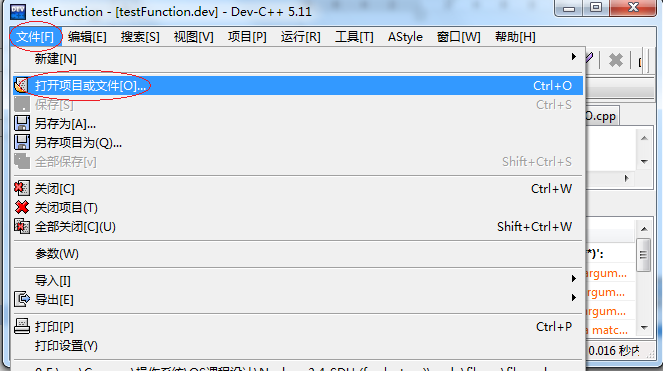


图13 打开已经存在的工程

几点注记：

**注1：关于工程（Project，项目）**

一个工程中可以包含多个C/C++源文件，这些文件逻辑上可以视为一个文件，因此这些文件只有一个程序入口，**即同一个工程中的所有文件，只有一个文件中可以有主函数main()；**

由于一个工程中包含的C/C++文件逻辑上可以视为一个文件，因此这些文件中定义的函数可以互相调用。

这种文件组织结构的优点之一就是方便一个大项目中的多人协作。

例如一个公司某部门的多个人员同时在开发一个大的工程，每人负责一个模块，可以将自己开发的程序模块保存在一个独立的文件中，最后将这些文件包含到同一个工程中，这些文件中的函数可以互相调用。

即使你自己编写的规模比较大的源程序，也建议将函数和类等分开存放到不同的源文件中，例如在头文件中对常量、全局变量、结构、函数、类等进行声明，函数与类的实现存放到其它文件中。

特别地，主函数main()中可以将每人编写的函数有机地组织起来，完成一个工程的执行。

**注2：工作空间（workspace）**

一般的IDE都有一个工作空间workspace，用于保存你所创建的工程文件、C/C++程序文件，及编译链接生成的C/C++程序对应的可执行文件，以及一些其它的辅助文件，如配置文件等。

