上机实验5

派生类与继承

程序代码：

#include<iostream>

using namespace std;

class Base{

public:

void setx(int i)

{ x=i; }

int getx()

{ return x; }

public:

int x;

};

class Derived:public Base{

public:

void sety(int i)

{ y=i; }

int gety()

{ return y; }

void show()

{ cout<<"Base::x="<<x<<endl; }

public:

int y;

};

int main()

{

Derived bb;

bb.setx(16);

bb.sety(25);

bb.show();

cout<<"Base::x="<<bb.x<<endl;

cout<<"Derived::y="<<bb.y<<endl;

cout<<"Base::x="<<bb.getx()<<endl;

cout<<"Derived::y="<<bb.gety()<<endl;

return 0;

}

运行结果：

Base::x=16

Base::x=16

Derived::y=25

Base::x=16

Derived::y=25

实验心得：  
按照以下要求，对程序进行修改后再调试，指出调试中出错的原因。

将基类Base中数据成员x的访问权限改为private时，会出现哪些错误，为什么？

5\_test.cpp: In member function ‘void Derived::show()’:

5\_test.cpp:19:25: error: ‘int Base::x’ is private within this context

19 | { cout<<"Base::x="<<x<<endl; }

| ^

5\_test.cpp:10:9: note: declared private here

10 | int x;

| ^

5\_test.cpp: In function ‘int main()’:

5\_test.cpp:29:26: error: ‘int Base::x’ is private within this context

29 | cout<<"Base::x="<<bb.x<<endl;

| ^

5\_test.cpp:10:9: note: declared private here

10 | int x;

| ^

5\_test.cpp:29:26: note: field ‘int Base::x’ can be accessed via ‘int Base::getx()’

29 | cout<<"Base::x="<<bb.x<<endl;

| ^

| getx()

原因：派生类无法访问基类私有成员

将基类Base中数据成员x的访问权限改为protected时，会出现哪些错误，为什么？

5\_test.cpp: In function ‘int main()’:

5\_test.cpp:29:26: error: ‘int Base::x’ is protected within this context

29 | cout<<"Base::x="<<bb.x<<endl;

| ^

5\_test.cpp:10:9: note: declared protected here

10 | int x;

| ^

5\_test.cpp:29:26: note: field ‘int Base::x’ can be accessed via ‘int Base::getx()’

29 | cout<<"Base::x="<<bb.x<<endl;

| ^

| getx()

原因：类外（main）无法访问类的保护成员

在原程序的基础上，将派生类Derived的继承方式改为private时，会出现哪些错误，为什么？

5\_test.cpp: In function ‘int main()’:

5\_test.cpp:26:16: error: ‘void Base::setx(int)’ is inaccessible within this context

26 | bb.setx(16);

| ^

5\_test.cpp:5:10: note: declared here

5 | void setx(int i)

| ^~~~

5\_test.cpp:26:16: error: ‘Base’ is not an accessible base of ‘Derived’

26 | bb.setx(16);

| ^

5\_test.cpp:29:26: error: ‘int Base::x’ is inaccessible within this context

29 | cout<<"Base::x="<<bb.x<<endl;

| ^

5\_test.cpp:10:9: note: declared here

10 | int x;

| ^

5\_test.cpp:31:31: error: ‘int Base::getx()’ is inaccessible within this context

31 | cout<<"Base::x="<<bb.getx()<<endl;

| ^

5\_test.cpp:7:9: note: declared here

7 | int getx()

| ^~~~

5\_test.cpp:31:31: error: ‘Base’ is not an accessible base of ‘Derived’

31 | cout<<"Base::x="<<bb.getx()<<endl;

| ^

原因：私有继承下派生类无法访问基类成员

在原程序的基础上，将派生类Derived的继承方式改为protected时，会出现哪些错误，为什么？

5\_test.cpp: In function ‘int main()’:

5\_test.cpp:26:16: error: ‘void Base::setx(int)’ is inaccessible within this context

26 | bb.setx(16);

| ^

5\_test.cpp:5:10: note: declared here

5 | void setx(int i)

| ^~~~

5\_test.cpp:26:16: error: ‘Base’ is not an accessible base of ‘Derived’

26 | bb.setx(16);

| ^

5\_test.cpp:29:26: error: ‘int Base::x’ is inaccessible within this context

29 | cout<<"Base::x="<<bb.x<<endl;

| ^

5\_test.cpp:10:9: note: declared here

10 | int x;

| ^

5\_test.cpp:31:31: error: ‘int Base::getx()’ is inaccessible within this context

31 | cout<<"Base::x="<<bb.getx()<<endl;

| ^

5\_test.cpp:7:9: note: declared here

7 | int getx()

| ^~~~

5\_test.cpp:31:31: error: ‘Base’ is not an accessible base of ‘Derived’

31 | cout<<"Base::x="<<bb.getx()<<endl;

| ^

原因：保护继承下类外（main）无法访问基类成员