**《高级语言程序设计》课后作业**

# 第5章 函数与变量

院系&专业：\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 学生姓名：\_\_\_\_\_\_\_\_\_\_\_\_ 学号：\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 填空题

**重复出现的相同/相似计算片段，可设法抽取**

1. 程序中需要自定义函数通常有两种情况：1、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**共同性的东西定义为函数。可以缩短程序，提高可读性和易修改性。**

**长计算过程中有逻辑独立性的片段，即使只出现一次也可定义为函数，以分解复杂性。**

1. 一个函数定义包括\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_两部分。

函数体

函数头部

1. 请指出如下函数定义（省略号表示一系列语句）中依次用下划线标出的各部分的含义：

double calculate (double r) ｛ … ｝

函数名

返回值类型

参数表

函数体

①\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ②\_\_\_\_\_\_\_\_ ③\_\_\_\_\_\_\_\_\_\_\_\_④\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. 当一个函数具有返回值时，则通常函数体内至少应该有一个\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_语句。

**void**

**return**

1. 当所定义的一个函数不需要返回值时，则在函数头部写返回值类型的位置写上\_\_\_\_\_\_\_\_\_\_\_\_。
2. 每个变量定义都有一个确定的作用范围，这个范围称为该变量定义的\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

变量定义的位置

作用域

1. 变量的作用域是由\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_确定的。
2. 一个变量在程序执行中从建立到销毁的存在时期称为这个变量的\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

生存期（存在期）

1. 在复合语句中定义的变量，其作用域是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_，根据它们的作用域这一特点而把它们称为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_，这类变量的存在期是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_，它们在内存中的建立与销毁都是由系统自动执行的，由此也把它们称为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

函数调用时先计算实参表达式的值，把值复制给

该变量定义所在的复合语句

该复合语句的一次执行

自动变量

局部变量

1. C/C++ 函数的值参数机制指的是：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

对应的形参，然后执行函数体。函数内部对形参的操作与实参无关。

1. C++ 程序中的函数以 引用 形式作参数时，与值参数的主要区别是：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

在函数中对形参的操作就是对实参的操作，对形参数值的修改会反映到实参。

1. 函数的“递归调用”是指\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

函数调用自身

1. 使用条件表达式补充完整如下用于计算整数的绝对值的函数：

**k >= 0? k : -k**

int absvalue(int k) { return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; }

1. 补充完整如下用于判断一个年份是否为闰年的函数：

**(year%4 == 0 && year %100 !=0) || year %400 == 0**

int isleapyear (int year) { return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; }

1. 在函数之外定义的变量称为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

**程序整个执行期间**

从变量的定义处开始到本程序文件的结尾

外部变量（全局变量）

1. 外部变量的作用域是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_，存在期是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
2. 静态局部变量的特点是：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

一次初始化

全局存在期

局部作用域

1. 在某函数中定义一个int 类型的静态局部变量 seed，则写成：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

函数原型声明

**static int seed;**

1. 如果在程序中想把自定义的函数写在 main() 函数下方，则需要在main() 之前写上\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. C/C++ 程序中，第一个非空白字符是 # 的行是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_命令行。

预处理

1. 在C++ 系统中，标准头文件中的函数、类、对象和类模板都是在名字空间 std 中定义的，为了在程序中方便地使用这些程序对象，就需要在程序前面部分写上这条语句：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**using namespace std;**

1/2

## 简答题

1、局部变量、全局变量和静态局部变量在初始化方面有何不同（需按照有无初始化语句讨论）？

局部变量是在每次执行所在的复合语句时就会进行一次初始化。如果该复合语句多次执行，每次执行时就会对内部的局部变量进行一次初始化，即在程序执行过程中可能会有多次初始化。全局变量和静态局部变量是在程序一开始执行时就执行一次初始化，程序执行过程中不会多次初始化。

程序中如果没有对变量进行初始化的语句，则局部变量不会自动初始化，初值无法确定；而全局变量和静态局部变量会自动初始化为0。

**2、**执行下面的程序之后，屏幕输出结果是什么？请详细说明为什么是这个结果。

#include <iostream>

using namespace std;

int a = 13, b = 24;

void func(int k1, int k2){ cout << k1 + k2 << "\t" << b; }

int main() { int a = 5, b = 6; func(a, b); return 0; }

**答：11 24**

**在 main 函数中定义了局部变量 a = 5 和 b = 6，遮蔽了同名的全局变量。**

**调用 func(a, b) 时使用了这两个局部变量作为实参，其值被复制给形参 k1 和 k2。**

**在 func 函数内部计算出 k1 + k2 的值为11并输出，然后输出 全局变量 b 的值 24。**

## 编程题

1、写函数判断一个正整数n是否为质数。

int isprime(int n) { int main() { //在6-200之间验证歌德巴赫猜想

**int k;**

**for (k = 2 ; k \* k <= n; k++)**

**if (n % k == 0) return 0;**

**return 1;**

**int m, n;**

**for (m = 6; m <= 200; m += 2)**

**for (n = 3; n <= m/2; n += 2) {**

**if (isprime(n) && isprime(m - n)) {**

**cout << m << " = "<< n << " + " << m-n << endl;**

**break;**

**}**

**}**

**return 0;**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

} \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2、使用已有的isprime函数在小范围内 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

验证歌德巴赫猜想：对6到200之间的 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

各偶数找出一种质数分解，即找出两个 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

质数，使它们的和等于这个偶数。 }\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3、已知有不为零的正整数m 和 n，请按照辗转相除法用递归方式和循环方式写出求最大公约数的函数。

int gcda(int m, int n) { //递归方式 int gcdb(int m, int n) { //循环方式

**int r;**

**for (r = m % n; r != 0; r = m % n) {**

**m = n; n = r;**

**}**

**return n;**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**if ( m % n == 0)**

**return n;**

**else**

**return gcda(n, m % n);**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

} }

**return m%n == 0 ? n : gcda(n, m % n);**

4、菲波那契数列的定义是：F1 = 1, F2 = 1, …… Fn = Fn−1 + Fn−2 (n>2)；写两个函数，分别用递归方式和循环方式计算菲波那契数列的第 n 项。 （如果纸面空间不够请自写）

int fib(int n){ //递归方式 int fibloop(int n) { //循环方式

**int a = 1, b = 1, tmp;**

**for (int k = 2; k < n; k++) {**

**tmp = b; b = b + a; a = tmp;**

**}**

**return b;**

**｝**

**return n < 2 ? 1 : fib(n-1) + fib(n-2);**

2/2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_