

**C/C++实验报告**

**实验一：C/C++程序运行环境VS和程序运行方法**

一、实验目的

认识Visual Studio，熟悉Visual Studio 运行环境，掌握运行一个C++程序的方法。

二、实验内容

1、编程计算圆周率，看看你能将圆周率精确到几位小数。参考教材中的级数计算方法。

2、比较参考资料中的圆周率公式的收敛速度。至少对比3个公式。

3、第一个π和1/π的小数点后首一百万位利用了[古腾堡计划](http://zh.wikipedia.org/wiki/古腾堡计划)。最新纪录是[2002年](http://zh.wikipedia.org/wiki/2002年)九月得出的1,241,100,000,000个小数位，由拥有1[TB](http://zh.wikipedia.org/wiki/TB)主内存的64-node[日立](http://zh.wikipedia.org/wiki/日立)[超级计算机](http://zh.wikipedia.org/wiki/超級電腦)，以每秒200亿运算速度得出，比旧纪录多算出一倍（206亿小数位）。此纪录由以下[梅钦类公式](http://zh.wikipedia.org/w/index.php?title=梅欽類公式&action=edit&redlink=1)得出：

(K. Takano, 1982年)



(F. C. W. Störmer, 1896年)



这么多的小数位没什么实用价值，只用以测试[超级计算机](http://zh.wikipedia.org/wiki/超級電腦)。

[1996年](http://zh.wikipedia.org/wiki/1996年)，David H. Bailey、Peter Borwein及[西蒙·普劳夫](http://zh.wikipedia.org/wiki/西蒙·普勞夫)发现了π的其中一个无穷级数：



以上述公式可以计算π的第n个二进制或十六进制小数，而不需先计算首n-1个小数位。此类π算法称为[贝利－波尔温－普劳夫公式](http://zh.wikipedia.org/wiki/贝利－波尔温－普劳夫公式)。请参考[Bailey's website](http://www.nersc.gov/~dhbailey/) 上相关[程式](http://zh.wikipedia.org/wiki/程式)。

三、源程序

PI\_head源程序：

#include<iostream>

#include<math.h>

#include<windows.h>

#include <iomanip>

using namespace std;

//(K. Takano, 1982年)提出公式

double Takano();

//(F. C. W. Stormer, 1896年)

double Stormer();

//1996年，David H. Bailey、Peter Borwein及西蒙,普劳夫发现了π的其中一个无穷级数

double David();

PI\_Count源程序：

#include"PI\_head.h"

int main()

{

double pi\_1=Takano();

double pi\_2=Stormer();

double pi\_3=David();

cout<<"Takano->pi为："<<setprecision(10)<<pi\_1<<endl;

cout<<"Stormer->pi为："<<setprecision(10)<<pi\_2<<endl;

cout<<"David->pi为："<<setprecision(10)<<pi\_3<<endl;

system("pause");

return 0;

}

David源程序：

#include"PI\_head.h"

//1996年，David H. Bailey、Peter Borwein及西蒙,普劳夫发现了π的其中一个无穷级数

double David()

{

double t=0;

double pi=0;

for(int i=0;i<=10000;i++)

{

pi+=1.0/pow(16,i)\*((4.0/(8\*i+1)-2.0/(8\*i+4)-1.0/(8\*i+5)-1.0/(8\*i+6)));

}

return pi;

}

Stormer源程序：  
#include"PI\_head.h"

//(F. C. W. Stormer, 1896年)

double Stormer()

{

return 4\*(44\*atan(1.0/57)+7\*atan(1.0/239)-12\*atan(1.0/682)+24\*atan(1.0/12943));

}

Takano源程序：

#include"PI\_head.h"

//(K. Takano, 1982年)提出公式

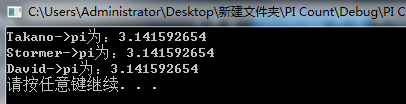
double Takano()

{

return 4\*(12\*atan(1.0/49)+32\*atan(1.0/57)-5\*atan(1.0/239)+12\*atan(1.0/110443));

}

四、实验结果



**实验二：输入数将其转换为中文大写金额形式**

一、实验目的

了解C++与C语言的异同，掌握多文件程序编译方法。

二、实验内容

编写控制台模式的程序，接收用户输入的整数，然后将其转换为中文大写金额的形式。如：输入： 1024 输出： 壹仟零贰拾肆

要求如下：

1）输入输出必须使用cin、cout。

2）要求每个源文件仅包含单一功能模块。

3) 先设计测试用的例子，并在test.cpp中实现，然后实现代码。

4) 在实现上述要求的基础上，修改代码，使其可以转换小数形式的数据，例如输入3.12时，转换为叁元壹角贰分。

5) 在实现上述要求的基础上，修改代码，使其在输入错误时，给出错误提示，并说明错误出现的位置。

三、源程序

Head源程序：

#include<iostream>

#include<string>

#include<algorithm>

#include<windows.h>

using namespace std;

void Translation(int number);

void Input(int &number);

