学号 E21714049 专业 计算机科学与技术 姓名 梅世祺
实验日期 2019.05.09 教师签字 成绩

实验报告

【实验名称】 继承、派生与重载

【实验目的】

1. 理解继承与派生的概念及其在面向对象程序设计中的作用。
2. 掌握通过继承派生出一个新类的方法。
3. 掌握运算符重载的基本方法。

【实验原理】

在多继承时，就可以通过将共同基类设置为虚基类，这时从不同的路径继承过来的同名数据成员在内存中就只有一个副本。

【实验内容】

实验一 大学教务人员管理系统

题目：设计一个简单的大学教务人员管理系统，系统中的人员包括学生、教师等。利用类的继承和派生实现这两类人员的类。

要求：1. 学生包括学号、姓名、性别、年龄、班级和专业等信息；教师包括工号、姓名、性别、年龄、部门、职称、研究方向等信息。

2. 在每个类中都可以实现各数据成员的设置与获取，并实现

人员全部信息的打印输出。

原理：设计三个类 Person、Student 和 Teacher，其中 Person 类作为 Student 类和 Teacher 类的基类，封装公共的数据成员和成员函数，如学工号、姓名、性别等。各类的数据成员作为私有成员，通过 getXXX/setXXX 成员函数获取和设置，通过 display 函数输出人员的全部信息。

提高（选做）：

1. 在 Student 类的基础上定义研究生类 GraduateStudent，包括导师、研究方向等属性。

2. 在 Person 类的基础上派生领导类 Leader，包含岗位、职责、任职年限等属性，并采用虚基类和多继承的方式实现学生干部类 StudentLeader 和教师干部类 TeacherLeader。

实验结果（含源码）：

一、部分头文件

首先是 Person 类的声明，包括对保护属性 id、name、age、gender 和 department 的声明，getter 方法 getId、getName、getAge、getGender 和 getDepartment 的声明以及 setter 方法 setId、setName、setAge、setGender 和 setDepartment 的声明；我们采用了组合类构造函数的格式声明了 Person 类的构造函数。请看下图：

```
class Person {
    public:
        Person() {}

        Person(
        string id,
        string name,
        string gender,
        int age,
        string department
        ) : id(id),
            name(name),
            gender(gender),
            age(age),
            department(department) {}

        string getId();
        string getName();
        string getGender();
        int getAge();
        string getDepartment();

        ~Person() {}

        void setId(string id);
        void setName(string name);
        void setGender(string gender);
        void setAge(int age);
        void setDepartment(string department);

    protected:
        string id, name, gender, department;
        int age;
};
```

由于 Student 和 Teacher 的功能类似，这里仅介绍 Student 类的声明结构，如下图：

```
class Student : virtual public Person {
    public:
        Student(
        string id,
        string name,
        string gender,
        int age,
        string department,
        string major
        ) : Person(id, name, gender, age, department),
            stuMajor(major) {}

        Student(string major) : stuMajor(major) {}

        string getMajor();

        void setMarjor(string major);
    private:
        string stuMajor;
};
```

在集成 Person 类的基础上，我们增加了私有属性 stuMajor 用于表示学生的专业信息，分别增加了 getter 方法 getMajor 和 setter 方法 setMajor 用于设置和获取学生的专业信息。

其次是 GraduateStudent 类的声明：

```
class GraduateStudent : public Student {
    public:
        GraduateStudent(
        string id,
        string name,
        string gender,
        int age,
        string department,
        string major,
        string mentor,
        string research
        ) : Person(id, name, gender, age, department),
            Student(major),
            research(research) {}

        string getMentor();
        string getResearch();

        void setMentor(string mentor);
        void setResearch(string research);

    private:
        string mentor, research;
};
```

同样的，我们增加了 mentor 和 research 私有属性以及相关的 getter 和 setter 方法。

由于 StudentLeader 类和 TeacherLeader 类的结构类似，这里只介绍 StudentLeader 类的头文件：

```
class StudentLeader : public Student, public Leader {
    public:
    StudentLeader(
    string id,
    string name,
    string gender,
    int age,
    string department,
    string major,
    string duties,
    string lengthOfService
    ) : Person(id, name, gender, age, department),
        Student(major),
        Leader(duties, lengthOfService) {}
};
```

这里运用了虚基类的知识，对于 id、name、gender、age 和 dapartment 等基本属性，我们借助虚基类可以使用 Person 的构造函数对 StudentLeader 类的对象进行初始化。

二、主函数

```
int main(int argc, char const *argv[])
{
    GraduateStudent* graduateStudent = new GraduateStudent(
        "E21714049",
        "梅世祺",
        "男",
        20,
        "英才班",
        "计算机科学与技术",
        "张伟",
        "计算机视觉"
    );

    StudentLeader* studentLeader = new StudentLeader(
        "E21714049",
        "梅世祺",
        "男",
        20,
        "英才班",
        "计算机视觉",
        "做好社会主义接班人",
        "19年3个月"
    );
```

```
    TeacherLeader* teacherLeader = new TeacherLeader(
        "E21714049",
        "张飞",
        "男",
        20,
        "院学生处",
        "培养社会主义接班人",
        "4年",
        "长江学者",
        "大数据"
    );

    cout<<graduateStudent->getAge()<<endl;
    cout<<studentLeader->getDepartment()<<endl;
    cout<<teacherLeader->getName()<<" "<<teacherLeader->getDuties()<<endl;

    teacherLeader->setDuties("培养祖国的花朵");

    cout<<teacherLeader->getName()<<" "<<teacherLeader->getDuties()<<endl;
}
```

三、输出