BattleBots Assignment

Welcome to the BattleBots arena! It is your job to create a unique Bot that will battle against all others in a last Bot standing death match. The base Bot class has been created for you. You need to now develop the AI, (artificial intelligence), which will make it work. You may work individually or in teams of two.

BattleBot Tasks

- Give your Bot a unique name
- Create, (or download), new images to be used in the arena
- Create unique messages and the logic for displaying them in the arena
- Create and implement an overall strategy for your Bot in terms of both movement and firing
- Document all changes and strategies in your code using Javadoc standards. Your strategy must be completely described in the class header. It should be well thought out and ambitious

Assessment Criteria

Only a small portion of your mark will be based on how your Bot performed during the actual game. Instead the majority of your mark will be based on the following criteria:

Are all required changes made? All the tasks above need to be completed.

Are the AI changes significant in nature? Your strategy must make use of at least two of the three arrays of information provided by the Arena in each call to *getMove()*.

Does your Bot generate errors? Any errors generated during execution of the game will be tracked and displayed at the end of each round. You should have 0 errors if all your code worked properly.

Is your Bot efficient? There should be no superfluous or unnecessary comparison or assignment operations. If the cumulative CPU time for any Bot exceeds 2 seconds the Bot will overheat and become disabled. That means you will need to make your AI code as simple and efficient as possible to reduce the amount of time spent in the *getMove()* method.

Rubric

Categories	Level 1 (50 - 59%)	Level 2 (60 - 69%)	Level 3 (70 - 79%)	Level 4 (80 - 100%)
Appearance and Messages - Bot displays correct name and has new images for each direction - Bot has multiple messages and outputs them at appropriate times	There is no new image for each direction and/or there are no unique messages, or the code does not allow for messages to be displayed	There are some new images for each direction but not all, and/or there are unique messages but they are not accessed properly, (i.e. not randomly)	There are new images for each direction, unique messages, and they do display to screen using simple logic, (i.e. randomly in random order)	There are new images for each direction, unique messages, and they display to the screen appropriately using advanced logic, (i.e. upon the death of a Bot, proximity of a Bot, etc.)
Efficiency - No superfluous or unnecessary comparison or assignment operations - Bot does not generate any errors	Many superfluous or unnecessary comparison or assignment operations exist and/or multiple errors are generated. Bot may overheat.	Some superfluous or unnecessary comparison or assignment operations exist and/or a few errors are generated. Bot may overheat.	No superfluous or unnecessary comparison or assignment operations exist nor any generated errors, but bot does still overheat	No superfluous or unnecessary comparison or assignment operations exist nor any generated errors , and bot does not overheat
AI Demonstrate ability to modify existing modular program code to enhance the functionality of a program.	Changes to AI are minor and do not in any way produce the results as intended in the strategy.	Changes to AI are minor, and strategy makes no use of the three arrays of information provided by the Arena in each call to *getMove()*.	Changes to AI are significant, and strategy makes use of one of the three arrays of information provided by the Arena in each call to *getMove()*.	Changes to AI are significant, and strategy makes use of at least two of the three arrays of information provided by the Arena in each call to *getMove()*.
Style Use professional conventions correctly, (e.g., naming, indenting, and fully commented according to industry standards)	Bot changes are not commented, and proper naming and structure conventions are rarely used.	Bot changes are not fully commented using javadoc standards, and proper naming and structure conventions are rarely used.	Bot changes are not fully commented using javadoc standards, or proper naming and structure conventions not always used.	Bot changes are fully commented using javadoc standards, and proper naming and structure conventions always used.