关于拷贝构造函数:

#include<string>

#include<iostream>

using namespace std;

class student

{

char \*specialty;

public:

student(char \*pspec = 0); //构造函数声明

~student();

void show();

};

student::student(char \*pspec)

{

if (pspec)

{

specialty = new char[strlen(pspec) + 1];

strcpy(specialty, pspec);

}

else

specialty = 0;

}

student::~student()

{

if (specialty)

delete[]specialty;

}

void student::show()

{

cout << "specialty=" << specialty << '/n';

}

int main()

{

char content1[10] = "computer";

char content2[10] = "zhang";

student zhang(content1);

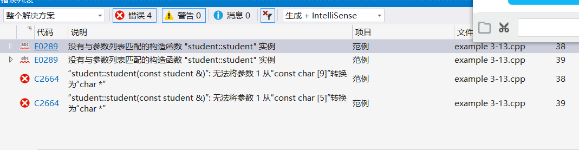
student wang(content2);

zhang.show();

wang.show();

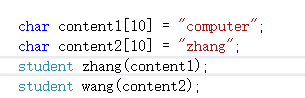
return 0;

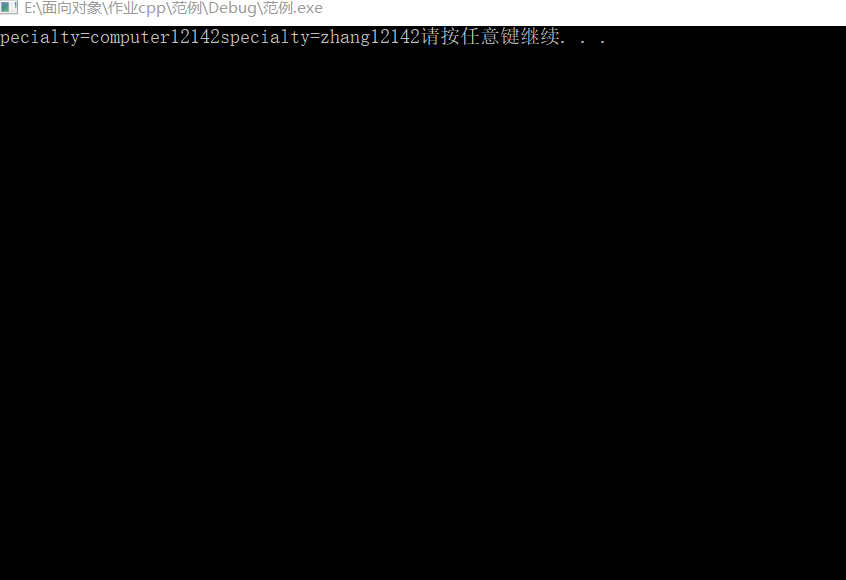
}

分析:在编译时会出现 

这是因为由于Student没有定义拷贝构造函数，因此当语句Student wang(zhang) ;定义对象wang时，系统将调用默认的拷贝构造函数，负责将对象zhang的数据成员指针变量specialty中存放的地址值赋给对象wang的数据成员指针变量specialty  
对象zhang复制给对象wang的仅是其指针类型数据成员speciaty的值，即仅是个地址值，并没有另外生成动态空间，因此没有把pecialy 所指向的动态存储空间的内容

通过定义拷贝构造函数实现深拷贝可以解决浅拷贝所带来的指针悬挂问题.



即出现所得结果. 

3.\_3\_1.CPP:

#include<iostream>

#include<windows.h>

using namespace std;

class B

{

int x, y;

public:

B()

{

x = y = 0;

cout << "con1\t";

}

B(int i)

{

x = i; y = 0;

cout << "con2\t";

}

B(int i, int j)

{

x = i; y = j;

cout << "con3\t";

}

~B()

{

cout << "Des\t";

}

};

int main()

{

B \*ptr;

ptr = new B[3];

ptr[0] = B();

ptr[1] = B(1);

ptr[2] = B(2, 3);

delete[]ptr;

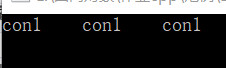
system("pause");

return 0;

}

分析: 三个重载函数

当运行 ptr[0] = B()时出现



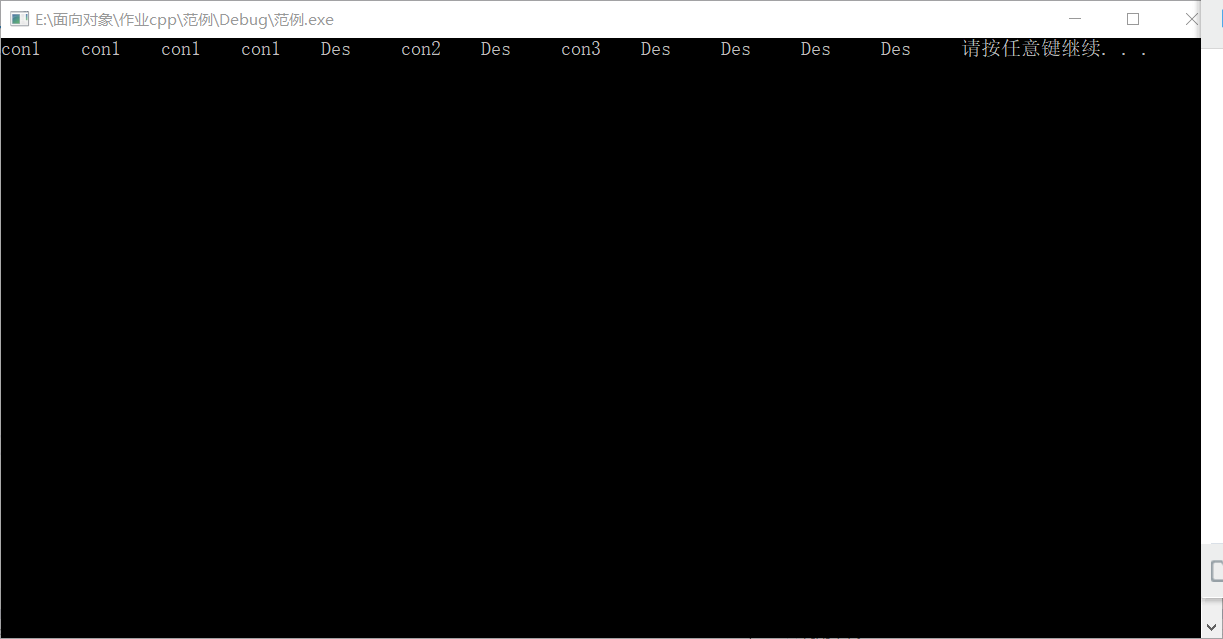
当运行ptr[1] = B(1)时出现



当运行ptr[2] = B(2, 3)时出现



最后结果如图所示:



3\_3\_2.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include<string>

#include<windows.h>

using namespace std;

class student

{

int age;

char \*name;

public:

student(int m, const char \*n)

{

age = m;

name = new char[strlen(n) + 1];

strcpy(name, n);

}

friend void disp(student&);

~student()

{

cout << "delete it." << name << endl;

delete[]name;

}

};

void disp(student & p)

{

cout << "student's name is " << p.name << ",age is " << p.age << endl;

}

int main()

{

student A(18, "wujiang");

student B(19, "xiayu");

disp(A);

disp(B);

system("pause");

return 0;

}

分析: 在student类中定义了 名字和年龄.包括两个函数.程序结果如下: