## 第3章 函数

**习题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

#include<iostream>

using namespace std;

int main()

{

    short int divide(unsigned short int x, unsigned short int y);

    unsigned short int x, y;

    cout<<"Please enter two integers:";

    cin>>x>>y;

    cout<<divide(x,y)<<endl;

    return 0;

}

short int divide(unsigned short int x, unsigned short int y)

{

    if(y!=0)

        return x/y;

    else

        return -1;

}

**习题3-8**

//编写函数把华氏温度转换为摄氏温度，公式为

//            C=5/9\*(F-32)

//在主程序中提示用户输入一个华氏温度，转化后输出相应的摄氏温度

#include<iostream>

using namespace std;

int main()

{

    float FahrenheitToCentigrade(float FahrenDegree);

    float fa;

    cout<<"Please enter a degree Fahrenheit:";

    cin>>fa;

    cout<<"the degree Centigrade is "<<FahrenheitToCentigrade(fa)<<endl;

    return 0;

}

float FahrenheitToCentigrade(float FahrenDegree)

{

    float C;

    C = (FahrenDegree-32)\*5/9;

    return C;

}

**习题3-9**

//编写函数判别一个数是否质数，在主程序中实现输入输出

#include<iostream>

using namespace std;

int main()

{

    bool isPrime(int num);

    int num;

    cout<<"Please enter an integer:";

    cin>>num;

    if(isPrime(num))

        cout<<num<<" is prime."<<endl;

    else

        cout<<num<<" is not prime."<<endl;

    return 0;

}

bool isPrime(int num)

{

    bool flag = true;

    for(int i=2;i<num;i++){

        if(num%i==0)

            flag = false;

    }

    return flag;

}

**习题3-10**

//编写函数求两个整数的最大公约数和最小公倍数

#include<iostream>

using namespace std;

int greatestCommonDivisor(int a, int b);

int leastCommonMultiple(int a, int b);

int main()

{

    int a,b;

    cout<<"Please enter two integers:";

    cin>>a>>b;

    cout<<"the greatest common divisor with "<<a<<" and "<<b<<" is "<<greatestCommonDivisor(a,b)<<endl;

    cout<<"the least common multiple with "<<a<<" and "<<b<<" is "<<leastCommonMultiple(a,b)<<endl;

    return 0;

}

int greatestCommonDivisor(int a, int b)

{

    if(b==0)

        return 0;

    int c = a%b;

    while (c!=0)

    {

        a = b;

        b = c;

        c = a%b;

    }

    return b;

}

int leastCommonMultiple(int a, int b)

{

    int c;

    c = greatestCommonDivisor(a,b);

    if(c!=0)

        return a\*b/c;

    else

        return 0;

}

**习题3-12**

//在主程序中提示输入整数n，编写函数

//用递归的方法求1+2+...+n的值

#include<iostream>

using namespace std;

int sum1\_n(int n);

int main()

{

    int n;

    cout<<"Please enter an integer:";

    cin>>n;

    if(n==1)

        cout<<"1 = "<<sum1\_n(n)<<endl;

    else if(n==2)

        cout<<"1+2 = "<<sum1\_n(n)<<endl;

    else

        cout<<"1+2+...+"<<n<<" = "<<sum1\_n(n)<<endl;

    return 0;

}

int sum1\_n(int n)

{

    if(n==1)

        return 1;

    else

        return sum1\_n(n-1)+n;

}

**习题3-13**

//用递归的方法编写函数求Fibonacci级数，公式为

//Fn=Fn-1+Fn-2(n>2), F1=F2=1

//观察递归调用的过程

#include<iostream>

using namespace std;

int Fibonacci(int n);

int main()

{

    int i,n;

    cout<<"Please enter item n:";

    cin>>n;

    cout<<"Fibnoacci:";

    for(i=1;i<=n;i++)

        if(i<n)

            cout<<Fibonacci(i)<<",";

        else

            cout<<Fibonacci(i)<<endl;

    return 0;

}

int Fibonacci(int n)

{

    if(n==1)

        return 1;

    else if(n==2)

        return 1;

    else

        return Fibonacci(n-1)+Fibonacci(n-2);

}

**习题3-14**

//用递归的方法编写函数求n阶勒让德多项式的值

//在主程序中实现输入输出，递归公式为

//pn(x)=1 (n=0)

//pn(x)=x (n=1)

//pn(x)=[(2n-1)x\*pn-1(x) - (n-1)pn-2(x)]/n (n>1)

#include<iostream>

using namespace std;

double legendre(double x, int n);

int main()

{

    double x;

    int n;

    cout<<"Please enter n:";

    cin>>n;

    cout<<"Please enter x:";

    cin>>x;

    cout<<"Legendre("<<x<<", "<<n<<")="<<legendre(x, n)<<endl;

    return 0;

}

double legendre(double x, int n)

{

    if(n==0)

        return 1;

    else if(n==1)

        return x;

    else

        return ((2\*n-1)\*x\*legendre(x, n-1) - (n-1)\*legendre(x,n-2))/n;

}

**习题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的函数

//int getPower(int x, int y);       //整型形式，当y<0时，返回0

//double getPower(double x, int y); //实型形式

//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？

#include<iostream>

using namespace std;

int getPower(int x, int y);       //整型形式，当y<0时，返回0

double getPower(double x, int y); //实型形式

int main()

{

    int a;

    double b;

    int m;

    cout<<"Please enter an integer a:";

    cin>>a;

    cout<<"Please enter a real b:";

    cin>>b;

    cout<<"Please enter an integer m:";

    cin>>m;

    if(m<0)

        cout<<a<<"^"<<m<<" = "<<1/(double)getPower(a,-m)<<endl;

    else

        cout<<a<<"^"<<m<<" = "<<getPower(a,m)<<endl;

    if(m<0)

        cout<<b<<"^"<<m<<" = "<<1/getPower(b,-m)<<endl;

    else

        cout<<b<<"^"<<m<<" = "<<getPower(b,m)<<endl;

    return 0;

}

int getPower(int x, int y)

{

    if(y==0)

        return 1;

    else

        return x\*getPower(x, y-1);

}

double getPower(double x, int y)

{

    if(y==0)

        return 1;

    else

        return x\*getPower(x, y-1);

}