实验一：

#include<iostream>

using namespace std;

enum CPU\_Rank { P1 = 1, P2, P3, P4, P5, P6, P7 };

class CPU {

public:

CPU() {};

CPU(enum CPU\_Rank rank2, int frequency, float voltage) :rank(rank2), freqt(frequency), vol(voltage) {};

~CPU() {};

void run() { cout << "come on baby" << endl; };

void stop() { cout<<"let's go party"; };

private:

enum CPU\_Rank rank;

int freqt;

float vol;

};

int main() {

CPU cpu1(P2, 2.2, 3);

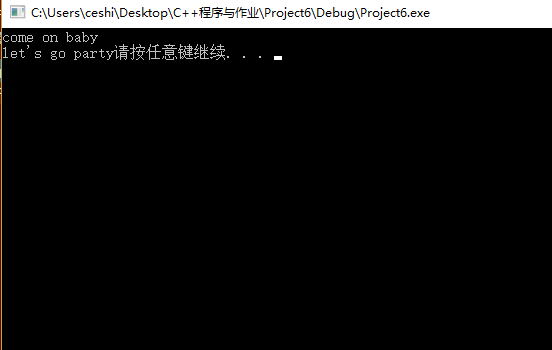
cpu1.run();

cpu1.stop();

system("pause");

return 0;

}



**实验二：**

#include <iostream>

using namespace std;

class CDROM {

public:

CDROM() {

cout << "我是CDROM" << endl;

}

};

class RAM {

public:

RAM() {

cout << "我是RAM" << endl;

}

};

class CPU {

public:

CPU() {

cout << "我是CPU" << endl;

}

};

class computer {

public:

computer(CDROM c, RAM r, CPU u) :c1(c), r2(r), u3(u) {

cout << "组合类的应用" << endl;

};

void run() {};

void stop() {};

private:

CDROM c1;

RAM r2;

CPU u3;

};

int main() {

CDROM CD;

RAM RA;

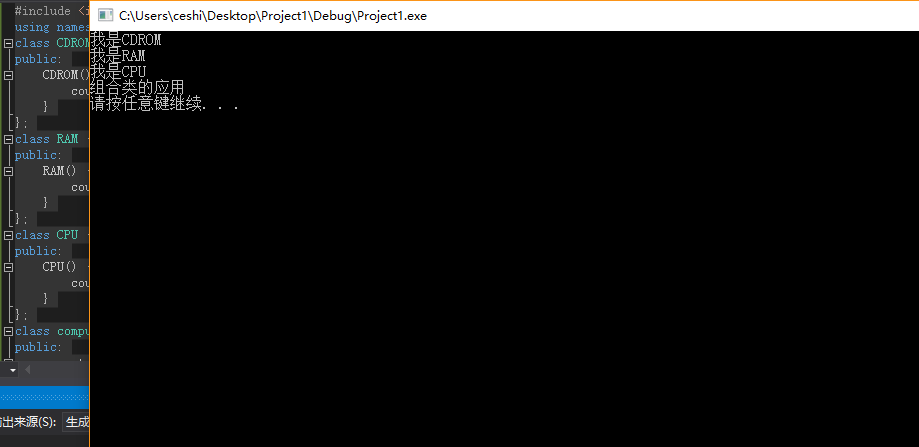
CPU A;

computer(CD, RA, A);

