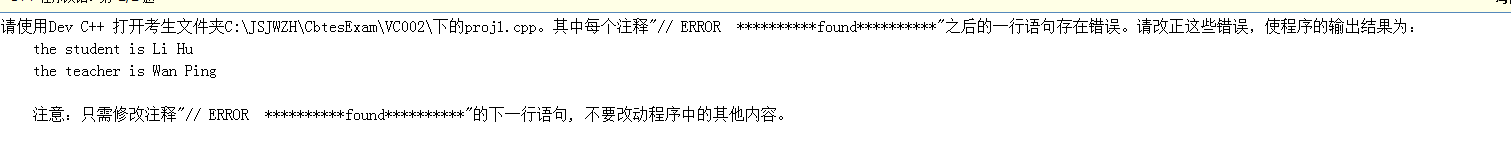
**改错题**



#include <iostream>

using namespace std;

class teacher ;

class student

{

const char \*name ;

public:

student(const char \*s)

{ name = s ; }

friend void print(student &, teacher &) ;

};

class teacher

{

const char \*name ;

public:

teacher(const char \*s)

{ name = s ; }

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

friend void print(student &, teacher &) ;

};

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

void print(student &a, teacher &b)

{

cout << "the student is " << a.name << endl ;

cout << "the teacher is " << b.name << endl ;

}

int main()

{

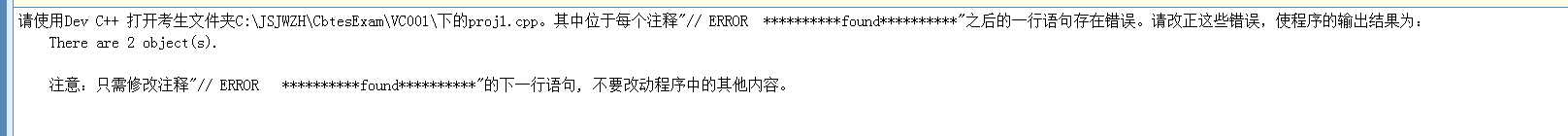
student s("Li Hu") ;

teacher t("Wan Ping") ;

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

print(s,t) ;

}



// proj1.cpp

#include <iostream>

using namespace std;

class MyClass {

public:

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

MyClass(int i = 0) :value (i)

// MyClass() :value(0)

{ count++; }

void Print()

{ cout << "There are " << count << " object(s)." << endl; }

private:

const int value;

static int count;

};

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

int MyClass::count = 0;

int main()

{

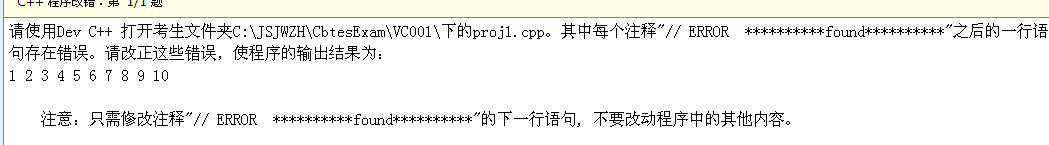
MyClass obj1, obj2;

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

obj1.Print();

return 0;

}



// proj1.cpp

#include <iostream>

using namespace std;

class MyClass {

public:

MyClass(int len)

{

array = new int[len];

arraySize = len;

for(int i = 0; i < arraySize; i++)

array[i] = i+1;

}

~MyClass()

{

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

delete []array;

}

void Print() const

{

for(int i = 0; i < arraySize; i++)

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

cout << array[i] << ' ';

cout << endl;

}

private:

int \*array;

int arraySize;

};

int main()

{

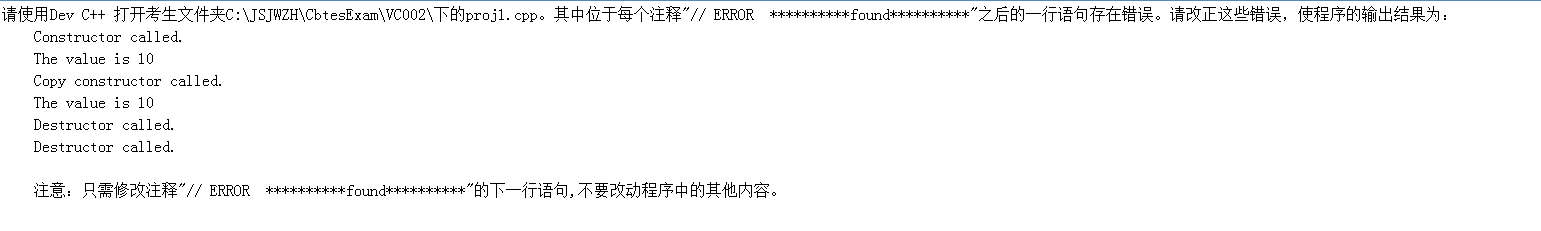
// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

