CMPT 213 Assignment 1

Due: Jan 20, 11:59:59pm.

Overview

A Tokimon is a rare creature found in the remote areas of Korea. For the most part, they resemble rabbits and the only form of communication they have with us is the ability to say their own name. Tokimons come in all shapes and sizes, have special abilities, and tend to fight with each other on occasion. Each Tokimon could have the ability to fly, fight, throw fire, spray water, electrify, and freeze other Tokimons; their strength is measured by an integer between 0 and 100. Because they are such primitive creatures but have such extraordinary abilities, they must be tracked.

Technical Requirements : Create a new project.
You must (at least) have the following classes:

- A class to hold Tokimon information: name(String), size(double), ability/type(String), strength(integer); you are free to add more :). The name may be more than one word (i.e. "Bobby Chan"). The class must properly implement the toString() method as discussed in class.
- A helper class for displaying a menu prompt to the user.
    - Pass the constructor the menu's title (String), and menu's options (array of Strings). The options should not include numbers ([give the name]).
    - Method which prints the menu (title and numbered menu options) to the screen. Automatically place a rectangle of *'s around the menu's title, sizing the rectangle to the length of the title. Automatically number the options starting at 1.
    - Method to prompt the user to enter a selection from the menu. When there's an error (invalid value) the method re-ask the user for a value. You may assume user always enters correct *type* of data: when asked for an int, it is OK if the program crashes when the user enters a non-int such as 'A'.
- A class for the main application. Contains a main() method which uses the menu and Tokimon classes to implement the application. Create an ArrayList of Tokimons to hold the set of Tokimons the user enters. Be careful not to have much duplicate code in your application! Use functions.

Your code must conform to the programming style guide (available in the resources section of the course website). All classes must have class-level JavaDoc comments describing the purpose of the class.

- Text Interface: your app should have the following general options to choose from:
    1. List all Tokimons including the name, height, ability and strength
    2. Add a new Tokimon. Prompt the user for a name, type, height, and ability. By default, new Tokimons will have strength 0.
    3. Delete a Tokimon. First list all current Tokimons, allow the user to choose the Tokimon to delete (or 0 to cancel). Entering an invalid number should be handled by the program, but invalid datatypes such as 'A' need not be handled.
    4. Augment strength. First list all current Tokimons, allow the user to choose the Tokimon to increase the strength, then prompt for the amount. Once again, you do not need to handle invalid datatypes.
    5. Display the toString() on each Tokimon in the system.
    6. Exit the program.

    Here is some sample output, you do not have to match the exact layout (it's just a suggestion):

```
*****************************************
* Tokimon Tracker by Bobby Chan sn 5555555 *
*****************************************

*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 2
Enter Tokimon's name:   Toki chu
Enter Tokimon's type:   Fire
Enter Tokimon's size:   .5

*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 1

*********************
* List of Tokimons: *
*********************
1. Toki chu, 0.5m, Fire ability, 0 strength

*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 2
Enter Tokimon's name:   Squir To
Enter Tokimon's type:   Water
Enter Tokimon's size:   3.1

*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 1

*********************
* List of Tokimons: *
*********************
1. Toki chu, 0.5m, Fire ability, 0 strength
2. Squir To, 3.1m, Water abiity, 0 strength

*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 5
All Tokimon objects:
1. ca.sfu.cmpt213.as1.Tokimon[Name:Toki chu, Strength:0, Height:0.5, Ability:Fire]
2. ca.sfu.cmpt213.as1.Tokimon[Name:Squir To, Strength:0, Height:3.1, Ability:Water]
```

```
*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 4

*********************
* List of Tokimons: *
*********************
1. Toki chu, 0.5m, Fire ability, 0 strength
2. Squir To, 3.1m, Water abiity, 0 strength
(Enter 0 to cancel)
> 1
By how much?:     3
Toki chu now has strength 3!

*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 4

*********************
* List of Tokimons: *
*********************
1. Toki chu, 0.5m, Fire ability, 0 strength
2. Squir To, 3.1m, Water abiity, 0 strength
(Enter 0 to cancel)
> 0

*************
* Main Menu *
*************
1. List Tokimons
2. Add a new Tokimon
3. Remove a Tokimon
4. Change Tokimon strength
5. DEBUG: Dump objects (toString)
6. Exit
> 6
BYE!
```

You may assume any requirements that are not explicitly stated in this description (within reason). If you're not sure of a requirement, please see the TA or myself.

Marking Scheme:

[20] Basic Functionality

- [5] Robust keyboard input: when given a number out of range it asks for retry.
  (Not testing typing text, like "A", instead of a number).
- [5] Able to add and remove Tokimons.
- [5] Able to augment strengths correctly.
- [5] Good listing of Tokimons, and debug listing (using toString).

[10] Code Quality and Style Guide

- * Reasonable object oriented structure
- * JavaDoc comment on each class (not needed on methods/fields)
- * Correct indentation, brackets, spacing (use IDE's reformat if needed).
- * Good intention revealing class, method, and variable names.

Submission

Submit a zip file of your project (according to the directions outlined in the assignments link of the course website) to the coursys server. https://courses.cs.sfu.ca/
Please note: all submissions are automatically checked for similarities of all other submissions on the server.

THE END