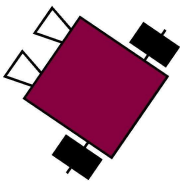


Brooklyn College The City University of New York

Exploring Robotics (CORC 3303)

UNIT D Lab, part 1 : Light Sensors and Forks

vocabulary

- light sensor
- fork
- merge
- jump
- loop

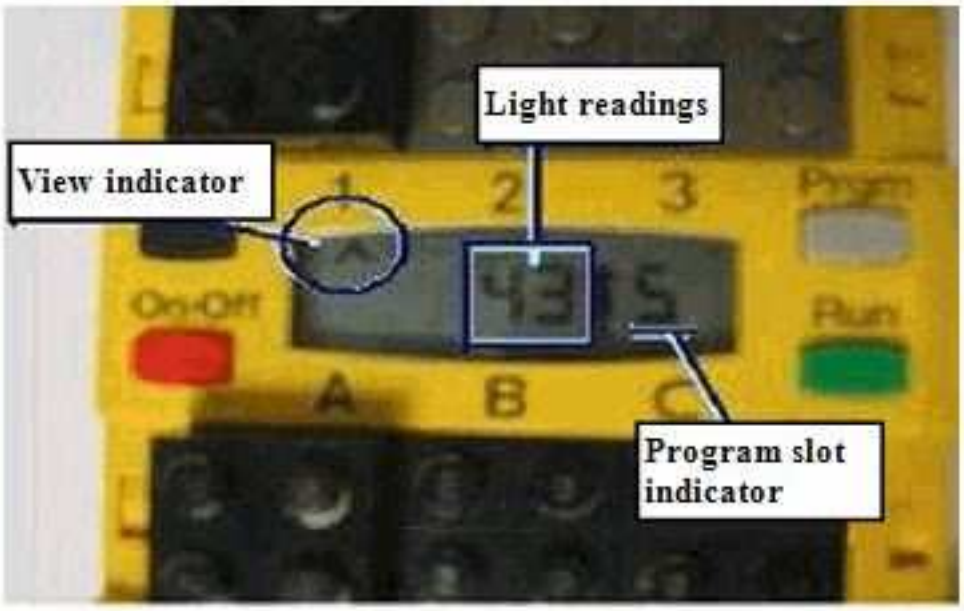
materials

Make sure you have all of the following materials before you start the lab:

- lego go-bot
- lego light sensor

light sensors

- The light sensor has a transmitter and a receiver. It transmits infrared light (also called "IR"), which bounces off objects and then returns in the direction of the receiver. The receiver records a value indicating how much light it read, which basically tells you about the brightness of the object that the light sensor is pointing at. The light sensor produces a value between 0 and 100, where 100 means very bright and 0 means very dark.
- Did you know that your TV remote control uses IR to talk to your television?
- **Here is how you can see what the light sensor's value is:**
  This is done by turning on the RCX and pressing the **VIEW** button until the arrow at the top of the screen is under the port the light sensor is plugged into. For example, my light sensor is connected to port 2. I would turn on the RCX and press the VIEW button twice. Next to the image of the robot man, there is a number between 0 and 100—this is the value being read by the light sensor.



