**实验7：继承**

**姓名\_\_金宣成\_\_\_\_\_班级\_\_计科03\_\_\_学号202203151310**

* **请阅读此说明：实验7满分100分。做完实验后请按要求将代码和截图贴入该文档。然后将此文档、源代码文件（.hpp, .cpp）打包上传到学习通。**

**实验目的：熟悉并掌握继承机制，能够利用公有继承方式建立符合用户需求的类族。**

**实验要求：按照每个类两个文件的方式（一个头文件，一个源文件）组织工程内的代码。**

**实验内容：**

**1、请仔细观察下列类声明，并回答:**

**class A { //基类**

**public:**

**A(int v1=0,int v2=0,int v3=0):a(v1),b(v2),c(v3){ }**

**void F1(){cout<<** **"F1"<<a<<" "<<b<<" "<<c<<endl;}**

**int a;**

**protected:**

**void F2( ) {cout<<"F2"<<a<<" "<<b<<" "<<c<<endl;}**

**int b;**

**private:**

**void F3(){cout<<"F3"<<a<<" "<<b<<" "<<c<<endl;}**

**int c;**

**};**

**class B: public A{**

**public:**

**//B的构造函数缺失**

**void F4( ) {cout<<"F4"<<Ba<" "<<Bb<<" "<<Bc<<endl;}**

**int Ba;**

**protected:**

**void F5( ) {cout<<"F5"<<Ba<<" "<<Bb<<" "<<Bc<<endl;}**

**int Bb;**

**private:**

**void F6(){cout<<"F6"<<Ba<<" "<<Bb<<" "<<Bc<<endl;}**

**int Bc;**

**};**

**class C: protected B{**

**public:**

**//C的构造函数缺失**

**void F7(){cout<<"F7"<<Ba<<" "<<Bb <<endl;}**

**void F8(){cout<<"F8"<<Ca<<" "<<Cb <<endl;}**

**int Ca;**

**private:**

**int Cb;**

**};**

**//测试主函数**

**int main()**

**{**

**A Aobj1,Aobj2(1,2,3);**

**B Bobj1,Bobj2(1,2,3,4,5);**

**C Cobj1,Cobj2(1,2,3,4,5,6);**

**......**

**return 0;**

}

1. **填写表格,写出第一行标识符在第一列所展示的各个作用域的访问控制方式(public,protected,private)。（10分）**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **访问域\成员名** | **a** | **b** | **c** | **F1** | **F2** | **F3** | **Ba** | **Bb** | **F4** | **F5** | **F6** | **Ca** | **Cb** | **F7** |
| **A** | Pub | Pro | Pri | Pub | Pro | Pri |  |  |  |  |  |  |  |  |
| **B** | Pub | Pro | Non | Pub | Pri | Non | Pub | Pro | Pub | Pro | Pri |  |  |  |
| **C** | Pro | Pro | Non | Pro | Non | Non | Pro | Pro | Pro | Pro | Non | Pub | Pri | Pub |
| **main函数A** | Yes | Non | Non | Yes | Non | Non |  |  |  |  |  |  |  |  |
| **main函数B** | Yes | Non | Non | Yes | Non | Non | Yes | Non | Yes | Non | Non |  |  |  |
| **main函数C** | Non | Non | Non | Non | Non | Non | Non | Non | Non | Non | Non | Yes | Non | Yes |

1. **补充完类B和类C缺失的构造函数，并将main的测试程序补充完整。要求在main中展示类A，类B，类C的所有可在main中访问的成员。（40分）**

* **补充B的构造函数：**

B(int v1 = 0, int v2 = 0, int v3 = 0, int v4 = 0, int v5 = 0, int v6 = 0) : A(v1, v2, v3), Ba(v4), Bb(v5), Bc(v6) {}

* **补充C的构造函数：**

C(int v1 = 0, int v2 = 0, int v3 = 0, int v4 = 0, int v5 = 0, int v6 = 0,int v7 = 0,int v8 = 0) : B(v1, v2, v3, v4, v5, v6), Ca(v7), Cb(v8){}

* **main函数：**

#include <iostream>

#include "../hpp/C.hpp"

using namespace std;

int main()

{

    A Aobj1, Aobj2(1, 2, 3);

    B Bobj1, Bobj2(1, 2, 3, 4, 5);

    C Cobj1, Cobj2(1, 2, 3, 4, 5, 6);

    //测试所有ABC类中的成员函数是否可以用，不可用的使用注释标出

    Aobj1.F1();

