**Grade: 15/20**

**Homework 4**

**Focus**

* Dynamic memory

**Problem:**

Building on the last two assignments, we will use the classes from hw03 together with the idea of reading a file of commands as we did in hw02. The classes from hw03 should not need any changes other than perhaps a getName method for Noble.

One key difference will be that each time a warrior or a noble is defined, we will create it on the heap. We will keep track of the warriors in a vector of pointers to warriors. And similarly, we will keep track of nobles using a vector of pointers to nobles.

The input file will be named "nobleWarriors.txt".

**Commands**

* Noble. Create a Noble on the heap.
* Warrior. Create a Warrior on the heap.
* Hire. Call the Noble's hire method.
* Fire. Call the Noble's fire method.
* Battle. Call the Noble's battle method.
* Status. The status command shows the nobles, together with their armies, in a similar way that the Noble display method did in hw03. In addition, it will show the warriors who do not have an employer, either because they were fired or never hired.
* Clear. Clear out all the nobles and warriors that were created.

Our application is going to rely on each Noble having a unique name and each Warrior having a unique name. Otherwise, how would we be sure who we were hiring (or firing). Note that this is not a requirement of the Noble and Warrior classes themselves, just of this particular use of them.

Handle errors! If a Noble with a given name already exists you should generate an error message, if we try to create another. If a Noble tries to hire a Warrior who doesn't exist, then similarly, you should generate an error message. We have not specified the format of these messages, so we'll leave that up to you. Make sure your code addresses any such possible errors. You do *not* have to worry about the format of the messages, i.e. we promise not to provide input with invalid command names or the wrong number of tokens.

Example input file:

Noble King\_Arthur  
Noble Lancelot\_du\_Lac  
Noble Jim  
Noble Linus\_Torvalds  
Noble Bill\_Gates  
Warrior Tarzan 10  
Warrior Merlin 15  
Warrior Conan 12  
Warrior Spock 15  
Warrior Xena 20  
Warrior Hulk 8  
Warrior Hercules 3  
Hire Jim Spock  
Hire Lancelot\_du\_Lac Conan  
Hire King\_Arthur Merlin  
Hire Lancelot\_du\_Lac Hercules  
Hire Linus\_Torvalds Xena  
Hire Bill\_Gates Hulk  
Hire King\_Arthur Tarzan  
Status  
Fire King\_Arthur Tarzan  
Status  
Battle King\_Arthur Lancelot\_du\_Lac  
Battle Jim Lancelot\_du\_Lac  
Battle Linus\_Torvalds Bill\_Gates  
Battle Bill\_Gates Lancelot\_du\_Lac  
Status  
Clear  
Status

**Example output**

Status

======

Nobles:

King\_Arthur has an army of 2

Merlin: 15

Tarzan: 10

Lancelot\_du\_Lac has an army of 2

Conan: 12

Hercules: 3

Jim has an army of 1

Spock: 15

Linus\_Torvalds has an army of 1

Xena: 20

Bill\_Gates has an army of 1

Hulk: 8

Unemployed Warriors:

NONE

You don't work for me anymore Tarzan! -- King\_Arthur.

Status

======

Nobles:

King\_Arthur has an army of 1

Merlin: 15

Lancelot\_du\_Lac has an army of 2

Conan: 12

Hercules: 3

Jim has an army of 1

Spock: 15

Linus\_Torvalds has an army of 1

Xena: 20

Bill\_Gates has an army of 1

Hulk: 8

Unemployed Warriors:

Tarzan: 10

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!

Status

======

Nobles:

King\_Arthur has an army of 1

Merlin: 0

Lancelot\_du\_Lac has an army of 2

Conan: 0

Hercules: 0

Jim has an army of 1

Spock: 15

Linus\_Torvalds has an army of 1

Xena: 12

Bill\_Gates has an army of 1

Hulk: 0

Unemployed Warriors:

Tarzan: 10

Status

======

Nobles:

NONE

Unemployed Warriors:

NONE

**Submit**

Submit your program as hw04.cpp.

Grader comments:

Warning: In void hire, check if the warrior is currently not hired and only then hire him/her.

The member variable strength should be of type double.

Accessor methods should be const.

totalStrOfThisNoble and totalStrOfOtherNoble should be of type double.

The clear command should loop through both vectors, deleting each entry. Afterwards, it should clear the vectors.