Fossil's Technical Test #3

For Senior level and above

Overview

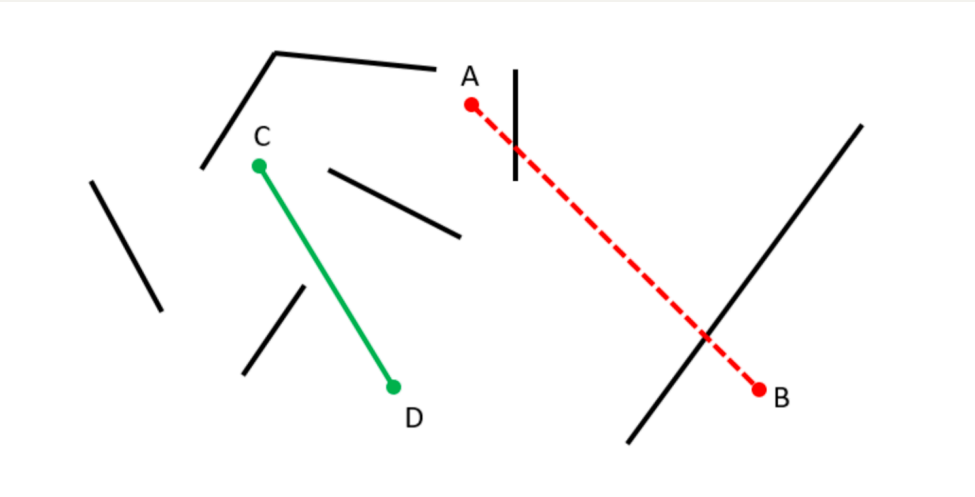
At senior level, besides strong programming skills, we expect our team member to excel at problem solving and technical writing skills. He/she needs to be able to transfer the solutions throughly to other colleagues, including junior members.

In this test, you will be put in a tough situation where you are required to write a solution/technical document for your team, so they can work on the concrete implementation of the problem *(with many weird technical constraints, of course)*.

You do not have any other way to communicate with your team except this document. So, your document should be as detailed as possible, as easy to understand as possible that even junior members can confidently do it.

Don't forget that "A picture is worth a thousand words"!

The problem - A Collision Detection Game



- A video game scene consists an huge number of **Walls**.
- The image above shows the aerial view of the scene, where a wall is displayed with a black line and each player is represented by the A-B-C-D point. *(A & B cannot see each other, while C & D can)*
- The game scene are enormous, it could be up to several hundred kilometers. *(The Earth's curve and the human eye's limitation are out of*

the scope of this test)

Functional requirements

- Creating/Loading/Saving a game scene from/to a single file.
- Adding/Updating/Removing a wall (or line segment) from the scene file.
- Performing a line-of-sight test, where with any two given positions, the application needs to determine whether they can see each other or not.

The challenge

- **Key requirement** - The line-of-sight test must has the complexity class better than O(n), where n is the number of the wall on the scene.
- **Key requirement** - Every scene is store in a single file only, including all the walls, indexes, additional data, etc. Adding a second file for a scene will result in a fatal error.
- The game could create multiple scenes, each scene is associate with a single file. However, this is not a requirement and you can skip it.
- You do not need to worry about the disk storage ability.
- Do not use any existing storage/database libraries, even the built-in ones. You can only use the standard library for `I/O handling`. In other words, any `file read/write` function will be implemented by our team, using the language's basic `I/O functions`.
- Text file format & system's `Serialization` are not allowed.

Test duration

- We expect the document be sent back in 1 week since you received the test.
- You are allowed to extend the test duration as you need. It will not affect your performance. We value the quality of the delivered solution. So do your best!