    Aobj2.F1();

    //A.F2();

    //A.F3();

    Bobj1.F1();

    Bobj2.F1();

    //B.F2();

    //B.F3();

    Bobj1.F4();

    //B.F5();

    //B.F6();

    //C.F1();

    //C.F2();

    //C.F3();

    //C.F4();

    //C.F5();

    //C.F6();

    Cobj1.F7();

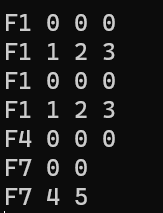
    Cobj2.F7();

    //C.F8();

    return 0;

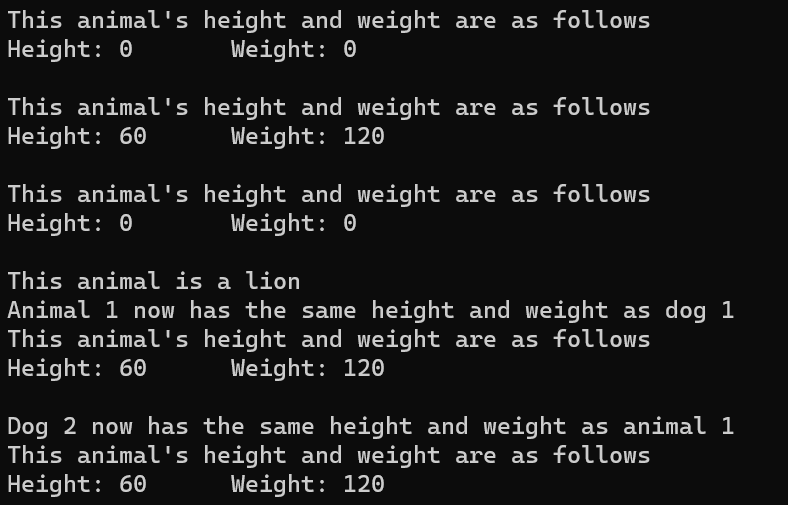
}

* **程序运行结果截屏：**

****

**2、代码调试: 附件中的代码在建立类族的过程中，由于编程人员的疏忽，出现了一些小问题，请帮忙修改过来。（20分）**

* **修改后的代码运行结果截屏：**

****

**3、设计交通工具类族: 开发一个名为Vehicle 的类的层次体系。创建两个类Taxi 和Truck，均以公有模式从类Vehicle 中继承而来。Taxi 类中应包含一个数据成员passenger说明其是否载客。Truck类应包含一个数据成员cargo说明其是否载货。根据题后附的测试程序输出结果 为类Vehicle添加必要的数据成员,并为所有类添加必要的函数来控制和访问类的数据。编写一段测试程序，将Vehicle对象、Truck 对象和Taxi对象打印到屏幕。（30分）**

**测试程序输出实例为：**

Vehicle

Number of doors: 2

Number of cylinders: 6

Transmission type: 3

Color: blue

Fuel level: 14.6

Taxi

Number of doors: 4

Number of cylinders: 6

Transmission type: 5

Color: yellow

Fuel level: 3.3

The taxi has no passengers.

Truck

Number of doors: 2

Number of cylinders: 16

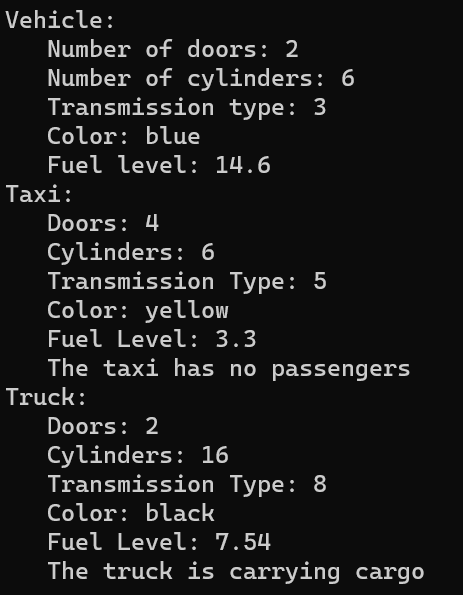
Transmission type: 8

Color: black

Fuel level: 7.54

The truck is carrying cargo.

