实验UNIT 03　　函数

《程序设计》课程组

第3讲上机练习

实验目的：

1. 函数的定义和调用
2. 练习重载函数的使用
3. 练习使用系统函数
4. 进一步熟练debug功能：step into的使用

第3讲上机练习

实验任务：

1. 课堂练习：函数的定义和调用、重载函数的使用、默认形参值
2. 练习2：函数的编程练习

第3讲上机练习

- 实验步骤提示：

1. 为每个题目建立一个新的控制台项目文件；
2. 向其中提交一个C++源文件；
3. 录入代码，检查是否有错误？有则改之；
4. 选择菜单“生成解决方案”编译源程序；
5. 执行程序，观察输出结果是否正确观察输出结果是否正确，如果有错误，可以执行第6步；
6. 使用debug功能：单步执行（step into、step over，注意观察两种单步的区别）、断点执行、观察变量值和输出结果是否正确？

现在开始课堂练习！

练习内容：函数的定义和使用，函数重载

默认形参值等

练习1：函数定义和使用

例子：计算5 的阶乘，请将阶乘定义为独立的函数。

练习1：函数的定义和使用

例子：计算5 的阶乘

```
#include <iostream>
using namespace std;
int fact(int);
int main()
{
        int j = fact(5);  //j等于120 ，也就是5的阶乘
        cout << "5! is " << j << endl;
        return 0;
}
int fact( int val )
{
        int rec = 1;
        while ( val > 1 )
                rec *= val--;
        return rec;
}
```

（1）运行此程序，观察程序运行结果

（2）使用单步运行模式，跟踪到fact函数内部，观察形参val的分配和释放过程，以及形参取值的变化！

练习1：函数的定义和使用

例子：计算5 的阶乘

```
#include <iostream>
using namespace std;
int fact(int);
int main()
{
        int j = fact(5);  //j等于120，也就是5的阶乘
        cout << "5! is " << j << endl;
        return 0;
}
int fact( int val )
{
        int rec = 1;
        while ( val > 1 )
                rec *= val--;
        return rec;
}
```

分别将此函数调用更换为以下形式，观察编译器是否出现报错信息：

```
fact("hello");
fact();
fact(42,10,0);
fact(3.14);
```

练习2：无返回值函数和引用参数

请仔细阅读以下函数，自己编写main函数，定义两个整数变量a 和b，使用cin给变量a和b输入数值，然后分别调用

（1）swap(a,a)

（2）swap(a,b)

请用单步调试功能，观察两种调用形式的执行过程.

```
void swap(int &v1, int &v2)
{
    if(v1 == v2)
        return;
    int temp = v2;
    v2 = v1;
    v1 = temp;
    //void函数，此处隐式执行return语句，无需显式return语句
}
```

练习3：函数重载

函数重载是指同一个作用域内的几个函数名字相同但形参列表不同。具有相似功能的不同函数使用同一函数名，此时这些同名函数的参数类型、参数个数、返回值、函数功能可以不同。

编译系统将根据函数参数的类型和个数来判断使用哪一个函数。体现了C++对多态性的支持。

一个小疑问：为什么C++不支持仅仅返回类型不一样而形参一样的函数重载?

```
// 参数类型不同的函数重载
#include <iostream>
using namespace std;
int abs(int x)                                  //形参为整型
{    return x>0?x:-x;    }
double abs(double x)                            //形参为双精度型
{    return x>0?x:-x;    }
long abs(long x)                                //形参为长整型
{    return x>0?x:-x;    }
int main()
{
  int x1=4;
  double x2=5.5;
  long x3=6L;
  cout<<"|x1|="<<abs(x1)<<endl;
  cout<<"|x2|="<<abs(x2)<<endl;
  cout<<"|x3|="<<abs(x3)<<endl;
  return 0;
}
```

观察此程序的运行结果，使用调试功能，观察main函数的三次abs函数调用分别调用的是哪个函数?

```
// 参数个数不同的函数重载
#include <iostream>
using namespace std;
int Add(int x，int y)                          //2个形参
{   return x+y;   }
int Add(int x，int y, int z)                   //3个形参
{   return x+y+z;   }
int main()
{
  int a=3,b=4,c=5;
  cout<<a<<"+"<<b<<"="<<Add(a,b)<<endl;
  cout<<a<<“+”<<b<<“+”<<c<<“=”<<Add(a,b,c)<<endl;
  return 0;
}
```

观察此程序的运行结果，使用调试功能，观察main函数的两次Add函数调用分别调用的是哪个函数?

