实验四报告

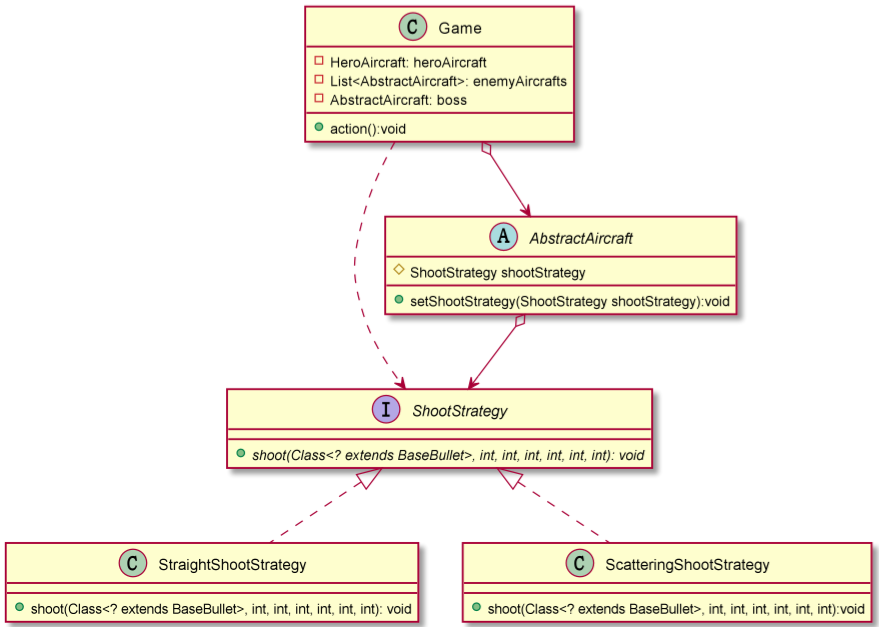
# 策略模式

1. 应用场景分析

在处理英雄机和敌机的射击方式时，可以用到策略模式。  
若不使用策略模式，则不同的射击方式之间难以实现灵活的切换（可能需要用到多重条件选择语句if else），也难以扩展新的射击方式。

使用了策略模式后，飞机类本身不必再关心射击方式的具体实现，只要在合适的时机去选择合适的射击方式即可，并且可以自由切换；在扩展新的射击方法时，不需要改动原有代码，符合开闭原则，符合面向对象编程的思想。

1. 解决方案



新建一个接口ShootStrategy，代表射击策略，再创建两种实现类，分别代表直射和散射。英雄机、精英敌机、boss机都可以作为上下文，持有ShootStrategy类型的成员变量，并且对外提供修改策略的方法，实现对射击方式的自由切换。

图中用AbstractAircraft抽象类作为上下文，持有ShootStrategy类型的成员变量，并且对外提供修改策略的方法。Game类中聚合了AbstractAircraft，其中的方法有对飞机射击方法的调用，间接使用到了策略。

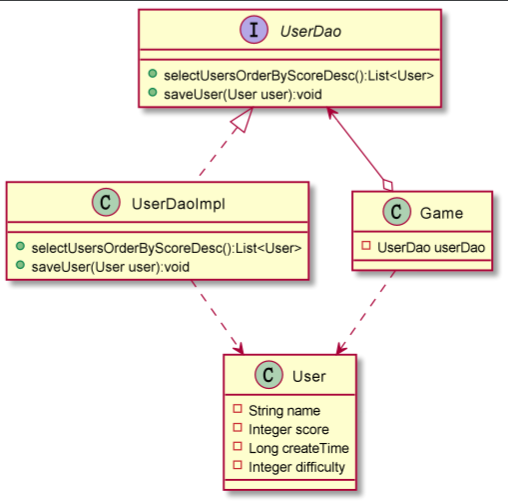
# 数据访问对象模式

1. 应用场景分析

在游戏结束后，要记录用户名、用户分数、时间等信息时可以，可以使用到数据访问对象模式。

使用该模式，可以将读写数据的功能解耦出来，符合单一职责原则。当需要增加对数据的其他操作时，也易于扩展，不必修改之前定义好的操作，符合OCP原则。同时为数据建立实体类也体现了面向对象编程的思想。

1. 解决方案



创建实体类User，将用户相关信息定义为属性，封装成该对象，便于存储和读取。

在UserDao接口中定义了对User实体类对象封装的数据的操作，包括按得分高低查询用户、存储用户等。

再创建UserDaoImpl类实现该接口，即实现其中定义的数据操作。

Game类持有UserDao类型成员变量，调用其封装好的方法，实现对数据的多种操作。