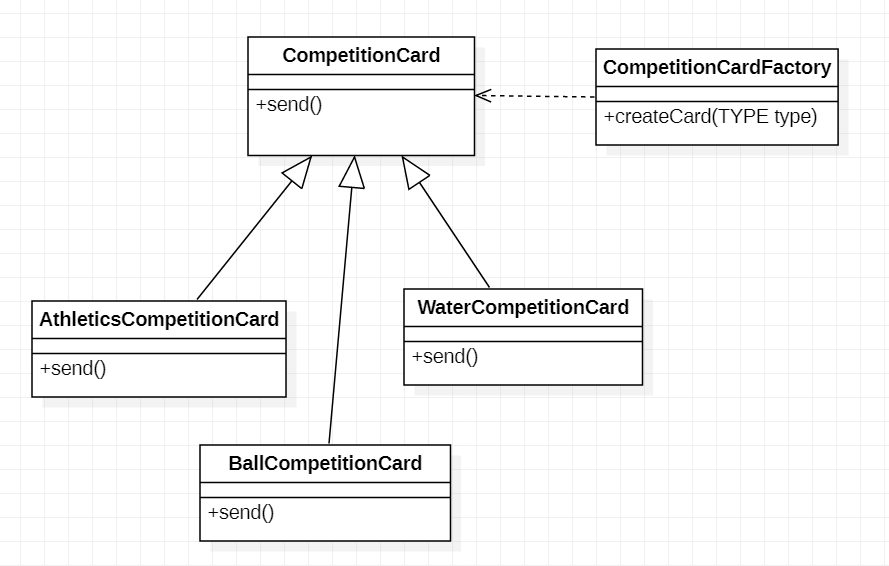
1. 设计模式StaticFactoryMethod
   1. 实现API描述

如类图所示，在父类CompetitionCard中，定义了一个分发参赛证即展示参赛证类型的抽象函数，在继承它的三个实体子类中，每个类都对此方法进行重写，工厂类通过逻辑判断返回其对应生产的参赛证子类的实例。这样就实现简单工厂模式中要求的统一通过工厂类根据必要的逻辑判断生产不同的具体类的对象，创建对象的逻辑在工厂中，外界直接通过工厂获取需要的对象，有利于整个软件体系结构的优化。测试时只需创建工厂子类的实例然后调用其createCard(TYPE type)的API即可获取对应参赛证实例。