Dev-Cpp没有显式地使用工作空间的概念，但保存工程文件的目录，如D:\CProgram\lab1，体现了工作空间的概念；

一个workspace可以包存储多个工程（Project）；

Dev-Cpp工程（项目）与C/C++源文件从属关系如图1-20所示。

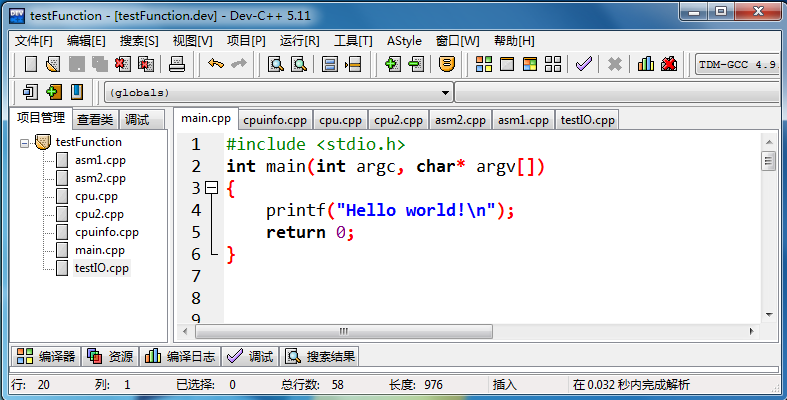


图1-20 工程（项目）与C/C++源文件从属关系

Code-Blocks中工作空间（Workspace）、工程、C/C++源文件从属关系如图1-21所示。

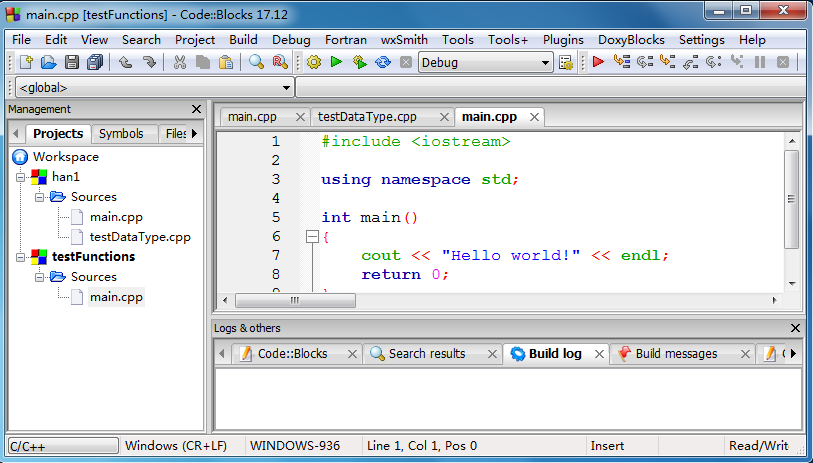


图1-21 Code-Blocks中工作空间、工程、C/C++源文件从属关系

# 附录4 练习题

下列题目作为同学自行练习之用，不要求提交实验报告。

1、输入系数a,b,c，求解一元二次方程，系数a不允许为0。

2、将金额转换成大写（限定金额5位十进制数）；

如输入32456，输出：叁万贰仟肆佰伍拾六

4、查看浮点数的机器数；

输入下列数值，输出相应数值的机器数的十六进制形式，根据IEEE 754标准理解浮点数的机内存储格式；

如：-0.0, +0.0, +1.0,-1.0,

注：对于单精度浮点数，的阶码是128；

5、测试你的CPU采用的是大端存储模式还是小端存储模式？

6、键盘输入金额数（整数，以圆为单位），按人民币面额划分，显示支付该金额的各种面额的人民币数量。

如输入587

输出100圆面额的5张

50面额的1张

20圆面额的1张

10圆面额的1张

5圆面额的1张

1圆面额的2张

7、计算两个数的最小公倍数，最大公约数；

8、利用筛法求1~100内的所有素数；

9、函数实现。给定两个字符串a与b，判断a是否是b的子串，如果是，返回a在b中的开始位置（如果a在b中多次出现，返回第一次出现的开始位置），否则，返回-1；

所谓子串指的是：设a、b是两个字符串，其长度分别是la与lb，且如果存在一个正整数i，使得 其中，；则称a是b的一个子串；

10、给定一个字符串a及正整数i，j，其中，将字符串a在i与j之间的字符所构成的串称为a的一个子串；

编写函数求a在正整数i与j之间的一个子串，根据a，i，j各种情况的取值，函数返回一个a的子串，或返回NULL；

11、判断一个数是否是回文数。

回文数：一个数从左往右读与从右往左读数字是相同的；898, 1221, 156651

12．利用递归编程将一个十进制数倒叙输出。

如输入：1234567，输出：7654321

13、函数实现。分别利用递推法与递归法求等差序列第n项的值。

如果一个等差数列的首项为 ，公差为d，该等差数列第n项的表达式为：，或 ；

在main()函数中调用你所编写函数，输出相应的结果。

如：a1=3，d=5，n=10；

14、函数实现。分别利用递推法与递归法求等比序列第n项的值。

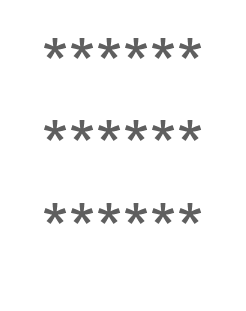
如果一个等比数列的首项为 ，公比为q，该等比数列第n项的表达式为：；

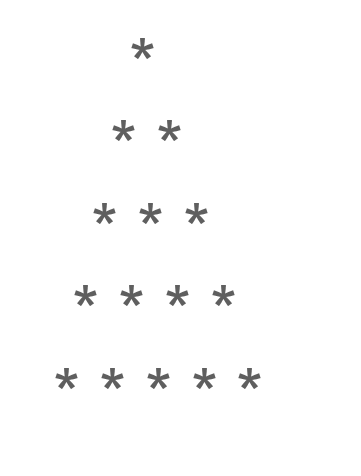
在main()函数中调用你所编写函数，输出相应的结果。

如：a1=6，q=4，n=10；

15、利用递归算法编程解决汉诺塔问题。

16、利用多重循环输出下列图形：





图形1 图形2

17、魔方阵，古代又称“纵横图”，是指组成元素为自然数1、2…n2的平方的n×n的方阵，其中每个元素值都不相等，且每行、每列以及主、副对角线上各n个元素之和都相等。

如：3X3魔方阵

8   1   6

3   5   7

4   9   2

奇阶魔方阵的排列方法：

　　⑴将1放在第一行中间一列；

　　⑵从2开始直到n×n止各数依次按下列规则存放；每一个数存放的行比前一个数的行数减1，列数加1；

　　⑶如果上一个数的行数为1，则下一个数的行数为n（指最下一行）；

　　⑷当上一个数的列数为n时，下一个数的列数应为1，行数减去1；

　　⑸如果按上面规则确定的位置上已有数，或上一个数是第一行第n列时，则把下一个数放在上一个数的下面。

编程实现nxn魔方阵的函数，并测试其正确性。

18、利用链表表示一个多项式，如：多项式，将系数 及对应的x的幂次分别存储到6个结点中；

给出两个多项式，将两个多项式的系数及对应的幂次存在到两个链表中，若系数为0，相应的结点不应存在。

实现两个多项式的加法运算及减法运算，运算结果仍然以链表方式表示；

注意多项式加减运算后，某项的系数可能为0，应将系数为0的结点删除。

结点可定义如下：

struct NODE

{ int coeff; //系数

int power; //幂次

struct NODE \*next; //指向下一个节点

}

19、教材课后习题。