

[Content](#) [Announcements](#) [Assignments](#) [Discussions](#) [Quizzes](#) [Grades](#) [Zoom](#) [More Tools](#) ▼ [Course Reports](#) ▼ [Help](#) ▼[Assignments](#) > [hw03](#)

hw03

▼ Hide Assignment Information

Instructions

Homework 3

Focus:

- Object-oriented programming
 - Data Hiding
 - Encapsulation
 - Delegation
 - Output operator Overloading

Problem:

We will expand our Warrior a little. Each Warrior will have a weapon. He is "born" with it, i.e. the weapon is created together with the warrior. It can only be accessed by him. It provides him with his strength. In battle, weapons lose their edge and weaken. When a Warrior's weapon loses all of its strength, the Warrior himself dies.

Implementation

- Remember that we are using ***data hiding***. Therefore, **every** field, aka member variable, ***must*** be private.
- What are the *types* of things in the problem? We will need a class for each type.
- What do the things / types *do*? These **"behaviors"** **should be represented as *methods***.
- Weapons have both a name and a strength. The weapon is created together with the Warrior and **must not** be accessed by anyone else.
- Note that the input file changed a little, compared to the previous assignment. When a Warrior is created, instead of simply specifying his name and strength, the Warrior command specifies the Warrior's name as well as his Weapon's name and *its* strength.
- The Status report is also modified to show the name of the Warrior's Weapon.
- ***No one*** can access a warrior's weapon **except the warrior himself**. But the weapon is what actually holds the warrior's strength. How does this effect the programming? Any time the code needs to know or change the warrior's strength, **the warrior who owns the weapon then "asks" the weapon** what the strength is or tells the weapon that the strength needs to be changed. This represents the idea of ***delegation***. We will see this concept frequently, where one object requests that another object do some task.
- It is in fact unnecessary for any code other than a Warrior to even know about the Weapon. We will enforce this by *nesting* the definition of the Weapon class inside the Warrior class. To make sure that no code other than Warrior's makes use of Weapon, we need to make the class private.
- The name of the input file will be "warriors.txt".
- One last implementation detail. to display the information about an object, whether it is a warrior or a weapon, we will use that object's output operator.

Input

Our sample input file might now look like:

```
Warrior Jim Glamdring 42
Warrior Lancelot Naegling 15
Warrior Arthur Excalibur 15
Warrior Torvalds Narsil 20
Warrior Gates Orcrist 8
Status
Battle Arthur Lancelot
Battle Jim Lancelot
Battle Torvalds Gates
Battle Gates Lancelot
Status
```

Output

The corresponding output would be:

```
There are: 5 warriors
Warrior: Jim, weapon: Glamdring, 42
Warrior: Lancelot, weapon: Naegling, 15
Warrior: Arthur, weapon: Excalibur, 15
Warrior: Torvalds, weapon: Narsil, 20
Warrior: Gates, weapon: Orcrist, 8
Arthur battles Lancelot
Mutual Annihilation: Arthur and Lancelot die at each other's hands
Jim battles Lancelot
He's dead, Jim
Torvalds battles Gates
Torvalds defeats Gates
Gates battles Lancelot
Oh, NO! They're both dead! Yuck!
There are: 5 warriors
Warrior: Jim, weapon: Glamdring, 42
Warrior: Lancelot, weapon: Naegling, 0
Warrior: Arthur, weapon: Excalibur, 0
Warrior: Torvalds, weapon: Narsil, 12
Warrior: Gates, weapon: Orcrist, 0
```

Notes

Remember this program is *significantly* different than the prior assignment, which did not use any OOP techniques. Some functions that existed in the previous assignment may now be methods. And displaying information about an object will now be done through its class's output operator.

Attachment

For your convenience, a file matching the above example input has been attached.

Turn in

Hand in a single cpp file, hw03.cpp, containing your program.

Due Date

Oct 5, 2022 11:59 PM

Attachments

 [warriors.txt](#) (234 Bytes)

Download All Files

Submit Assignment

Files to submit *

(0) file(s) to submit

After uploading, you must click **Submit** to complete the submission.

Add a File

Record Audio

Record Video

Comments

Submit

Cancel