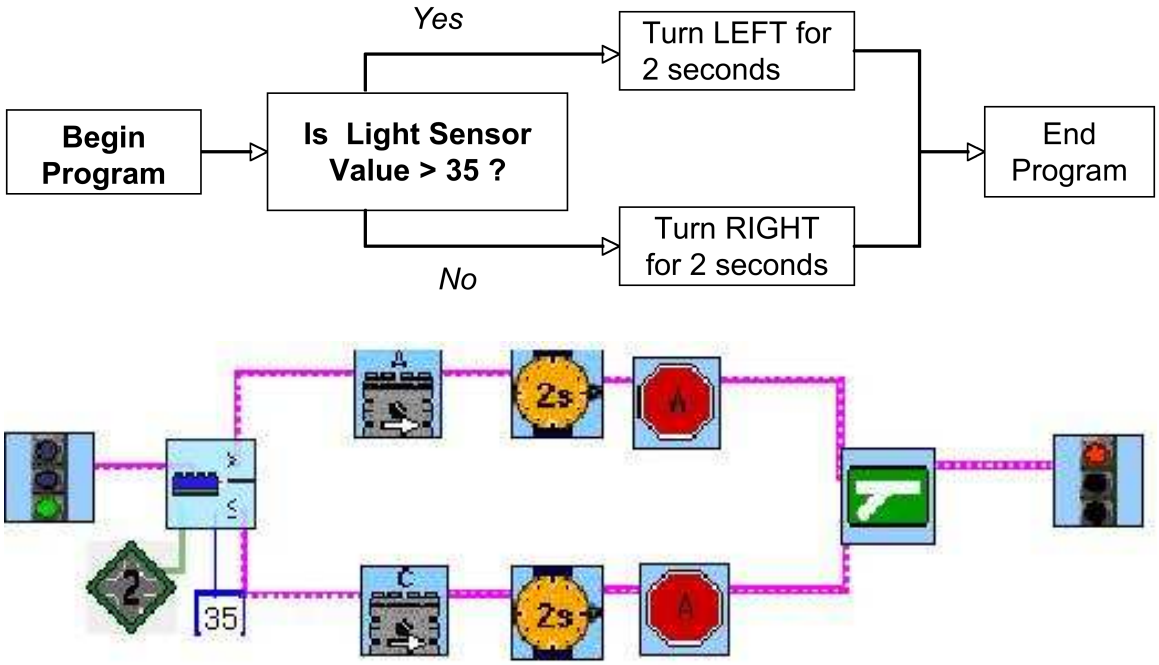
- Put your robot on at least four different locations: (a) white foamcore or paper, (b) black tape, (c) green tape, and (d) silver tape. View the different values for the light sensor and record the values in the table below:

	black	white	golden	silver
light sensor value				

decision-making

- Up to now, your programs have been sequential and linear. Basically, your programs have been executing each commands starting from the **green light** and going until the **red light**, without skipping any commands. For instance, we might like to program the robot to do one action (i.e., go forward) when some condition is true (i.e., light sensor sees black) or else do some other action (i.e., go backward) when the condition is not true (i.e., light sensor sees white). However, we need a **decision-making** mechanism so that our robots can react to their environment **autonomously** (i.e., without a human touching it). This decision making ability is based on its sensory inputs (e.g., the value of the light sensor) and will make the robot a little more intelligent! Decision making is achieved by **conditional execution** in the programming environment. In RoboLab, by using **fork** structures, we can allow programs to behave differently based on different values of sensor inputs.
- **How does a fork work?**
  - When a fork is reached in a program, one of two **branches** will be taken based on the value read from one of the robot's sensors.
  - All of the fork commands on the forks palette use the greater-than ($>$) or less-than-or-equal ($\leq$) condition to determine which branch to execute. These conditions are based on a **threshold value**  . If the value read by the fork's sensor is greater than the threshold value, then the program follows the top branch. Otherwise, if the value read by the fork's sensor is less than or equal to the threshold value, then the program follows the bottom branch. For example, RoboLab can let a program make a decision depending on the value it gets from the **light sensor** when you use the **light sensor fork**  .
  - You must attach the correct port number modifier (i.e., port 1  , port 2  or port 3 ) to the fork where your actual sensor is conncected.
  - A **fork merge**  is required at some point in the program with any fork command. It completes your conditional structure and brings the two branches of the program back together.

