

ITP 165: Introduction to C++ Programming

Homework 11: Animal Training 2.0

Due: 11:59pm, November 20th, 2015

Goal

This assignment will now let you train and level up your Animals while gaining further practice with dynamic memory allocation, as well as an introduction to vectors. The complete Animal class is given to you in this assignment and it is up to you to implement the user interaction. Feel free to modify and expand the Animal class to improve upon the fight interaction either via the math or the output text.

Setup

- Create a new project in Visual Studio or Xcode called **homework11** and add three new files named **trainer2.cpp**, **Animal.h**, and **Animal.cpp**. The **Animal.h** and **Animal.cpp** files have already been given to you with your assignment file.
- Each file must begin with the following (replace the name and email with your actual information):

```
// Name
// ITP 165, Fall 2015
// Homework 11
// USC email
```

Part 1: RandOpponent function

- In your **trainer2.cpp** file, before your main function – you will be creating a function that returns a pointer to a dynamically allocated Animal object with random starting values.
- In this **RandOpponent** function, create a new Animal on the heap with:
 - a health of 50,
 - a random level between 1 and 5,
 - a random power stat between 20 and 40
 - a random armour stat between 20 and 40
 - 0 experience
 - Any name you want that designates that this is an opponent.
- Return the pointer to this Animal object

Part 2: User interaction – main function

- You will need to begin your main function with seeding your random number generator like we have done before with **srand**. Next you will also need one pointer for your user's animal as well as a vector of Animal pointers to hold the opponents for this program. You may also need other temporary variables for user input or calculations.
- Start off your program by creating a new Animal on the heap with a name input by the user.
- Next you will display a menu of options for your user, this time with only 4 options:
 - Quit
 - This case handles cleaning up the memory of the program as well as telling the user the program is ending.

- Print Stats
 - This case will call the `PrintStats` function on the user's animal. This function is now a part of the Animal class.
- Heal Animal
 - This case will increase the user Animal's current health with a random number between 10 and 50. Tell the user how much the animal was healed by.
- Train Animal
 - This case will have three parts. It will require using two new functions that have been created in the Animal class: `isAwake` and `Fight`.
 1. Fill the vector with a random number of opponents (between 1 and 3) created by the `RandOpponent` function from Part 1.
 2. For each opponent in this vector, if the user's animal is still awake –
 - Announce each animal's stats before the start of each round
 - The fight between the user animal and the current opponent begins when the user has input ANY character via the console
 3. After all fights are complete, be sure to clean up your vector and the objects created inside it.
 - Be sure to have a default case to catch invalid input
- Loop over the menu options until the user has entered the Quit option.

Full Sample Output

Below is sample output for a full run-through of the program. Your output should resemble the following – but doesn't need to match it exactly (user input is in **red**).

Welcome to the Training Arena!

Animal name? **Test Animal**

MENU OPTIONS:

0. Quit

1. Print Stats

2. Heal Animal

3. Train Animal

What would you like to do? **1**

Name: Test Animal

Level: 1

Health: 100

Power: 50

Armour: 50

XP to next level: 100

MENU OPTIONS:

0. Quit

1. Print Stats

2. Heal Animal

3. Train Animal

What would you like to do? **3**

Let's get ready to train your animal!

3 opponents generated!

Ready for Round 1? **y**

***** Round 1 *****

Name: Opponent
Level: 3
Health: 50
Power: 43
Armour: 46
XP to next level: 100

VERSUS

Name: Test Animal
Level: 1
Health: 100
Power: 50
Armour: 50
XP to next level: 100

Test Animal hits for 12
Opponent hits for 11
Test Animal hits for 13
Opponent hits for 11
Test Animal hits for 15

Test Animal wins and gains 30 experience!
Ready for Round 2? y

***** Round 2 *****

Name: Opponent
Level: 3
Health: 50
Power: 56
Armour: 52
XP to next level: 100

VERSUS

Name: Test Animal
Level: 1
Health: 78
Power: 50
Armour: 50
XP to next level: 70

Test Animal hits for 13
Opponent hits for 15
Test Animal hits for 13
Opponent hits for 13
Test Animal hits for 14

Test Animal wins and gains 30 experience!
Ready for Round 3? y

***** Round 3 *****

Name: Opponent
Level: 3

Health: 50
Power: 54
Armour: 55
XP to next level: 100

VERSUS

Name: Test Animal
Level: 1
Health: 50
Power: 50
Armour: 50
XP to next level: 40

Test Animal hits for 13
Opponent hits for 13
Test Animal hits for 11
Opponent hits for 14
Test Animal hits for 11

Test Animal wins and gains 30 experience!

MENU OPTIONS:

0. Quit
1. Print Stats
2. Heal Animal
3. Train Animal
What would you like to do? 2

Healed by 40!

MENU OPTIONS:

0. Quit
1. Print Stats
2. Heal Animal
3. Train Animal
What would you like to do? 1

Name: Test Animal
Level: 1
Health: 63
Power: 50
Armour: 50
XP to next level: 10

MENU OPTIONS:

0. Quit
1. Print Stats
2. Heal Animal
3. Train Animal
What would you like to do? 0

Quitting program...

Thanks for playing!
Press any key to continue . . .

A Note on Style

Be sure to comment your code. Also, remember to comment out any extra debug statements.

As we discussed in lecture, it is extremely important that your code is properly indented as it greatly adds to readability. Because of this, if you submit a code file that is not reasonably indented, you will have points deducted.

Likewise, you will lose points if your variable names are not meaningful. Make sure you use variable names that correspond to what you are actually storing in the variables.

Deliverables

1. A compressed folder containing only the **trainer2.cpp**, **Animal.h**, and **Animal.cpp** files, named **HW11**. This can be done by:
 - a. WINDOWS:
 - i. Select all your files
 - ii. Right click
 - iii. Send to ->
 - iv. Compressed (zipped) folder
 - v. Rename this zipped folder to HW11
 - vi. Submit this zipped folder through Blackboard
 - b. OSX:
 - i. Select all your files
 - ii. Right click
 - iii. Compress 3 items
 - iv. Rename this Archive.zip to HW11
 - v. Submit this zipped folder through Blackboard

Grading

Item	Points
Part 1: Random opponent function	20
Part 2: User interaction – main function	10
Part 3: Fight case	10
Total*	40

* Points will be deducted for poor code style, or improper submission.