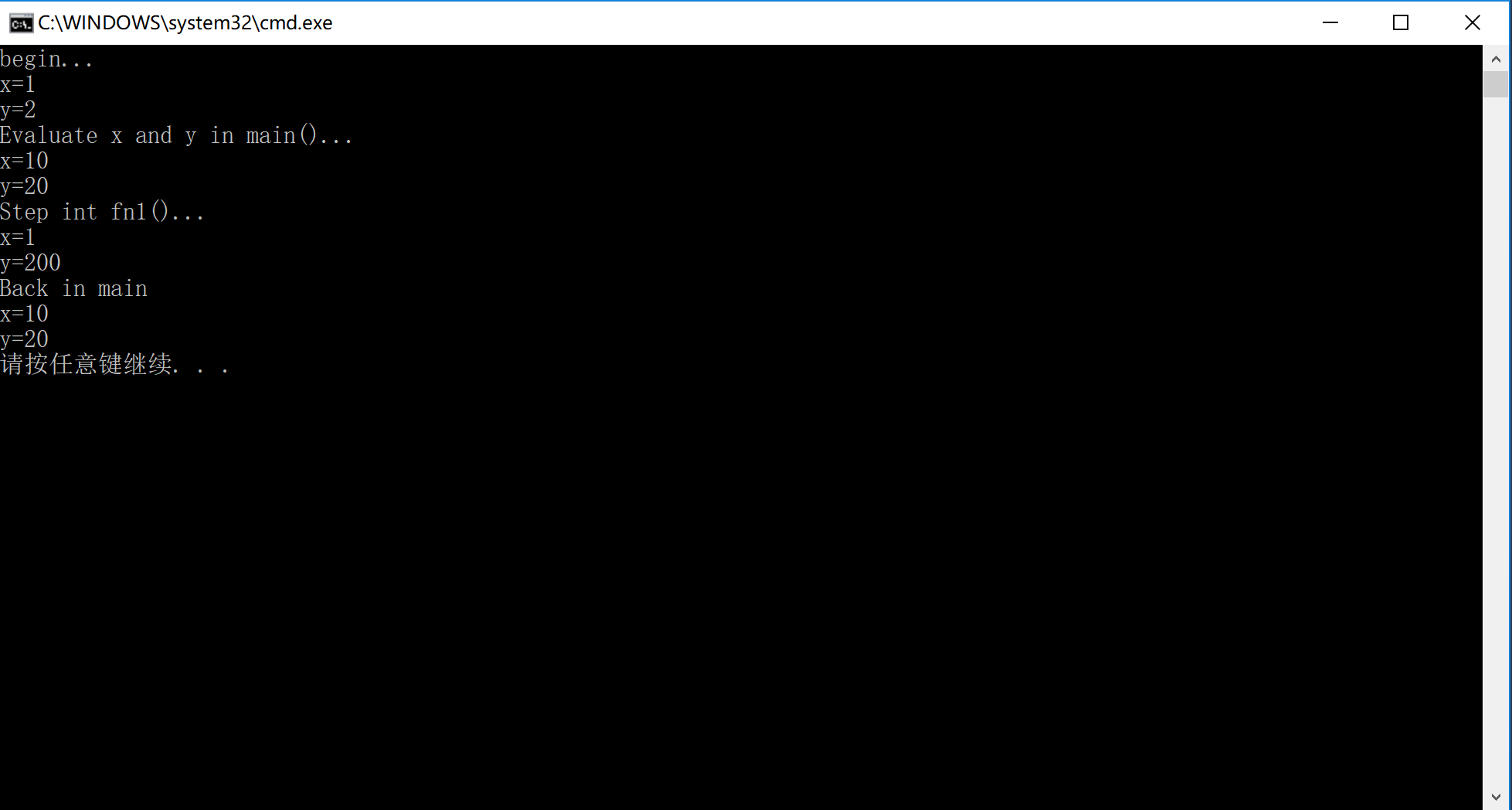
system("pause");

return 0;

}



**实验三**

**（1）**

**实验三（2）**

//client.h

#include<iostream>

#ifndef client\_h

#define client\_h

class client

{

public:

client() {};

client(int i);

client operator + (client &c);

client(client & c);

client operator++(int);

void show();

private:

static int count;

int i;

};

#endif

//client.cpp

#include "client.h"

using namespace std;

int client::count = 0;

void client::show() {

cout << i << endl;

}

client client::operator++(int) {

i++;

return client(i);

}

client client::operator +(client &c) {

count += c.i;

return client(i + c.i);

};

client::client(int i) :i(i) {

count += i;

};

client::client(client &c) :i(i) {

count += i;

cout << "CopyConsturctor" << endl;

};

int main() {

client a1(5), a2(7), a3,a4;

cout << "a1=";

a1.show();

cout << "a2=";

a2.show();

a3 = a1 + a2;

a4 = a2++;

cout << "a1+a2=";

a3.show();

cout << "a2+1=";

a4.show();

return 0;

}