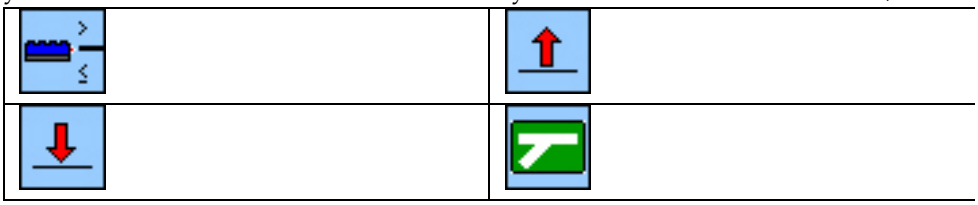
- Consider the following program:



  - Create this program in RoboLab and use the light sensor value returned when the robot sees black tape as the threshold value. **REMEMBER:**
    - The light sensor fork requires you to specify a **threshold** value. For instance, the above example uses a light sensor fork with a threshold value of 35—but you need to **calibrate** your robot to determine which value it reads when it sees black tape, and then use that value instead of 35.
    - Don't forget to attach the correct port number to the fork. For example, our light sensor is attached to **port 2**.

putting it all together

1. **Add a light sensor to the go bot.**
   - Attach a light sensor to the front of your go-bot. Make sure it is looking down and attached as closely as possible to the ground.
   - Connect the light sensor to port 2 of the RCX with a connecting lead.
2. **Find the icons.**
   - Locate the functions palette. See if you can find each of the following icons. What do you think each icon does? Write down your ideas in the boxes below, next to each icon.

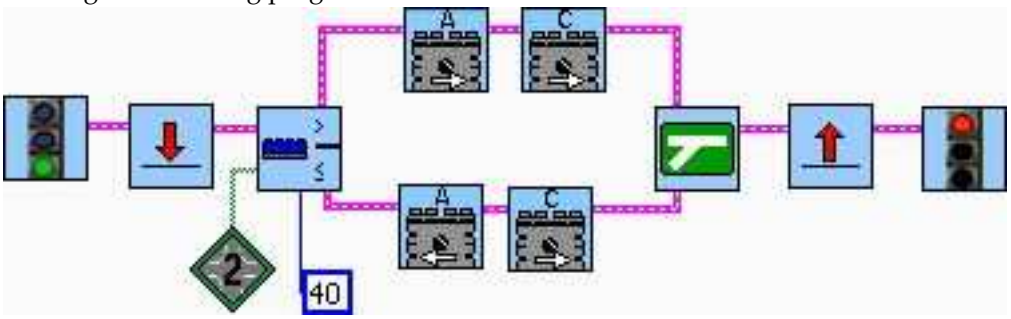


3. **Jumps**
   - Unlike the loop  , which repeats a certain number of times based on a counter or a condition, a jump repeats forever. This is also called an *infinite loop*.
   - In RoboLab, a jump is created by using a land  icon in place of a start-of-loop icon and a jump  icon in place of an end-loop icon.
   - The jump icons comes in several colors: red, blue, yellow, green and black. Make sure to use the pair that match in color.
   - Jump icons will run forever. If you use them wrong, the only way to stop the program is to turn off the RCX.
   - Remember to place the land  icon BEFORE the jump  in your programs!
4. **Light Sensor Fork**
   - A LEGO light sensor, when attached to the RCX, emits *InfraRed (IR)* signal and reads values indicating the amount of light that bounces back from the surface that it is directed at (this is called "active mode").
   - The light sensor returns a value between 0 (dark) and 100 (bright).
   - In order to perform different actions based on light sensor values, the light sensor fork  icon can be helpful. This icon compares the values read by the light sensor to a user-specified (or default) value and chooses a branch of the program to execute.
   - The light sensor fork has 2 modifiers:
     (a) Port: corresponds to the RCX port number to which the sensor is connected.
     (b) Compare to: a numerical constant modifier that holds the value to compare with the light sensor reading. If this modifier is not specified, the default value, 55, is used.

- Each fork branch needs to be connected with a fork merge  icon before the program terminates.

5. **Create your program**

   - Use the mouse to select, drag and drop RoboLab icons onto your canvas and wire them together, creating the following program:



   - What do you think it does? Write down your ideas below: