11j192

20191003359

黄邦宇

一、

1-5 bdcbd

6-10 bcbdd

11-15 dcdab

二、

16. ->17. this指针

18. 友元

19. 初始化列表

20. delete [] pa

21. 不是

22. 纯虚函数 纯虚函数 抽象类

23. 先执行派生类的析构函数，再执行基类的析构函数

24. 多态、参数多态、包含多态

25. 可以 可以

26. 虚基类 作用域符

27. private 加点操作符

28. 抛出异常、传递信息

捕获异常

29.答：

（1）用一个对象初始化另一个对象

（2）如果函数的形参是类的对象,调用函数时,进行形参和实参结合时。

（3）

30.答：

（1）不同：

编译时的多态是通过静态连编来实现的；

运行时的多态是用动态连编来实现的。

（1）实现：

编译时的多态性主要是通过函数重载和运算符重载来实现的；

运行时的多态性主要是通过虚函数来实现的。

31.答：

优点：

因为类的特性使其便于维护扩展

缺点:

过于繁琐复杂

三、

32.答：

20

20

30

20

33.答：

Instance 1 is constructed

Instance 2 is constructed

Instance 3 is constructed

34.答：

i:10

j:12

35.答：

B's construcyor called.

B's default constructor called.

C's constuctor called.

5

6

C's destructor called.

B's destructor called.

B's destructor called.

36.答：

错误1:

15行，显示的是x、y而非a、b

修改为：

void f1（）{A::Show();}

错误2：

12行，B的构造函数的初始化列表

修改为：

B(int i,int j,int k,int l): A(i,j){x=k;y=l;}

37.答：

【1】：virtual

【2】：3.14\*r\*r

38.答：

#include<iostream>

using namespace std;

class Complex {

private:

double m\_true；

double m\_image；

public:

Complex(double a, double b) :m\_true(a), m\_image（b) {}

Complex operator+(const Complex &a) {

this->m\_true += a.m\_true;

this->m\_false += a.m\_false;

Complex \*m(this);

return \*m;

}

Complex operator-(Complex& a) {

this->m\_true -= a.m\_true;

this->m\_image -= a.m\_image；

Complex\* m(this);

return \*m;

}

Complex operator\*(Complex& a) {

double f, t;

t = this->m\_true \* a.m\_true - this->m\_false \* a.m\_false;

this->m\_true = t;

f = this->m\_true \* a.m\_image + this->m\_image \* a.m\_true;

Complex\* m(this);

return \*m;

}

friend ostream& operator<<(ostream& b, Complex a);

};

ostream& operator<<( ostream &b,Complex a) {

cout << a.m\_true;

if (a.m\_image > 0)

cout << "+" << a.m\_false << "i" << endl;

if (a.m\_image < 0)

cout << a.m\_false << "i" << endl;

if (a.m\_image == 0)

cout << endl;

return b;

}

void main() {

Complex c1(2.0, 3.0);

Complex c2(1.0, 2.0);

cout << c1 + c2 << endl;

cout << c1 - c2 << endl;

cout << c1 \* c2 << endl;

}

39