Challenge 13: Auto-Thresholding

Constance has built a robot that uses a light sensor as a “shadow detector” – when passersby cast a shadow on the robot, it springs into action. In the robot lab, she notices that whenever anybody walks in front of her robot, the light sensor reading goes above 200. (Remember, higher numbers correspond to less light reaching the light sensor.) She gets her robot working perfectly with the following *PicoBlocks* code:

```
to react-to-shadow
    waituntil [sensor1 > 200]
    spring-into-action
end
```

On the day of the big robot exhibition, Constance moves her robot to a sunlit exhibition space. In the mid-afternoon, there is so much light that her robot never turns on when people pass by it. After the sun has set, her robot turns on whether or not people are passing by.

Of course, the cause of this heart-breaking behavior is the fact that Constance has “hard-wired” (*i.e.*, fixed as a constant) the threshold value used by her program. A threshold value that works well in one ambient lighting condition may fail miserably in other environments.

A much better strategy is *auto-thresholding*, in which the robot detects and records the ambient lighting conditions each time its main program starts, and uses this information to automatically determine the correct shadow threshold. Your challenge is to implement this strategy by hooking a light sensor to the Handy Board and writing code that can reliably detect shadows under a variety of ambient lighting conditions (e.g., robot lab with lights on and off, Sage lounge, outdoors on a sunny day). Your shadow detector can trigger any simple action of your choice – e.g., turn on a motor, beep, *etc*.