Robocode Project

Overview

The Robocode Project is a way for students practice programming skills in a competitive team-based context. You will implement a single software robot from the ground up that functions in the Robocode environment. Robocode robots are tanks with radar and a single turret. Your goal will be to match or exceed the performance (based on score as computed by the environment) of the robot team provided to you with the project materials, affectionately known as **TeamSocket** (composed of **FailBot** and **PassBot**). Your robot will have the following specification **which students are required to follow**:

Package:	`edu.ufl.cise.cs1.robots`
Class Name:	`[Determined by Student]`
Extends:	`robocode.TeamRobot`

Getting the Code Base

This project requires students to work in an existing code base. You will need to download the project from the Git repository on GitHub, using the following URL to clone the repository:

https://github.com/uf-cise-cs1/robocode.git

Students must only use those resources defined in this document. Teamwork is for coordination and protocols, but not for the code of individual robots; each student must design and implement a unique robot that will be part of a robot team. If and when a student uses one of the sanctioned outside resources, proper citation of strategies developed by others is ***crucial***. Failure to properly cite the use of ideas developed and/or published by others will be considered academic dishonesty.

When a student uses one of the sanctioned resources in the development of his/her robot, the student will clearly site the source of the strategy in source code and in their report.

UNDER NO CIRCUMSTANCES ARE STUDENTS PERMITTED TO COPY CODE FROM ANOTHER SOURCE. All implementations of code must be done “clean room” style – students may read about techniques but are required to write all code from scratch.

In addition, robots may only use data generated by the student or derived from references. While students may use training data from outside sources to train their robots, they may not use any data or script which defines the fundamental architecture of the robot's decision-making process.

Tournament Rules

The tournament will adhere to the following rules:

- Robots submitted will be provided (compiled) to future classes.
- Students may submit at most one robot as an entrant into the tournament.
- All robots submitted for a grade must be submitted as entrants to the tournament.

Robots could be disqualified if they…

- Do not start due to bugs in their codes, or cause the Robocode environment to crash
- Use neural network weights / scripts / code generated or provided by a third party

However, an acceptable robot might still…

- Use training sets generated by a third party to train a neural network
- Dictate behavior based on a script rather than hard-coded instructions

Team Requirements

Each student must include exactly one robot on a team with other students. The follow rules apply:

- All teams must be 2-4 robots in size
- All teams must choose a team name and team color
- All robots in a team must use the same radar color and bullet color

Students will work together to prepare their team strategy. Students may not share any code to implement such a strategy, though they may develop a communication strategy and protocol code that can be shared.

Building a Tournament Team Package

Before the tournament each team must submit a single jar package containing all team robots, built as follows:

1. Place all robot source (java) files in the "Student Robot/edu/ufl/cise/cs1/robots" folder in the project.
2. Make sure all robots have "`package edu.ufl.cise.cs1.robots;`" as the first line in the source.
3. Clean and rebuild the project using IntelliJ. **DO NOT SKIP THIS STEP.**
4. Check that each robot functions in the Robocode environment.
5. Create a team with the robots (Robot → Create a robot team.)
6. Create a team package (Robot → Package robot for upload.)

NOTE: Be sure the checkbox marked "Include source" is checked!

Robot Scoring

This assignment consists of two parts: **robot performance** (for a grade) and the **tournament**.

Robot Performance

Each team's performance will be measured against the robot team provided as part of the assignment (TeamSocket) and will be graded as follows:

- Student robots will be posed one-on-one against TeamSocket for 100 rounds (1200x1200 battlefield)
- Teams must score as well or better than TeamSocket in order to be graded 100%
- Teams failing to score as well as TeamSocket will be graded less than 100% (proportionally)

Tournament
The tournament battles consist of a team bracket on terrain as follows:

Rounds of Battle:	5
Battle Field Size:	1200x1200

Battle Ranking
Each round, robots will be awarded points according to survival, damage, and elimination of opponents. Robots / teams scoring 1st the team competition will advance to the next round.

Ties
In the event of a tie for a single battle, sudden death rounds will be added until clear winners emerge. In the event of a tie for the win, the tied competitors will participate in three rounds of battle on a 1200x1200 battlefield, with additional sudden death rounds as necessary to determine clear ranking.

Hall of Fame
The winning team members will be placed in a battle with current Hall of Fame robots. All robots will compete together in a Free-For-All battle on a 1200x1200 battlefield for 1000 rounds. The top ten robots will be placed in the Hall of Fame according to rank; the top four teams will be placed in the Team Hall of Fame. Ties will be broken using sudden death rounds (on 1200x1200) with all competitors until a clear order emerges.

Deliverables
Students will submit their robots **individually** and will also provide a **team package** (jar) for the competition as well as a **team report**:

1) Robot source file (as defined above) on Canvas
2) Team package (jar file), submitted directly to your TA (email or Canvas mail)
3) Design and Post-Mortem document including identification of the following:
    a. Diagram(s) and description of robot behavior and methods with description (100-300 words each)
    b. Identifications of successes (“what went right”) and failures (“what went wrong”) (~300 words)
    c. Reflection on project (one per student; 100-300 words)

Grading
Grading of the project will be as follows:

Design & Post-Mortem Document – 50%
This section of the grade will be based on completion and quality of the design and post-mortem document.

Team Performance – 50%
The remaining portion of the grade will be based on assessment of team’s performance, with a score at or exceeding that of TeamSocket over 100 tests attaining 100% credit.

Submission
Students will submit a single zip file with a “src” folder and a PDF (Acrobat) or HTML document on Canvas.

The path within the zipfile to the source (.java) file should be as follows:
```
src/edu/ufl/cise/cs1/robots/YourRobotNameHere.java
```