单项选择

1 对使用关键字new所开辟的动态存储空间，释放时必须使用【】。

A、free

B、create

C、delete

D、realse

2 以下对一维数组a的正确说明是（　　）

A、int a(10);

B、int n;

cin>>n;

int a[n];

C、int n=10,a[n];

D、const int n=10;

int a[n];

3 当一个类对象生命期结束时，系统自动调用\_\_\_\_\_并销毁对象。

A、析构函数 B、静态函数

C、友元函数 D、构造函数

4 一个C++程序的执行是从（ ）

A、本程序文件的第一个函数开始,到本程序main函数结束

B、本程序的main函数开始,到本程序文件的最后一个函数结束

C、本程序文件的第一个函数开始,到本程序文件的最后一个函数结束

D、本程序的main函数开始,到main函数结束

**5** 下列关于类的权限的描述错误的是【】。

A、类的对象只能访问该类的公有成员

B、一个类可以将另一个类的对象作为成员

C、普通函数不能直接访问类的公有成员，必须通过对象访问

D、类本身的成员函数只能访问自身的私有成员

6 在循环语句的循环体中使用break语句的作用是( )

A、结束开关语句 B、结束程序执行

C、结束本次循环 D、结束该层的循环

7 下面程序运行结果是( )

#include <iostream>

using namespace std;

int main()

{

int x=0,y=0;

while(x<15)

{

y++;

x+=y;

}

cout<<y<<","<<x;

return 0;

}

A、5,7 B、20,8 C、6,12 D、5,15

8 多重if~else语句嵌套使用时，寻找与else配对的if的方法是【】。

A、缩进位置相同的if B、其上最近的if

C、其上最近未匹配的if D、下面最近的if

9 如下代码，对其描述正确的是【 】。int \*p;p=new int(5);

A、动态开辟5个int元素，初值不确定 B、存在语法错误

C、动态开辟5个int元素，初值为0 D、动态开辟初值为5的一个int元素

10 一个类可包含析构函数的个数是【】。

A、1个 B、0个 C、至少一个 D、0个或多个

11 下列对构造函数的描述中，【 】是错误的。

A、函数名与类名完全相同 B、构造函数允许重载

C、构造函数允许使用缺省参数 D、返回类型应该为void

12 对于有返回值的函数，要结束函数运行必须使用语句（ ）

A、return B、main C、break D、continue

13 main()函数的返回类型是【 】。

A、float B、任意类型 C、不返回值 D、int

14 int x=-1;

do

{

x=x\*x;

} while(!x);

下列说法正确的是( )

A、是死循环 B、有语法错误 C、循环执行一次 D、循环执行两次

15 下面不正确的字符串常量是( )

A、‘rst’ B、“14’14” C、“” D、“0”

**程序设计**

[题目] 定义一个类prime，用于求在一个指定的范围内素数的个数。

例如：在[3，50]范围内的素数有14个

[编程要求]

（1）私有数据成员。

int low：存放给定整数范围的下限

int up：存放给定整数范围的上限

int count：存放满足条件的整数的个数

（2）公有成员函数

prime(int n1,int n2)构造函数，用参数n1,n2初始化下限low和上限up,count的初值为0；

void process（）：在[low，up]范围中找出素数的个数，并把值赋给count;

void print（）：输出素数的个数.

（3）在主函数中定义prime类的对象p,给定的范围为[3，50]，

通过对象p调用成员函数实现计算及输出调试。

本题要求的输出结果为：[3，50]范围内有14个素数。

#include<iostream>

#include <cmath>

#include<fstream>

using namespace std;

class prime{

int low;

int up;

int count;

public:

prime(int n1,int n2)

{low=n1;up=n2;count=0;}

void process()

{

for(int i=low;i<=up;i++)

{

for(int j=2;j<i;j++)

if(i%j==0)break;

if(j==i) count++;

}

void print()

{

cout<<'['<<low<<','<<up<<"]范围内有"<<count<<"个素数\n";

//此处将结果输出到文件"bc02.in"，请勿改动，否则影响判分

ofstream outf("bc02.in");

outf<<'['<<low<<','<<up<<"]范围内有"<<count<<"个素数\n";

outf.close();

}

};

int main()

{

prime p(3,50);

p.process();

p.print();

return 0;

}

**程序填空**

**==================================================**

/\*1 输入20个整数到一维数组，统计正整数的个数及其和#include<iostream>

using namespace std;

int main()

{

int a[20],i,sum,count;

sum=count=0；

for(i=0;i<20;i++)

cin>>a[i];

for(i=0;i<20;i++)

{

if( a[i]>0 )

{ \_\_\_\_\_\_count+=1; \_\_\_\_\_\_\_\_\_ //计数

\_\_\_\_\_\_sum+=a[i]; \_\_\_\_\_\_\_\_\_ //求和

}

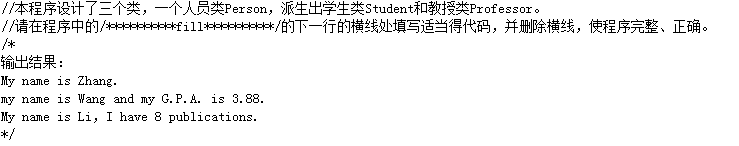
}

cout<<count<<" "<<sum<<endl;

return 0;