练习4：默认形参

- C++允许在定义函数时给其中的某个或某些形参指定缺省值(定义函数参数时赋初值)，这样，当发生函数调用时，若指定实参值则用该值；如果省略了对应位置上的实参的值时，则在执行被调函数时，以该形参的缺省值进行运算。
- 引入缺省参数的目是为了让编程简单，让编译系统做更多的检查错误工作，同时增强了函数的可重用性。

练习4：默认形参

```
// 带默认参数的函数
#include <iostream>
using namespace std;
void sum(int num=10)     //带默认参数函数的定义
{       int i,s=0;
        for(i=1;i<=num;i++)
                s=s+i;
        cout<<"sum is:"<<s<<endl;
}
int main()
{
  sum(100);
  sum();
  return 0;
}
```

观察此程序的运行结果，使用调试功能，观察main函数的两次sum函数调用时形参num的取值是多少？

练习5：默认形参

（1）所有默认参数必须放在参数表的最后。

例如：

```
int Fun(int i, int j=5, int k);     //对还是错？为什么？
```

练习5：默认形参

（2）当程序中既有函数声明又有函数定义时，则定义函数时不允许再定义参数的默认值，即使指定的默认值完全相同也不行。也就是说，缺省值必须出现在函数声明中。

如程序中只有函数的定义，而没有声明函数，则默认参数才可出现在函数定义中。

（3）默认参数的声明必须出现在函数调用之前。

练习5：默认形参

（4）如果默认参数在调用时，又得到了实参，则实参值优先。

（5）函数调用时，如果某个参数省略，则其后的参数都应该省略而采用默认值。

例如：

```
int Init(int x=5, int y=10);
Init( ,20);     //对还是错？为什么？
```

练习6：内联函数

- 内联函数的函数体内不能含有复杂的结构控制语句，如：switch和while 等，否则编译系统将该函数视为普通函数。
- 递归函数不能定义为内联函数
- 内联函数一般适合于只有1~5条语句的小函数，对一个含有很多语句的大函数，没有必要使用内联函数。
- 内联函数的定义必须出现在内联函数 的第一次被调用之前。内联函数只能 先定义后使用。

练习6：内联函数

```
//观察此函数的运行结果，
//用内联函数实现求两个整数中最大数。
#include <iostream>
using namespace std;
inline int max(int x,int y)     //内联函数
{
    return x>y?x:y;
}
int main()
{
    int a,b,c;
    cout<<"Input two data:";
    cin>>a>>b;
    cout<<"The max is:"<<max(a,b)<<endl;
    return 0;
}
```

练习6：内联函数

```
// 观察此函数的运行结果，内联函数使用示例。
#include <iostream>
using namespace std;
inline double Circle(double r)      //内联函数
{
    return 3.1416*r*r;
}
int main()
{
    for(int i=1; i<=3;i++)
    cout<<"r="<<i<<"area="<<Circle(i)<<endl;
    return 0;
}
```

本次课堂练习结束！

任务2!

练习内容：函数的编程练习

第3讲 编程练习

完成以下教材习题编程：

3-7、完成函数，参数为两个unsigned short int型数，返回值为第一个参数除以第二个参数的结果，数据类型为short int；如果第二个参数为0，则返回值为-1。在主程序中实现输入输出。注意观察函数的参数传入传出、返回值

3-13、用递归的方法编写函数 求Fiboonacci级数，公式为：

$$F_n = F_{n-1} + F_{n-2} (n>2),\ F_1 = F_2 = 1$$

注意观察递归调用的过程。

3-15、编写递归函数getPower计算x^y,在同一个程序中针对整型和实型实现两个重载函数：

int getPower(int x, int y); //整型版本，， 当y<0时，返回0

double getPower(double x, int y); //实型版本

在主程序中实现输入输出，分别输入一个整数a和一个实数b作为底数，再输入一个整数m作为指数，输出a^m和b^m。

另外请读者思考，如果在调用getPower函数计算a^m时希望得到一个实型结果（实型结果表示范围更大，而且可以准确表示m<0时的结果），该如何调用？注意观察递归调用的过程。

第3讲 编程练习

完成以下教材习题编程：

3-10、编写函数求两个整数的最大公约数和最小公倍数。要求两个整数 以及结果在 主函数中输入输出。

3-9、编写函数判别一个整数是否是质数？要求在主函数中输入输出。

本 讲 结 束

有问题吗?