类型兼容

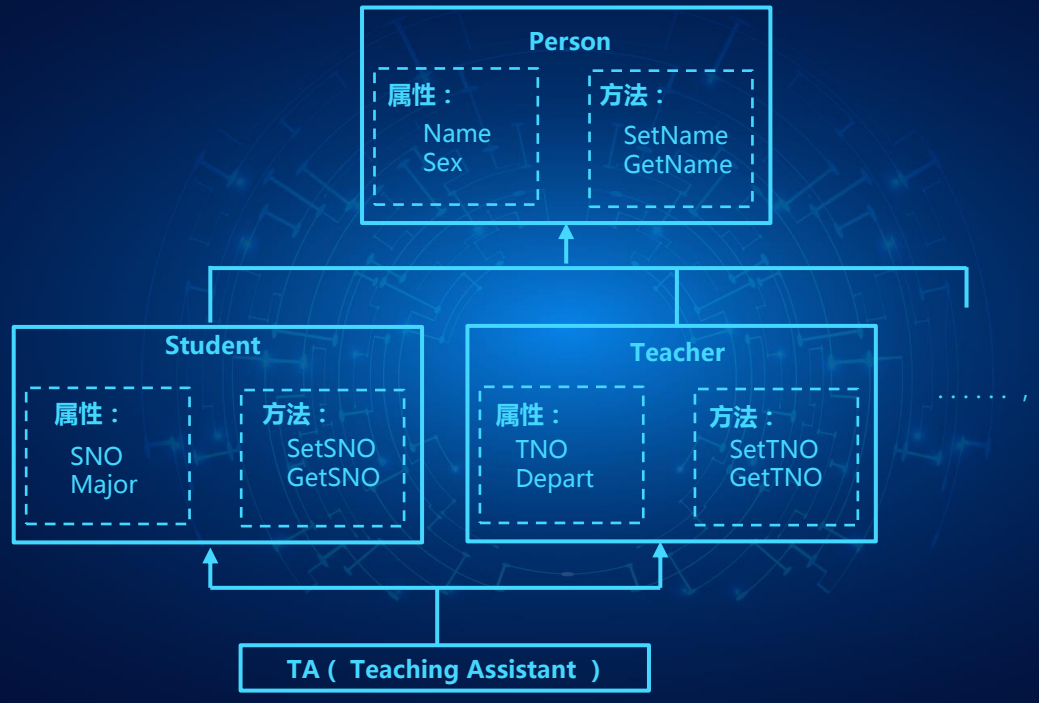
- 类型兼容是多态性的前提，指在基类对象可以出现的任何地方，都可以用公有派生类的对象来替代。类型兼容所指的是如下三种情况：

（1）可以用派生类对象为基类对象赋值；

（2）可以用派生类对象初始化基类引用；

（3）可以用派生类对象地址为基类指针赋值。

例如，对于下图所示的类的继承关系：



- 以下程序能正常编译和运行

```
Student student;
Person person,*pPerson;
person = student;              // 用派生类对象为基类对象赋值
pPerson = &student;            // 用派生类对象地址为基类指针赋值
Person &rPerson = student;// 用派生类对象初始化基类引用
```

提示：

- 需要派生类对象的地方，不能以基类对象来替代。就像“猴子是动物”是正确的，但“动物是猴子”就错了。
- 用派生类对象替代基类对象进行赋值操作后，通过基类对象、基类对象引用和基类指针只能访问派生类从基类继承的成员。例如：

```
// 正确：使用基类对象能够访问派生类从基类继承的的成员
cout<<person.GetName()<<endl;
// 错误：使用基类对象不能访问派生类中新定义的成员
cout<<person.GetMajor()<<endl;
```