Homework 4

Focus

- Classes
- Association
- As always: good coding

Problem:

We will [continue to] model a game of medieval times. Our world is filled with not only warriors but also nobles. Nobles don't have much to do except do battle with each other. (We'll leave the feasting and other entertainments for add-ons.) Warriors don't have much to do except hire out to a noble and fight in his behalf. Of course the nobles are pretty wimpy themselves and will lose if they don't have warriors to defend them. How does all this work?

- Warriors start out with a specified strength.
- A battle between nobles is won by the noble who commands the stronger army.
- The army's strength is simply the combined strengths of all its warriors.
- A battle is to the death. The losing noble dies as does his warriors.
- The winner does not usually walk away unscarred. All his men lose a portion of their strength equal to the ratio of the enemy army's combined strenth to their army's. If the losing army had a combined strength that was 1/4 the size of the winning army's, then each soldier in the winning army will have their own strength reduced by 1/4.

Hiring and Firing

- Warriors are hired and fired by Nobles. Lack of adequate labor laws, have left the Warriors without the ability to quit, nor even to have a say on whether or not a Noble can hire them.
- 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 depending on whether they succeed or not.

- A Warrior can 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 should not use those. (As a heads up, if you see a technique that requires you to call a vector's begin() method, that is using iterators.)
    - 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. Patience, please.

Death

It's a sad topic, but one we do have to address.

- 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.
- However curiously, as has been seen before, 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.

A test program and output are shown below. Note that the output shown is what you are expected to generate. Pardon us, we don't like limiting your creativity, but having your output consistent with ours makes grading a bit easier.

Programming Constraints

What would a homework assignment be without unnecessary and unreasonable constraints?

Your classes must satisfy the following:

- 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 anything for kicking around a dead guy. And 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 by the order in which they were *hired*.

And some things to make life easier

Just in case you are confused, let me point out that this problem does **not** involve the use of the heap. That means your program will **not** make use of `new` or `delete`.

Turn in

Hand in the file named hw04.cpp.

Sample Test Code

```
int main() {
    Noble art("King Arthur");
    Noble lance("Lancelot du Lac");
    Noble jim("Jim");
    Noble linus("Linus Torvalds");
    Noble billie("Bill Gates");

    Warrior cheetah("Tarzan", 10);
    Warrior wizard("Merlin", 15);
    Warrior theGovernator("Conan", 12);
    Warrior nimoy("Spock", 15);
    Warrior lawless("Xena", 20);
    Warrior mrGreen("Hulk", 8);
    Warrior dylan("Hercules", 3);

    jim.hire(nimoy);
    lance.hire(theGovernator);
    art.hire(wizard);
    lance.hire(dylan);
```

```
     linus.hire(lawless);
     billie.hire(mrGreen);
     art.hire(cheetah);

     jim.display();
     lance.display();
     art.display();
     linus.display();
     billie.display();

     art.fire(cheetah);
     art.display();

     art.battle(lance);
     jim.battle(lance);
     linus.battle(billie);
     billie.battle(lance);
}
```

```
Output for Sample Code

Jim has an army of 1
        Spock: 15
Lancelot du Lac has an army of 2
        Conan: 12
        Hercules: 3
King Arthur has an army of 2
        Merlin: 15
        Tarzan: 10
Linus Torvalds has an army of 1
        Xena: 20
Bill Gates has an army of 1
        Hulk: 8
Tarzan, you're fired! -- King Arthur
King Arthur has an army of 1
        Merlin: 15
King Arthur battles Lancelot du Lac
Mutual Annihalation: King Arthur and Lancelot du Lac die at 
each other's hands
Jim battles Lancelot du Lac
He's dead Jim
Linus Torvalds battles Bill Gates
Linus Torvalds defeats Bill Gates
```

```
Bill Gates battles Lancelot du Lac
Oh, NO!  They're both dead!  Yuck!
```