TRADECORE

Node.js test

Battle simulator

This is a test used for the assessment of the node.js back-end developer position.

Introduction

The focus of the test is to assess the maintainability and extensibility of the code you present.

Description

This simulator is supposed to determine a battle outcome.

The Battlefield will support a configurable number of armies. Each army can have a configurable number of squads (>2). Each squad consisted of a number of units (>5).

Once the simulator is started all army squads will start attacking each other until there is only one army left.

Units

Each unit represents either a soldier or a vehicle manned by a predetermined number of soldiers.

All units have the following properties:

Property	Range	Description
health	% [0-100]	Represents the health of the unit
recharge	[100-2000]	Represents the number of ms required to recharge the unit for an attack

Soldiers

Soldiers are units that have an additional property:

Property	Range	Description
experience	[0-50]	Represents the soldier experience

The **experience** property is incremented after each successful attack; and is set to calculate the attack success probability and the amount of damage inflicted.

Soldiers are considered active as long as they have any health.

Attack

Soldiers attack success probability is calculated as follows:

0.5 * (1 + health/100) * random (30 + experience, 100) / 100

'random' (min, max) returns a random number between min and max (inclusive)

Damage

The amount of damage a soldier can afflict is calculated as follows:

0.05 + experience / 100

Vehicles

A battle vehicle has these additional properties:

Property	Range	Description
operators	[1-3]	The number of soldiers required to operate the vehicle

The **recharge** property for a vehicle must be greater than 1000 (ms).

The total health of a vehicle unit is represented as the average health of all its operators and the health of the vehicle.

A vehicle is considered active as long as it has any amount of health and there is a vehicle operator with health.

If the vehicle is destroyed, any remaining vehicle operator is considered as inactive (killed).

Attack

The vehicle attack success probability is determined as follows:

0.5 * (1 + vehicle.health / 100) * gavg(operators.attack_success)

‘**gavg’** is the geometric average of the attack success of all vehicle operators

Damage

The damage afflicted by a vehicle is calculated:

0.1 + sum(operators.experience / 100)

The total damage inflicted on the vehicle is distributed to the operators as follows: 30% of the total damage is inflicted on the vehicle, 50% of the total damage is inflicted on a single random vehicle operator.

The rest of the damage is inflicted evenly to the other operators. If there are no additional vehicle operators, the rest of the damage is applied to the vehicle.

Squads

Squads consist of a number of units (soldiers or vehicles) that behave as a coherent group.

A squad is active as long as is contains an active unit.

NOTE: Only property a squad has, is an attack strategy.

Attack

The attack success probability of a squad is determined as the geometric average of the attack success probability of each member.

The squad will attack when all units in the squad have recharged.. Even if the squad is attacked his recharge time will not be interrupted and he will be able to attack when it is ready.

NOTE: Squads do not attack in specific order, nor they wait for anything else, except recharge time of their units.

Damage

The damage received on a successful attack is distributed evenly to all squad members. The damage inflicted on a successful attack is the accumulation of the damage inflicted by each squad member.

Attacking & Defending

Each time a squad attacks it must choose a target squad, depending on the chosen strategy:

Strategy	Description
random	attack any random squad
weakest	attack the weakest non-ally squad - Total squad health - Experience per unit - Number of units - Total squad damage
strongest	attack the strongest non-ally squad - Total squad health - Experience per unit - Number of units - Total squad damage

Once the target is determined both the attacking and defending squads calculate their attack probability success and the squad with the highest probability wins. If the attacking squad wins, damage is dealt to the defending side. If the attacking squad loses, no damage is dealt to either side.

Configuration

The following constraints should be configurable:

- The number of armies: 2 <= n
- The choice of attack strategy per army: random|weakest|strongest
- The number of squads per army: 2 <= n

- The number of units per squad: 5 <= n <= 10 (vehicle is always calculated as a single unit no matter how many operators it has)

UI

There are no UI requirements for this task

Requirements

- Use **Node.JS** for development (do not use transpilers).
- You are required to provide a url to the git repository with the task including all commits/changes. If you complete the task and send it without a git repository, your application will be refused automatically.
- **Task deadline is 5 days.**
- Please use ES6+
- A battle log must exist.
- Please provide readme file with installation/run instructions
- Some **unit tests** for the project should be created.