Battlebots Assignment

# Checklist

☐ Does the bot shoot?

☐ Does the bot shoot at another bot?

☐ Does the bot dodge bullets?

☐ Does the bot move to attack another bot?

☐ Does the bot have a picture?

☐ Does the bot have a name?

# Rubric

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Level 1 | Level 2 | Level 3 | Level 4 |
| **Efficiency** | Much superfluous or unnecessary code exists and/or multiple errors are generated. Bot may overheat. | Some superfluous or unnecessary code exists and/or a few errors are generated. Bot may overheat. | Very little superfluous or unnecessary code exists or rarely generates errors. | No superfluous or unnecessary code exists nor any generated errors, and bot does not overheat. |
| **Use of Arrays** | Does not use the arrays for any useful purpose. | Uses arrays to gather info, but bot strategy does not utilize the array info effectively. | Uses arrays to produce a strategy for the bot. The strategy is somewhat sophisticated. | Uses arrays properly and effectively. The bot’s strategy is highly sophisticated and directly dependent on the info in the arrays. |
| **Success** | The bot cannot beat any bot. | The bot defeats the built-in bots some of the time | The bot defeats the built-in bots most of the time | The bot can match up or defeat a human bot |
| **Style/**  **Doc** | 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, including use of **Arena constants**. |
| **Comments** |  | | | |

# So How do you get started?

1. Get a picture and load it onto your bot
2. Dodge – probably the most important action you can take. After all, staying alive gets you points! Creating these methods might be useful:
   1. double getDistance(int x1, int y1, int x2, int y2) – use your trusty Grade 10 distance formula!
   2. int getDangerLevel(BotInfor me, Bullet b) – is he coming toward me? Will need to check all 4 scenarios
   3. int getMostDangerousBullet(BotInfo me, Bullet[] bullets) – find all the bullets that are head in your direction and are within a certain range
   4. dodge(BotInfo me, Bullet[] bullets) – use the above two methods wisely to create an algorithm for dodging.
3. Once you can successfully dodge, it’s time to make your AI a bit more sophisticated. Some ideas for additional methods:

isAtTopEdge(),… - you don’t want your bot move to an edge and stay there

getClosestBot(BotInfo me, BotInfo[] liveBots)

shoot(closestBot)

move(closestBot)

dodgeTombstone(deadbots) – bots tend to get stuck to tombstones. Don’t let it happen to you!