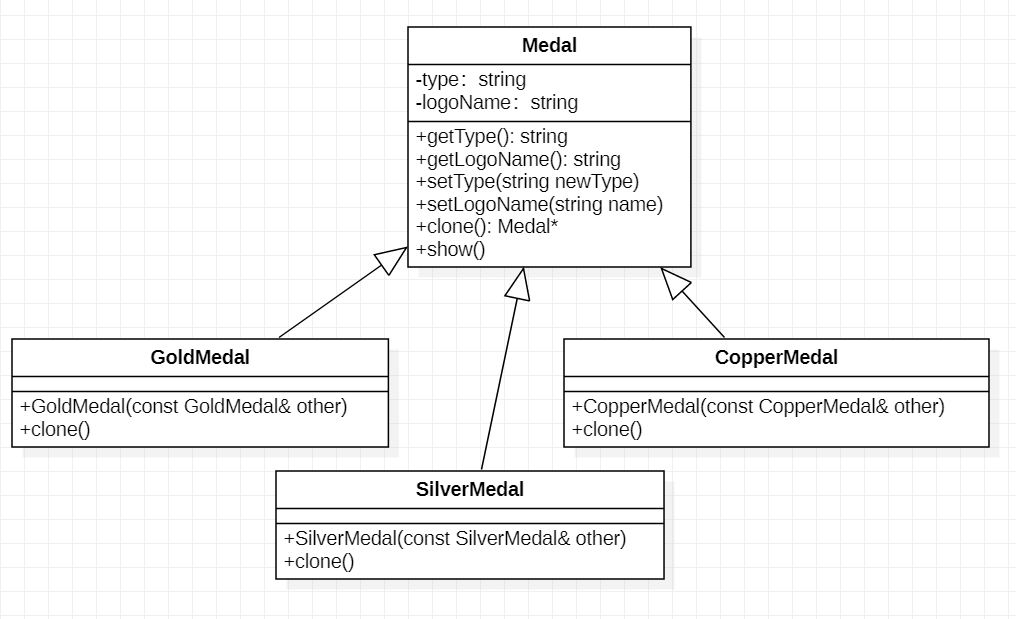
* 1. 类图



1. 设计模式Prototype
   1. 实现API描述

如类图所示，建立基类Medal，设置它的相关属性：type表示奖牌的类别、logoName表示奖牌上印有的logo，定义了接口函数，setType：设置奖牌的类别、getType：获取奖牌的类别、setLogoName：设置奖牌的logo、getLogoName：获取奖牌的logo，show：展示次奖牌的信息，还有一个最重要的clone函数，可通过调用clone函数快速生成一个和自身一模一样的实例。还有三个子类代表着三种奖牌，在子类中重写了构造函数和clone函数以实现克隆自身的功能。

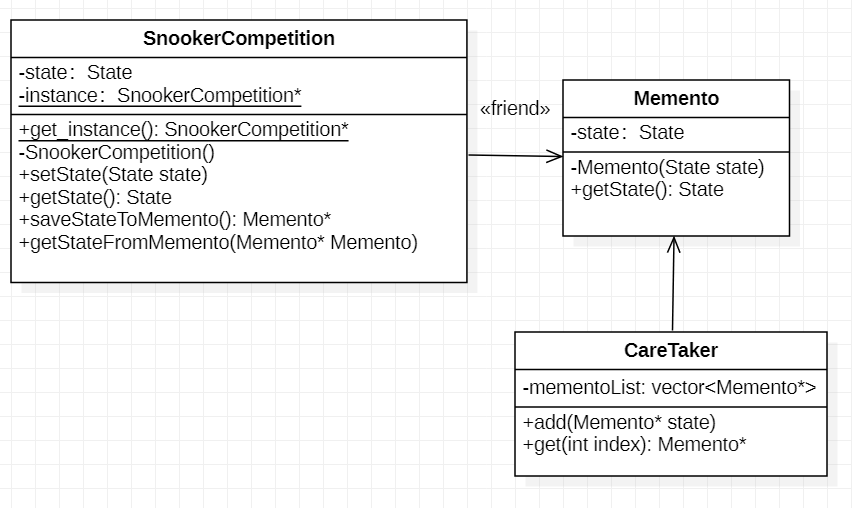
* 1. 类图



1. 设计模式Singleton
   1. 实现API描述

如类图所示，将SnookerCompetition的构造函数和自身的实例设置为私有，通过接口函数get\_instance,当系统没有这个实例时创建一个实例并将其返回、如果有则直接返回该实例。

* 1. 类图



1. 设计模式Memento
   1. 实现API描述

如类图所示，首先定义了SnookerCompetition类描述比赛当前的状态，设计接口函数setState：设置比赛状态、getState：获取比赛当前状态。设计一个Memento类来保存比赛的每个状态设置成备忘录，设计CareTaker类来储存每个备忘录。SnookerCompetition类通过saveStateToMemento()函数创建一个保存当前比赛状态的备忘录对象，可通过调用CareTaker类的add函数将备忘录加入备忘录列表。当想要获取之前的比赛状态时，通过index从备忘录列表中拿到对应的备忘录，比赛类调用getStateFromMemento函数通过Memento类的getState接口函数获取该备忘录中记录的比赛状态并将其设置成此时比赛的状态，由此可实现记录比赛的状态、也可以将比赛还原到之前的某个状态。

* 1. 类图

