ITEC 120: Principles of Computer Science I

Homework 7 – Army battle simulator

>>>>>>>> Due Date: Friday, March 16th at 10:00PM via D2L <<<<<<<

In doing this homework, remember to abide by the RU Honor Code.

Problem 1
30 points

World of Wyverns Inc. has decided that that they are going to start offering the ability for players to control thousands of warriors in a simulated battle. A new competitor is producing one on one battle simulators that are similar to your current products. Because of the added competition, your bosses have decided to write a game allowing players to unleash a horde of warriors against computerized enemies. Your job will be to write the simulation engine that will determine which side will win.

So far you have battled one monster against another until one wins or loses. With this simulator you be simulating thousands of one on one battles at the same time. There are two sides in the battle. Each side has the same number of warriors, but each side may have differing hp, armor, and damage dealing capabilities. In order for this to be possible, each side will have arrays of the same size that hold information about the warriors in the battle.

The arrays for each army are the HP of each warrior, the damage they deal, and the percentage of damage that their armor absorbs. For the first version of the simulator the armies will be the same size and each warrior will fight only one other warrior.

The simulator will be interactive and will behave in a manner similar to previous simulators. The commands that the simulator will respond to are help, size, hp, armor, damage, quit and side. Help will print out the possible commands and quit will stop the execution of the program.

Commands:
size – Creates the armies for each side. Use this variable to create arrays that can hold the HP, damage, and armor for warriors in each army. This command must be used before the hp, damage, armor, and battle commands can be used.

side – Choose which side of the army that you are sending commands to. By default, you should work on side 1. This command prompts the user for an integer (1 or 2), and then sets an internal flag that affects the hp, armor, and damage commands for the specified side.

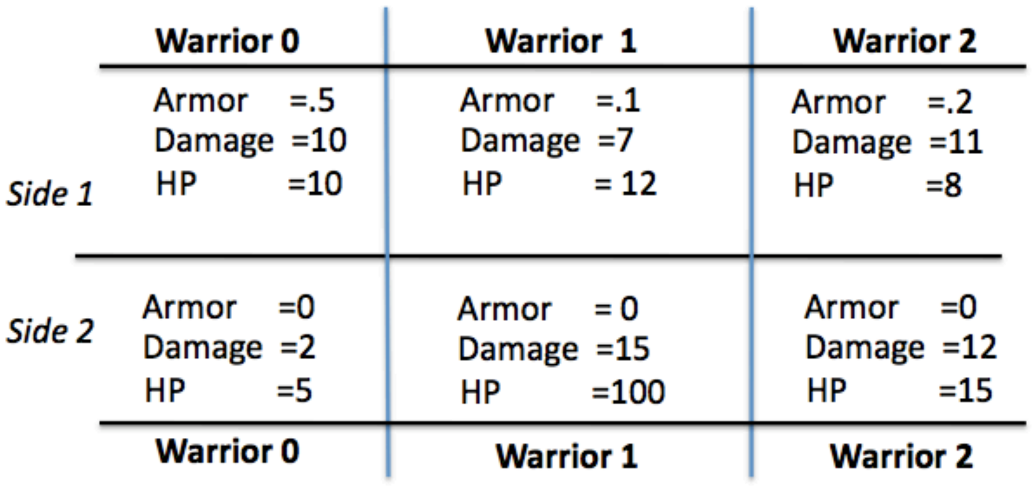
hp – Sets the HP for each warrior in the selected army to be between a lower number and a higher number. Prompt the user for the lower and upper bounds of HP and then randomly assign each warrior on the selected side a HP value that is between the bounds the user entered *Hint:* You can accomplish this with code similar to: `lowHP+rand.nextInt(highHP-lowHP).`

damage – Sets the damage for each warrior in the selected army to be between a lower number and a higher number. Prompt the user for the lower and upper bounds of damage and then randomly assign each warrior on the selected side a damage value that is within the bounds the user entered.

armor – Sets the percent of damage reduction for each warrior in the selected army to be between a lower number and a higher number. Prompt the user for the lower and upper bounds of damage reduction and then randomly assign each warrior on the currently selected side a damage reduction percentage within the entered values.

battle – Create copies of the HP arrays of each warrior (so the battle is not permanent), then prompt the user for how many times the warriors should hit each other. For each time the warriors hit each other, the HP of each warrior should be reduced by the damage the warrior on the other side deals minus the amount that their armor reduces (exactly the same as previous assignments). Note: in order to accomplish this, each warrior fights the warrior represented by the same array index in the opposite army. The winner of the battle is the side that lost the least number of warriors. If the number of soldiers lost is equal, then the battle is a draw. You should display the number of warriors lost on each side after the battle is finished.

Sample array visualization
Arrays representing the warriors in a situation where the army size is 3.



		Warrior 0	Warrior 1	Warrior 2
Side 1	Armor	=.5	=.1	=.2
	Damage	=10	=7	=11
	HP	=10	= 12	=8
Side 2	Armor	=0	= 0	=0
	Damage	=2	=15	=12
	HP	=5	=100	=15
		Warrior 0	Warrior 1	Warrior 2

Input:
Input will consist of commands sent to your program. Make sure you tell the user what they should be inputting into the program each time they enter a command.

Output:
You should display a prompt on each line informing the user of what they should type.

Sample usage scenario:
```
Command>help
Commands are:  help/size/side/armor/damage/hp/battle/quit
Command>size
Enter army size>300
Army size is 300
Command>damage
Lower damage>10
Higher damage>20
Command>hp
Lower hp>10
Higher hp>20
Command>side
Enter side>2
Command>damage
Lower damage>10
Higher damage>20
Command>hp
Lower hp>100
Higher hp>200
Command>battle
Number of turns>2
=======Battle results are in!==========
Side 1 lost: 300 warriors
Side 2 lost: 0 warriors
Army two wins
Command>battle
Number of turns>1
=======Battle results are in!==========
Side 1 lost: 175 warriors
Side 2 lost: 0 warriors
Army two wins
Command>quit
```

Testing note: your program must allow for a file to be redirected as input to the program. If it does not, then you will receive a 0 on this program.

Constraints:
Your main method should read in commands until the user types quit. You must use at least two functions in your solution. You must also submit a test case that is similar but

not identical to the previous usage scenario that demonstrates that you have tested your program thoroughly.

No fields, instance, or class variables may be used in this program. If you don't know what they are, don't worry about it. The purpose for this rule is to make sure you must pass information through the program with parameters.

The reference solution for this project is 149 lines of code without comments. Feel free to use less or more code in your program.

Submission requirements:
You must submit the .java file containing your program to Desire 2 Learn under the Homework #7 assignment. If your submitted file does not compile, it will receive a 0. You can demo the homework the next school day after it is due, or it will be graded automatically. If you choose not to demo your homework you cannot contest the grade you receive.

Grading Rubric
15 Points - Do the commands work as expected?
7 Points - Are parallel arrays used properly?
5 Points - Is the program commented using inline and javadoc comments?
3 Points – Was a proper test case submitted along with the source code?