}

**2**



#include <iostream>

#include <cstring>

using namespace std;

class Person{

public:

\_\_\_\_ Person()\_\_\_\_\_\_\_\_\_\_\_{name=NULL;}

Person(char\* s)

{

name = new char[strlen(s)+1]; strcpy(name, s);

}

~Person()

{

if(name!=NULL) delete [] name ;

}

\_\_\_\_ virtual void\_\_\_\_\_\_\_\_Disp() // 声明虚函数

{

cout << "My name is " << name << ".\n";

}

void setName(char\* s)

{

name = new char[strlen(s)+1]; strcpy(name, s);

}

protected:

char\* name;

};

class Student : public Person{

public:

Student(char\* s, double g) \_\_:Person(s), gpa(g)\_\_{ }

void Disp()

{

cout << "my name is " << name << " and my G.P.A. is " << gpa << ".\n";

}

private:

float gpa;

};

class Professor : public Person{

public:

void setPubls(int n){publs=n; }

void Disp()

{

cout<< "My name is "<<name<<"，I have " << publs << " publications.\n";

}

private:

int publs;

};

int main()

{ \_\_\_\_\_\_ Person\* p\_\_\_\_\_\_\_\_\_;

Person x("Zhang");

p = &x; p->Disp();

Student y("Wang", 3.88);

p = &y; p->Disp();

Professor z;

z.setName("Li");

z.setPubls(8);

p = &z; p->Disp();

return 0;

}

//3 输入一个5个字符组成的字符串，将其倒序后输出.

// 注意： 先删除\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_，然后在该行填入正确代码。

#include<iostream>

using namespace std;

int main()

{

int i,j;

char st[6],ch;

cin>>st;

for(i=0,j=4;j>i; i++,j--)

{ ch=st[i];

st[i]=st[j];

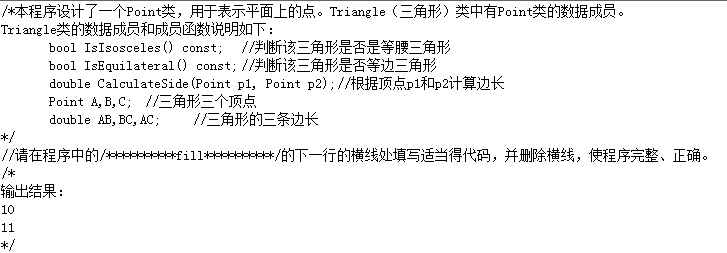
st[j]=ch;

}

cout<<st<<endl;

}

**4**



#include <iostream>

#include <cmath>

using namespace std;

#define MINNUM 0.00001

class Point

{

public:

Point(double x = 0.0, double y = 0.0): X(x), Y(y) {}

~Point() {}

double GetX() { return X; }

double GetY() { return Y; }

private:

double X,Y;

};

class Triangle

{

public:

Triangle(Point a, Point b, Point c);

~Triangle() {}

bool IsIsosceles() const; //判断该三角形是否是等腰三角形

bool IsEquilateral() const; //判断该三角形是否等边三角形

double CalculateSide(Point p1, Point p2); //根据顶点p1和p2计算边长

private:

Point A,B,C; //三角形三个顶点

double AB,BC,AC; //三角形的三条边长

};

Triangle::Triangle(Point a, Point b, Point c)

{

A = a;

B = b;

C = c;

AB = CalculateSide(A,B);

AC = CalculateSide(A,C);

BC = CalculateSide(B,C);

}

double Triangle::CalculateSide(Point p1, Point p2)

{

return sqrt(\_(p1.GetX()-p2.GetX())\*(p1.GetX()-p2.GetX())+(p1.GetY()-p2.GetY())\*(p1.GetY()-p2.GetY()));

}

bool Triangle::IsIsosceles() const

{

if( fabs(AB-BC) < MINNUM || fabs(AB-AC) < MINNUM || fabs(BC-AC) < MINNUM)

return \_ true\_\_\_\_\_\_\_\_\_\_\_;

else

return \_ false\_\_\_\_\_\_\_\_\_\_\_;

}

bool Triangle::IsEquilateral() const

{ if (\_fabs(AB-BC)<MINNUM && fabs(AB-AC)<MINNUM && fabs(BC-AC)<MINNUM))

return true;

else

return false;

}

int main()

{

Point p1(2.0,0.0), p2(0.0,2.0), p3(0.0,0.0),p4(1,sqrt(3));

Triangle t1(p1,p2,p3), t2(p1,p3,p4);