Input源程序：

#include"Head.h"

void Input(int &number)

{

cout<<"请输入一个正数:";

cin>>number;

while(number<0)

{

cout<<"输入数据不合法,请重新输入一个正数:";

cin>>number;

}

}

Translation源程序：

#include"Head.h"

void Translation(int number)

{

string str="";

int a[20]={-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1};

int i=0;

while(number!=0)

{

a[i]=number%10;i++;

number=number/10;

}

i=0;

while(a[i]!=-1)

{

switch (a[i])

{

case 0: str="零"+str;break;

case 1: str="壹"+str;break;

case 2: str="贰"+str;break;

case 3: str="叁"+str;break;

case 4: str="肆"+str;break;

case 5: str="伍"+str;break;

case 6: str="陆"+str;break;

case 7: str="柒"+str;break;

case 8: str="捌"+str;break;

case 9: str="玖"+str;break;

}

i++;

}

cout<<"翻译成大写的数字为:"<<str<<endl;

}

Main源程序：

#include "Head.h"

int main()

{

int number;

Input(number);

Translation(number);

system("pause");

return 0;

}

四、实验结果



**实验三：类和对象**

一、实验目的

理解面向对象的基本概念，能够对简单问题运用类和对象概念分析建模。

二、实验内容

1、 定义复数类

**实现如下功能：**

* 定义构造函数，实现多种初始化方式；
* 通过重载运算符的函数，实现复数与复数的加减乘除运算；
* 复数的模运算；
* 能按照如下格式“1+2i” “3i” “7.5”等转换为字符串输出；
* 统计程序运行期间，共创建多少个复数对象。

**其它要求：**

* 修改完善参考代码，实现复数类；
* 代码分别放在多个文件中，其中类的定义、类的成员函数定义、主函数、测试函数分别放在不同的文件中
* 正确实现类的定义，使得参考代码中的测试函数可以通过编译
* 必须定义5个以上的测试函数
* 必须包含3个以上的代码文件

三、源程序

Head源程序：

#include<iostream>

#include<cmath>

#include<string>

#include<windows.h>

using namespace std;

//定义一个类

class Complex

{

private :

double a;//实部

double b;//虚部

public:

Complex(double i=0,double j=0);//构造函数

//复数运算符重载

Complex operator + (const Complex C) const

{

Complex t;

t.a+=C.a;

t.b+=C.b;

return t;

}

Complex operator - (const Complex C) const

{

Complex t;

t.a-=C.a;

t.b-=C.b;

return t;

}

Complex operator \* (const Complex C) const

{

Complex t;

t.a\*=C.a;

t.b\*=C.b;

return t;

}

Complex operator / (const Complex C) const

{

Complex t;

t.a/=C.a;

t.b/=C.b;

return t;

}

double Length();//取模运算

void Tostring ();//字符串化

};

Complex源程序：

#include"Head.h"

Complex::Complex(double i,double j)

{

a=i;

b=j;

}

double Complex::Length()//取模运算

{

return sqrt(a\*a+b\*b);

}

void Complex::Tostring ()//字符串化

{

cout<<a<<"+"<<b<<"i"<<endl;

}

Main源程序：

#include "Head.h"

int main()

{

double a,b;

cout<<"请输入一个复数的实部：";

cin>>a;

cout<<"请输入一个复数的虚部：";

cin>>b;

double i,j;

cout<<"请输入另一个复数的实部：";

cin>>i;

cout<<"请输入另21一个复数的虚部：";

cin>>j;

Complex C1=Complex(a,b);

cout<<"输入的复数为：";

C1.Tostring();

Complex C2=Complex(i,j);

cout<<"输入的复数为：";

C2.Tostring();

cout<<"复数加法运算：";

Complex r1=C1+C2;

r1.Tostring();

cout<<"复数减法运算：";

Complex r2= C1+C2;

r2.Tostring();

cout<<"复数乘法运算：";

Complex r3= C1\*C2;

r3.Tostring();

cout<<"复数除法运算：";

Complex r4= C1/C2;