MyClass obj(10);

obj.Print();

return 0;

}



// proj1.cpp

#include <iostream>

using namespace std;

class MyClass {

public:

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

MyClass(int i=10)

{ value = i; cout << "Constructor called." << endl; }

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

MyClass(const MyClass &p)

{

value=p.value;

cout << "Copy constructor called." << endl;

}

void Print()

{ cout << "The value is " << value << endl; }

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

~MyClass()

{ cout << "Destructor called." << endl; }

private:

int value;

};

int main()

{

MyClass obj1;

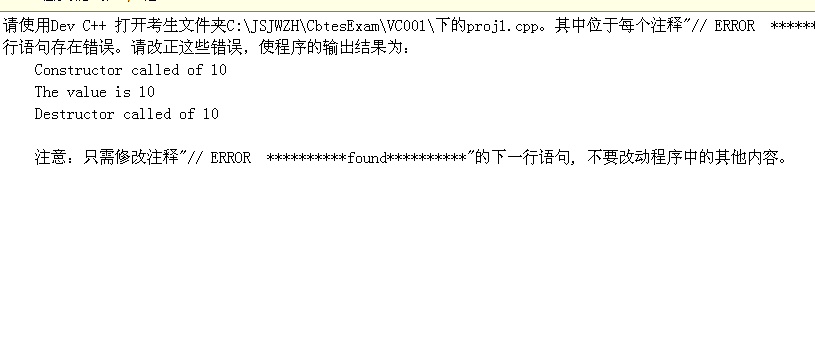
obj1.Print();

MyClass obj2(obj1);

obj2.Print();

return 0;

}



// proj1.cpp

#include <iostream>

using namespace std;

class MyClass {

public:

MyClass(int i)

{

value = i;

cout << "Constructor called of " << value << endl;

}

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

void Print()const

{ cout << "The value is " << value << endl; }

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

~MyClass()

{ cout << "Destructor called of " << value << endl; }

private:

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

int value ;

};

int main()

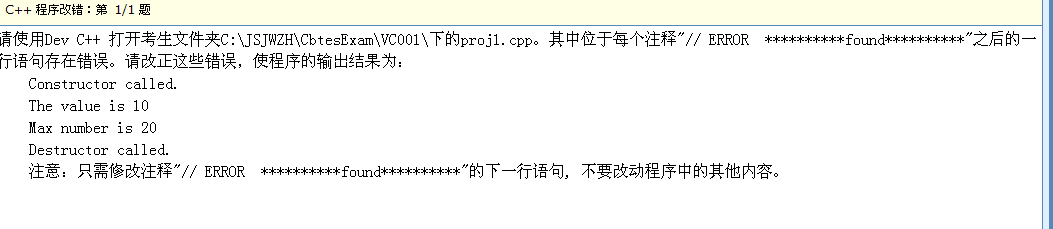
{

const MyClass obj(10);

obj.Print();

return 0;

}



// proj1.cpp

#include <iostream>

using namespace std;

class MyClass {

public:

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

MyClass(int i)

{ value = i; cout << "Constructor called." << endl; }

int Max(int x, int y) { return x>y ? x : y; } // 求两个整数的最大值

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

int Max(int x, int y, int z ) // 求三个整数的最大值

{

if (x > y)

return x>z ? x : z;

else

return y>z ? y : z;

}

int GetValue() const { return value; }

~MyClass() { cout << "Destructor called." << endl; }

private:

int value;

};

int main()

{

MyClass obj(10);