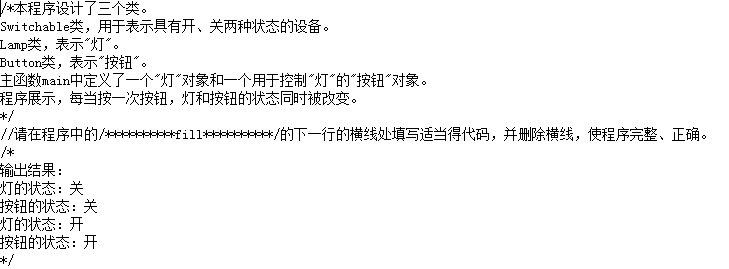
cout << t1.IsIsosceles() << t1.IsEquilateral() << endl;

cout << t2.IsIsosceles() << t2.IsEquilateral() << endl;

return 0;

}

5



#include<iostream>

using namespace std;

class Switchable{ //具有开、关两种状态的设备

bool is\_on; //为 true 表示"开"，为 false 表示"关"

public:

Switchable(): is\_on(false){}

void switchOn(){ is\_on=true; } //置为"开"状态

void switchOff(){ is\_on=false; }//置为"关"状态

bool isOn(){ \_\_return is\_on;\_\_\_\_\_\_\_\_\_\_\_\_\_ } //返回设备状态

virtual const char \*getDeviceName()\_\_\_=0\_\_\_\_; //返回设备名称的纯虚函数

};

class Lamp: public Switchable{

public:

//返回设备名称，用于覆盖基类中的纯虚函数

const char \*getDeviceName(){ return "Lamp"; }

};

class Button{ //按钮

Switchable \*device; //按钮控制的设备

public:

Button(Switchable &dev):\_\_\_\_ device(&dev)\_\_\_\_\_\_\_\_{} //用参数变量的地址初始化device

bool isOn(){ return device->isOn(); } //按钮状态

void push(){ //按一下按钮改变状态

if(isOn()) \_\_\_\_\_\_device->switchOff()\_\_\_\_\_\_\_\_\_\_\_;

else device->switchOn();

}

};

int main(){

Lamp lamp;

Button button(lamp);

cout<<"灯的状态："<<(lamp.isOn()? "开" : "关")<<endl;

cout<<"按钮的状态："<<(button.isOn()? "开" : "关")<<endl;

button.push(); //按一下按钮

cout<<"灯的状态："<<(lamp.isOn()? "开" : "关")<<endl;

cout<<"按钮的状态："<<(button.isOn()? "开" : "关")<<endl;

return 0;

}

参考答案

单项选择

C D A D D

D D C D A

D A D C A

程序设计

1 示例代码：

int func(int n)

{

int num=0;

while(n!=1)

{

if(n%2==0)

n/=2;

else

n=(3\*n+1)/2;

num++;

}

return num;

}

2 示例代码：

int num,sum=0;

for(num=m;num<=n;num++)

{

if(num%3==0&&num%7==0)

sum+=num;

}

return sum;

3 示例代码：

for(int i=low;i<=up;i++)

{

for(int j=2;j<i;j++)

if(i%j==0)break;

if(j==i) count++;

程序填空

1 答案：

位置 1:0;

位置 2:count+=1; 【或】 count=count+1; 【或】 count++; 【或】 ++count;

位置 3:sum+=a[i]; 【或】 sum=sum+a[i];

2 答案：

位置 1:Person()

位置 2:virtual void

位置 3: :Person(s), gpa(g)

位置 4:Person\* p

3 答案：

位置 1:j=4

位置 2:

j>i 【或】 i<j 【或】 j>=i 【或】 i<=j 【或】 i<3 【或】 i<=2 【或】 i<2 【或】 i<=1 【或】 j>1 【或】 j>2 【或】 j>=2 【或】 j>=3

位置 3:ch=st[i];

位置 4:st[i]=st[j];

4 答案：

位置 1:

(p1.GetX()-p2.GetX())\*(p1.GetX()-p2.GetX())+(p1.GetY()-p2.GetY())\*(p1.GetY()-p2.GetY()) 【或】 (p2.GetX()-p1.GetX())\*(p2.GetX()-p1.GetX())+(p2.GetY()-p1.GetY())\*(p2.GetY()-p1.GetY()) 【或】 pow(p1.GetX()-p2.GetX(),2)+pow(p1.GetY()-p2.GetY(), 2) 【或】 pow(p2.GetX()-p1.GetX(),2)+pow(p2.GetY()-p1.GetY(), 2) 【或】 pow(p1.GetY()-p2.GetY(), 2)+pow(p1.GetX()-p2.GetX(),2) 【或】 pow(p2.GetY()-p1.GetY(), 2)+pow(p2.GetX()-p1.GetX(),2)

位置 2:true

位置 3:false

位置 4:fabs(AB-BC)<MINNUM && fabs(AB-AC)<MINNUM && fabs(BC-AC)<MINNUM)

5 答案：

位置 1:return is\_on;

位置 2:=0

位置 3:device(&dev)

位置 4:device->switchOff()