# Invaders

You need to create the board computer for a spaceship that is attacked by invaders. The computer should support the following functionality:

* void AddInvader(IInvader) - Adds an invader to the computer records
* void Skip(int turns) - Skips the given turns in time. Every invader moves a single step closer to the ship with each turn. All invaders that reach the ship are destroyed, even if the ship is already destroyed.
* void DestroyTargetsInRadius(int radius) - Destroys all targets in the given radius (all invaders such that **distance ≤ radius**)
* void DestroyHighestPriorityTargets(int count) - Destroys the given count of targets prioritizing them **first by distance**, **then by damage**
* IEnumerable<Invader> Invaders() - returns a collection with all remaining invaders **ordered by their appearance on the radar**.

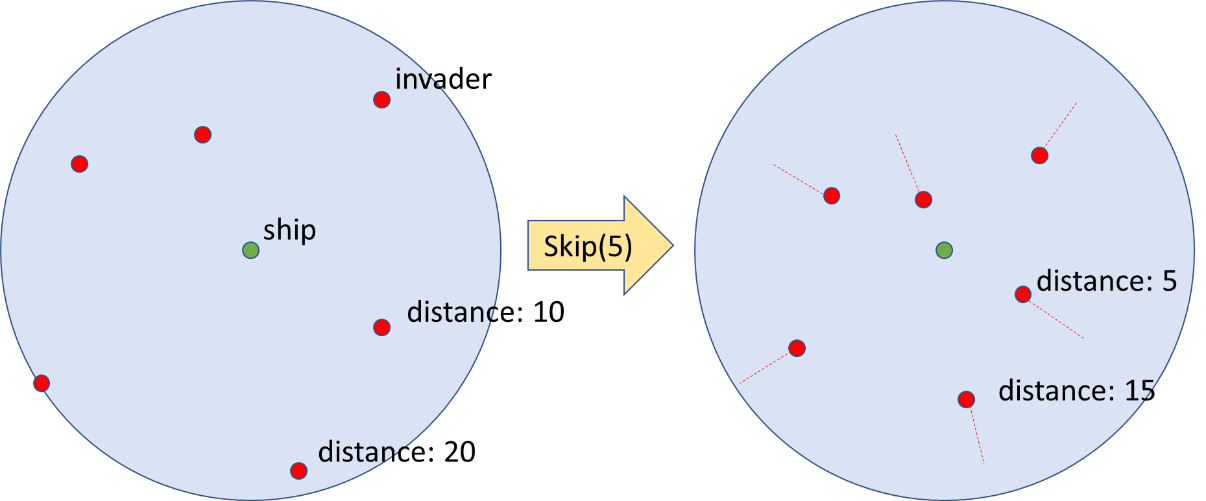
And the following properties (getters for Java):

* int Energy - returns the current health of the spaceship. If an invader distance to the ship becomes zero or less the ship takes the damage of the invader. If the energy of the ship falls below zero, the property should return zero instead. Initial energy is given as parameter when initializing new Computer. If negative energy is given, throw Argumen/IllegalArgument Exception.

**Invaders** are objects flying towards the ship that have the following properties/getters:

* int Damage
* int Distance

Invaders **advance 1 point of distance** toward the ship **with each turn passed**.



### Input and Output

You are given a **Visual Studio C# project skeleton** (unfinished project) / **IntelliJ Java project skeleton** (unfinished project) holding the interface IComputer and IInvader. You are also given the unfinished classes Computer and Invader and **tests** covering the computers **functionality** and **performance**.

Your task is to **finish the given Computer class** and make the tests run correctly.

* You are **not allowed to change the tests**.
* You are **not allowed to change the interface**.

### Constraints

* Invader **damage** and **distance** will be positive integers in the range [0…232-1]
* **Invader count** will be in the range [0...30000].
* Paremeters for methods will be integers in the range [0…10000]

### Submissions

Submit an archive (.zip) of the source code + external libraries.

# Scoring

Each implemented method brings you a specific amount of points, some of the points are awarded for correct behavior, others for performance. The total points are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Correct Behaviour | Performance | Total |
| Functionality | 80 | 120 | 200 |