1. 简述构造函数和析构函数的作用。

用于新建对象的初始化工作。 析构函数的作用：用于在撤销对象前，完成一些清理工作，比如：释放内存等。

2. Vector和数组的区别。并且回答Vector是如何创建兼容不同类型的变量。

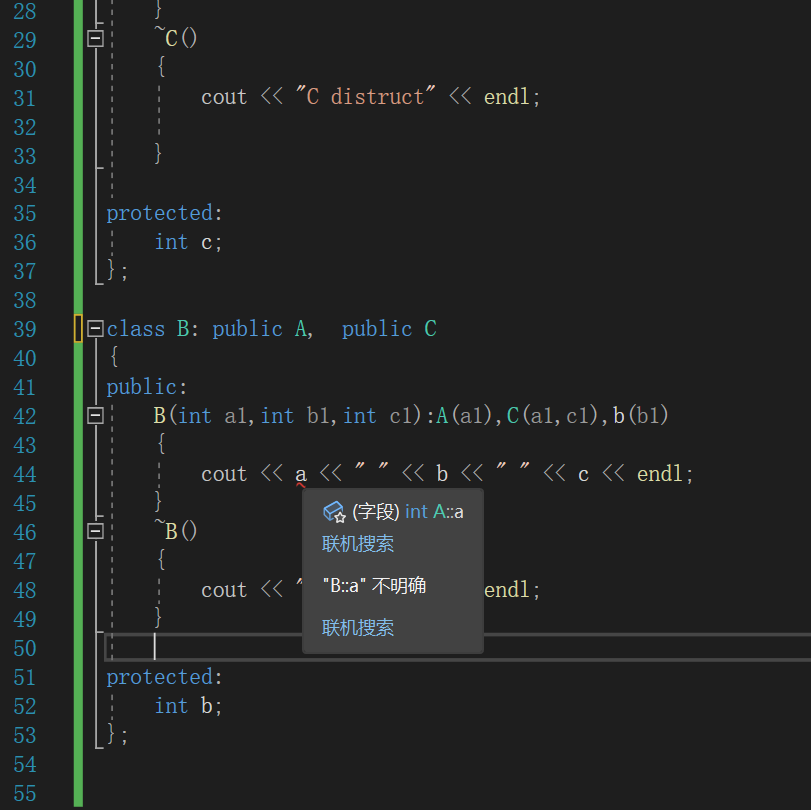
数组是存放类型相同的对象的容器，数组的大小确定不变，不能随意向数组中增加元素。

vector是类型相同的对象的容器，vector的大小可以变化，可以向数组中增加元素。

数组是静态分配空间，一旦分配了空间的大小，就不可再改变了；而vector是动态分配空间，随着元素的不断插入，它会按照自身 的一套机制不断扩充自身的容量。

3.若a->b表示b继承于a，则当A->B，A->C，C->B时，当C修改A中的变量会发生什么bug，如何解决

我通过实验发现和之前程序课程里面那种A->B , A->C, D->B, D->C 的问题类似；在相同情况下改变C会出现变量不明确的问题，



我通过虚拟继承来实现例如：

#include <iostream>

using namespace std;

class A

{

public:

A(int a1):a(a1)

{

cout << a << endl;

}

~A()

{

cout << "A distruct" << endl;

}

protected:

int a;

};

class C:virtual public A

{

public:

C(int a1,int c1):A(a1),c(c1)

{

cout << a << " " << c << endl;

}

~C()

{

cout << "C distruct" << endl;

}

protected:

int c;

};

class B:virtual public A, virtual public C

{

public:

B(int a1,int b1,int c1):A(a1),C(a1,c1),b(b1)

{

cout << a << " " << b << " " << c << endl;

}

~B()

{

cout << "B distruct" << endl;

}

protected:

int b;

};

int main()

{

int a, b, c;

cin >> a >> b >> c;

A a1(a);

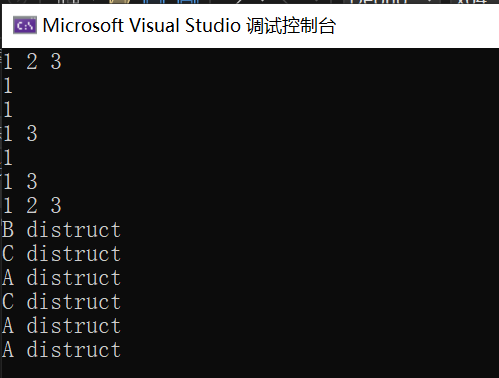
C c1(a, c);

B b1(a, b, c);

return 0;

