Nobles battle with each other. will lose if they don't have warriors to defend them

Warriors hire out to a noble and fight in his behalf.

* Warriors start out with a specified strength.
* A battle between nobles is won by combined strengths of all its warriors.
* A battle is to the death. The losing noble dies as does his warriors.
* All his men lose a portion of their strength equal to the ratio of the enemy army's combined

**Hiring and Firing**

* However it is possible that an attempt to hire or fire may fail. Naturally the methods should not "fail silently". Instead, they will return true or false
* only be employed by one Noble at a time and *cannot* be hired away if he is already employed.
* As noted below, Nobles who are dead can neither hire nor fire anyone. (Note this will *implicitly* prevent dead Warriors from being hired.)
* When a warrior is fired, he is no longer part of the army of the Noble that hired him. He is then free to be hired by another Noble.
  + How do you remove something from a vector.
  + While there are techniques that make use of iterators, we have not yet discussed iterators so you will not use them here. (As a heads up, if you see a technique that requires you to call a vector's begin() method, that is using iterators. Don't use it.)
  + While it may seem a slight burden, certainly it does not require more than a simple loop to remove an item from a vector. No do not do something silly like create a whole new vector.
  + Soon we will cover iterators and then you will be freed from these constraints.

**Death**

* People die when they lose a battle, whether they are a Nobles or Warriors.
* Nobles who are dead are in no position to hire or fire anyone. Any attempt by a dead Lord to hire someone will simply fail and the Warrior will remain unhired.
* Nobles can declare battle even though they are dead.
* Note that when a Noble is created he does not have any strength. At the same time he is obviously alive. So lack of strength and being dead are clearly *not* equivalent.

**Programming Constraints**

* The battle method will announce who is battling whom, and the result (as shown in the example output).
  + If one or both of the nobles is already dead, just report that.  
    The "winner" doesn't win. His warriors don't use up any strength.
  + Look at the output for the sample test program to see what you should be displaying.
* A noble's army is a **vector of pointers** to warriors. Warriors will be ordered in the army by the order in which they were *hired*. That affects how you remove a Warrior that gets fired.