**第五次上机实验**

1. **感想：**

这次上机的目的是掌握派生类的声明方法和派生类构造函数的定义方法掌握不同方式下，基类成员在派生类中的访问属性

并学习出现问题时，找到将基类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？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？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？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？

这三个问题。通过本次实验，我掌握了派生类的声明方法和派生类构造函数的定义方法

也掌握了不同方式下，基类成员在派生类中的访问属性

1. **代码：**

|  |
| --- |
| //版权声明  //Copyright 2021-2099 ZhantengZhang.  //202030310284 张展腾  #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; // 语句1  }  public:  int y;  };  int main()  {  Derived bb; // 语句2  bb.setx(16); // 语句3  bb.sety(25); // 语句4  bb.show(); // 语句5  cout << "Base::x=" << bb.x << endl; // 语句6  cout << "Derived::y=" << bb.y << endl; // 语句7  cout << "Base::x=" << bb.getx() << endl; // 语句8  cout << "Derived::y=" << bb.gety() << endl; // 语句9  return 0;  } |

1. **截图：**