}

结果



4. 简述拷贝构造函数模板

**浅构造：**

#include<iostream >

using namespace std;

class Complex

{

public:

double real, imag;

Complex(double r, double i)

{

real = r; imag = i;

}

};

int main() {

Complex cl(1, 2);

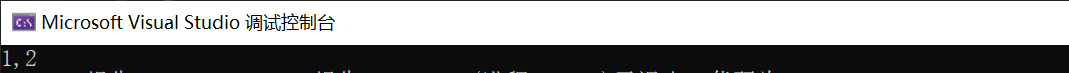
Complex c2(cl); //用复制构造函数初始化c2,浅构造没有写构造函数的默认构造；

cout << c2.real << "," << c2.imag; //输出 1,2

return 0;

}

**结果：**



非默认构造：

#include<iostream>

using namespace std;

class Complex {

public:

double real, imag;

Complex(double r, double i) {

real = r; imag = i;

}

Complex(const Complex& c) {

real = c.real; imag = c.imag;

cout << "Copy Constructor called" << endl;

}

};

int main() {

Complex cl(1, 2);

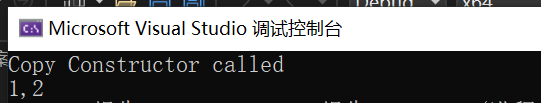
Complex c2(cl); //调用复制构造函数

cout << c2.real << "," << c2.imag;

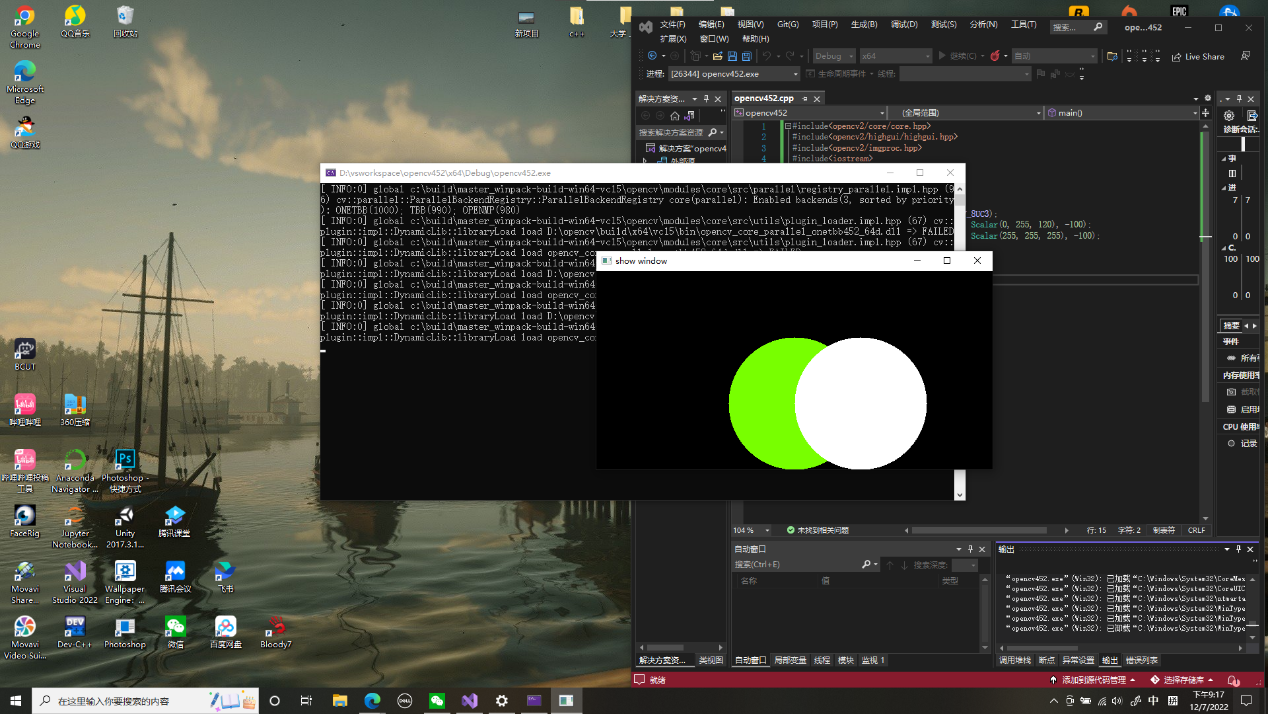
return 0;

}

结果



5. 安装C++opencv（版本4.5.2或以上）并安装对应版本opencv\_contrib



6.简述矩阵乘法，并说明矩阵相乘的前提条件：

最简单的说法就是两个矩阵相乘左矩阵的列数等于右矩阵的行数。