Medrick, Game development department.

Dear Soheil,

Please build a clone of the game `LigthBot hour`, and send your project till Friday night, 8th of July.

You may download the original game via the link provided below:

<https://play.google.com/store/apps/details?id=com.lightbot.lightbothoc&hl=en&gl=US>

As you would notice the goal of this game is to teach its users how to code (or at least to think as a programmer), it has 3 different seasons, each with ~8 levels, we do not ask you to develop the whole game, just 1st season and first level of the 2nd season is enough.

Also take note that we don’t need you to develop the main menu, you can jump straight to the level, maybe a simple menu to choose the target level would be enough.

Keep note that the level visuals also don’t matter, you can do it with simple primitive cubes, the character also can be anything, even a capsule.

Tutorials are also not important and not needed.

Music/audio system is not important.

What we expect to see as for the frontier:

* A menu letting us select a level 1 to 9
* User can select All mechanics available in that level, exactly the same as the original game
* Each mechanic (move, jump, rotate and etc.) should work as the original game.
* After completing a level, a button appears letting us jumping to the next level, again as original game.
* As said the first 9 levels are required, so as you see in the original game, not only basic block (mechanics) is needed but also procedure blocks (introduced in season 2 of the original game) are also needed.
* selecting a block from down blew should add that to “main” block, touching a block in the `main` box should remove it. Note that the drag and drop feature is ***not*** needed.

What we expect to see as the back bone of your scripts:

* Clean architecture. Including:
  + Possibility to add a new mechanic (block) without need of changing previous developments, yeah, it’s called open/close principle.
  + Possibility to add new levels easily.
  + Clean separation between logic/presentation (do it whenever/however you think its needed)
  + Clean, maintainable and logical dependency handling
* Clean code, what can better guarantee long living of a project else than a clean code base