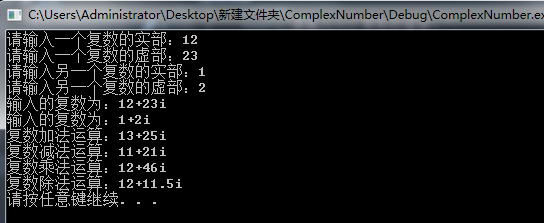
r4.Tostring();

system("pause");

return 0;

}

四、实验结果



**实验四：继承与多态**

一、实验目的

了解继承和多态的作用和实现方式；掌握使用C++的抽象类和派生类实现继承性。

二、实验内容

**1．设计一个立体图形类（CStereoShape类），并满足如下要求：**

* CStereoShape类有一个纯虚函数GetArea，能够获取立方体的表面积；
* CStereoShape类有一个纯虚函数GetVolume，能够获取立方体的体积；

**2．设计一个立方体类（CCube类），该类继承于CStereoShape类，功能如下：**

* CCube类有一个带参数的构造函数，其参数分别对应于立方体的长、宽、高。
* 用一个成员函数来实现对立方体长、宽、高的设置。
* 重载CStereoShape类的GetArea和GetVolume，分别完成立方体的表面积和体积的计算。

**3．设计一个球体类（CSphere），该类继承于CStereoShape类，功能如下：**

* CSphere类有一个带参数的构造函数，其参数对应于球体的半径。
* 用一个成员函数来实现对球体半径的设置。
* 重载CStereoShape类的GetArea和GetVolume，分别完成球体的表面积和体积的计算。

**4．实现一个main函数，在main函数中至少完成如下工作：**

* 实例化一个CCube类的对象a\_cube和CSphere类的对象c\_sphere；
* 定义一个CStereoShape类的指针p；
* 将a\_cube的长、宽和高分别设置为4、5和6；将p指向a\_cube，通过p将a\_cube的表面积和体积打印到屏幕上；
* 将c\_sphere的半径设置为7；将p指向c\_sphere，通过p将c\_sphere的表面积和体积打印到屏幕上。

构造一个链表，链表每个结点存储CStereoShape类的指针，每个指针指向CCube、CSphere或者CSphere类的对象，计算所有对象的体积之和。

三、源程序

Head源程序：

#include<iostream>

#include<cmath>

#include<windows.h>

#define pi 3.1415926

using namespace std;

class CStereoShape

{

public:

virtual double GetArea()=0;

virtual double GetVolume()=0;

};

class CCube:public CStereoShape

{

private:

double Length;

double Width;

double Height;

public:

CCube(double L,double W,double H)

{

this->Length=L;

this->Width=W;

this->Height=H;

}

void Setter(double L,double W,double H)

{

this->Length=L;

this->Width=W;

this->Height=H;

}

virtual double GetArea()

{

return 2\*(Length\*Width+Length\*Height+Width\*Height);

}

virtual double GetVolume()

{

return (Length\*Width\*Height);

}

};

class CSphere:public CStereoShape

{

private :

double Radius;

public:

CSphere(double R)

{

Radius = R;

}

void Setter(double R)

{

Radius = R;

}

virtual double GetArea()

{

return 4\*(pi\*Radius\*Radius);

}

virtual double GetVolume()

{

return (4./3)\*(pi\*Radius\*Radius\*Radius);

}

};

struct Node

{

CStereoShape\* p;

Node\* Next;

};

Main 源程序：

#include "Head.h"

int main()

{

CCube a\_cube=CCube(12.4,13.5,14.6);

CSphere c\_sphere=CSphere(12.3);

CStereoShape \*p;

a\_cube.Setter(4,5,6);

p=&a\_cube;

cout<<"===========多面体========="<<endl;

cout<<"表面积："<<(\*p).GetArea()<<endl;

cout<<"体积："<<(\*p).GetVolume()<<endl;

c\_sphere.Setter(7);

p=&c\_sphere;

cout<<"===========球体========="<<endl;

cout<<"表面积："<<p->GetArea()<<endl;

cout<<"体积："<<p->GetVolume()<<endl;

Node s1;

Node s2;

Node \*t;

s1.p=&a\_cube;

s1.Next=&s2;

s2.p=&c\_sphere;

s2.Next=NULL;

cout<<"===========链表求和========="<<endl;

double v=0;

double s=0;

t=&s1;

while (t != NULL)

{

v+=(t->p)->GetVolume();

s+=(t->p)->GetArea();

t=t->Next;

}

cout<<"表面积和："<<s<<endl;

cout<<"体积和："<<v<<endl;

system("pause");

return 0;

}

四、实验结果