**3.1程序测试截图**



**3.2源代码**

1.Vehicle.hpp

#ifndef VEHICLE\_HPP

#define VEHICLE\_HPP

#include <string>

using namespace std;

class Vehicle

{

    public:

        // Constructors

        Vehicle(int = 0, int = 0, int = 0, string ="", double = 0.0);

        // Getters

        int getDoors() const;

        int getCylinders() const;

        int getTransmissionType() const;

        string getColor() const;

        double getFuelLevel() const;

        // Setters

        void setDoors(int);

        void setCylinders(int);

        void setTransmissionType(int);

        void setColor(string);

        void setFuelLevel(double);

        // Other

        void print() const;

    protected:

        int doors;

        int cylinders;

        int transmission\_type;

        string color;

        double fuel\_level;

};

#endif

2.Taxi.hpp

#ifndef TAXI\_HPP

#define TAXI\_HPP

#include <string>

#include "Vehicle.hpp"

class Taxi:public Vehicle

{

    public:

        // Constructors

        Taxi(int = 0, int = 0, int = 0, string ="", double = 0.0, bool = false);

        // Getters

        bool getCustomers() const;

        // Setters

        void setCustomers(bool);

        // Other

        void print() const;

    protected:

        bool passengers;

};

#endif

3.Truck.hpp

#ifndef TRUCK\_HPP

#define TRUCK\_HPP

#include "Vehicle.hpp"

class Truck:public Vehicle

{

    public:

        // Constructors

        Truck(int = 0, int = 0, int = 0, string ="", double = 0.0, bool = false);

        // Getters

        bool getCargo() const;

        // Setters

        void setCargo(bool);

        // Other

        void print() const;

    protected:

        bool cargo;

};

#endif

4.Vehicle.cpp

#ifndef TRUCK\_HPP

#define TRUCK\_HPP

#include "Vehicle.hpp"

class Truck:public Vehicle

{

    public:

        // Constructors

        Truck(int = 0, int = 0, int = 0, string ="", double = 0.0, bool = false);

        // Getters

        bool getCargo() const;

        // Setters

        void setCargo(bool);

        // Other

        void print() const;

    protected:

        bool cargo;

};

#endif

5.Taxi.cpp

#include "../hpp/Taxi.hpp"

#include <iostream>

#include <string>

using namespace std;

// Constructors

Taxi::Taxi(int doors, int cylinders, int transmission\_type, string color, double fuel\_level, bool passengers):

    Vehicle(doors, cylinders, transmission\_type, color, fuel\_level), passengers(passengers)

{

}

// Getters

bool Taxi::getCustomers() const

{

    return passengers;

}

// Setters

void Taxi::setCustomers(bool passengers)

{

    this->passengers = passengers;

}

// Other

void Taxi::print() const

{

    cout << "Taxi: " << endl;

    cout << "   Doors: " << doors << endl;

    cout << "   Cylinders: " << cylinders << endl;

    cout << "   Transmission Type: " << transmission\_type << endl;

    cout << "   Color: " << color << endl;

    cout << "   Fuel Level: " << fuel\_level << endl;

    if(passengers){

        cout << "   The taxi has passengers" << endl;

    }else{

        cout << "   The taxi has no passengers" << endl;

    }

}

6.Truck.cpp

#include <iostream>

#include "../hpp/Truck.hpp"

using namespace std;

// Constructors

Truck::Truck(int doors, int cylinders, int transmission\_type, string color, double fuel\_level, bool cargo):

    Vehicle(doors, cylinders, transmission\_type, color, fuel\_level), cargo(cargo)

{

}

// Getters

bool Truck::getCargo() const

{

    return cargo;

}

// Setters

void Truck::setCargo(bool cargo)

{

    this->cargo = cargo;

}

// Other

void Truck::print() const

{

    cout << "Truck: " << endl;

    cout << "   Doors: " << doors << endl;

    cout << "   Cylinders: " << cylinders << endl;

    cout << "   Transmission Type: " << transmission\_type << endl;

    cout << "   Color: " << color << endl;

    cout << "   Fuel Level: " << fuel\_level << endl;

    if(cargo){

        cout << "   The truck is carrying cargo" << endl;

    }else{

        cout << "   The truck is not carrying cargo" << endl;

    }

}

7.main.cpp

#include <iostream>

#include "../hpp/Vehicle.hpp"

#include "../hpp/Taxi.hpp"

#include "../hpp/Truck.hpp"

int main(){

    Vehicle vehicle1(2, 6, 3, "blue", 14.6);

    Taxi taxi1(4, 6, 5, "yellow", 3.3, false);

    Truck truck1(2, 16, 8, "black", 7.54, true);

    vehicle1.print();

    taxi1.print();

    truck1.print();

    return 0;

}