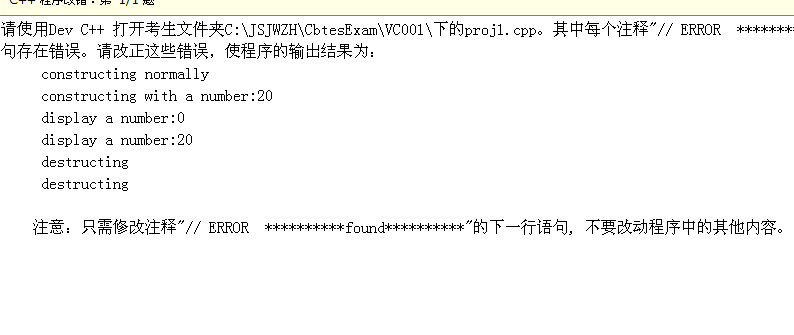
// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

cout << "The value is " << obj.GetValue() << endl;

cout << "Max number is " << obj.Max(10,20) << endl;

return 0;

}



#include <iostream>

using namespace std;

class Sample

{

public:

Sample()

{

x = 0 ;

cout << "constructing normally" << endl ;

}

Sample(int m)

{

x = m ;

cout << "constructing with a number:" << x << endl ;

}

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

~Sample()

{

cout << "destructing" << endl ;

}

void display()

{

cout << "display a number:" << x << endl ;

}

protected:

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

int x ;

};

int main()

{

Sample obj1 ;

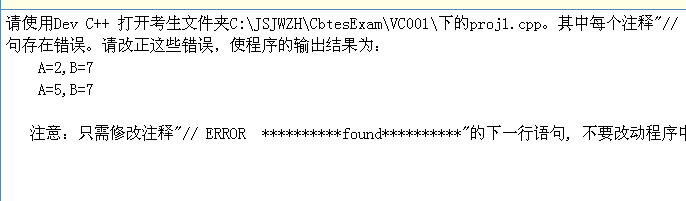
// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

Sample obj2(20) ;

obj1.display() ;

obj2.display() ;

}



#include <iostream>

using namespace std;

class Sample

{

int A ;

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

static int B ;

public:

Sample(int a) { A = a ; B += a ;}

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

static void func( Sample s)

{

cout << "A=" << s.A << ",B=" << B << endl ;

}

};

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

int Sample::B =0;

int main()

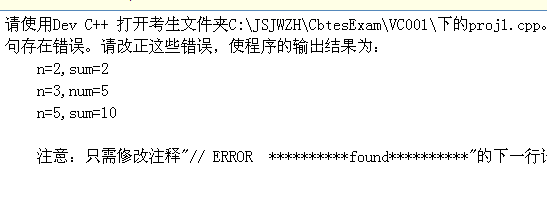
{

Sample s1(2), s2(5) ;

Sample::func(s1) ;

Sample::func(s2) ;

}



#include <iostream>

using namespace std;

class Sample

{

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

int n ;

static int sum ;

public:

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

Sample(int x)

{

n = x ;

}

void add()

{

sum += n ;

}

void display()

{

cout << "n=" << n << ",sum=" << sum << endl ;

}

};

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

int Sample::sum = 0 ;

int main()

{

Sample a(2), b(3), c(5) ;

a.add() ;

a.display() ;

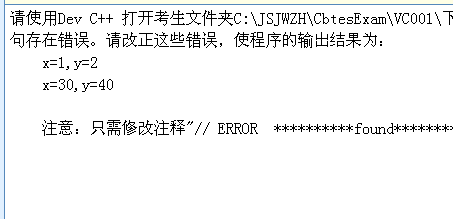
b.add() ;

b.display() ;

c.add() ;

c.display() ;

}



#include <iostream>

using namespace std;

class Sample

{

int x,y ;

public:

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

Sample()

{

x = y = 0 ;

}

Sample(int i, int j)

{

x = i ; y = j ;

}

void copy(Sample &s)

{

x = s.x ; y = s.y ;

}

void setxy(int i, int j)

{

x = i ; y = j ;

}

void print()

{

cout << "x=" << x << ",y=" << y << endl ;

}

};

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

void func(Sample s1, Sample &s2)

{

s1.setxy(10,20) ;

s2.setxy(30,40) ;

}

int main()

{

Sample p(1, 2), q ;

// ERROR \*\*\*\*\*\*\*\*\*\*found\*\*\*\*\*\*\*\*\*\*

q.copy(p) ;

func(p, q) ;

p.print() ;

q.print() ;

}