Goal: *Familiarity with the NXT brick and with basic movement in RobotC.*
Materials Needed: *Assembled robot with one downward-facing light sensor*

**Part 1: The NXT Brick**
You should have discussed the NXT with your group last week, but review the basics again at the start of this week. The Lego NXT has 4 sensor ports, 3 motor ports, 4 buttons, and a screen. Ask your students which of these are input and which are output. If they don’t know, provide examples of I/O with a computer: mouse and keyboard are input while monitor and speakers are output. Make sure to show them the battery, the charger, the battery life indicator, and how to power the robot on and off.

**Part 2: Working in RobotC**

A. *Basic Program Structure*

Creating a program in RobotC starts with navigating to Robot -> Motors and Sensors Setup. Choose "Motor equipped" in the dropdown menu for each active motor. You can give your motors names, such as "mLeft" and "mRight." Navigate to the “Sensors” tab to set up an sensors you will use in your program.
In RobotC, every program needs a “Main” function, shown below. For now, all the code you write will go inside main’s curly braces:

```c
task main()
{
}
```

B. Moving Forward

Since our robots are built with one motor on each side, moving straight happens when both motors travel at the same speed. When controlling the motors in RobotC, you can either use the default names, motorA and motorB, or the names you gave your motors in setup shown above, like mLeft and mRight. The following code is equivalent:

```c
task main()
{
    motor[motorA] = 50;
    motor[motorC] = 50;
}
```

C. Counting and Time

To wait for a specific amount of time, use the wait1Msec() function.

D. Turning

The robots we are working with aren’t built to tilt the wheels to control steering. How, then, can we get the robot to turn? Get your students to figure out the solution to this question: power only one wheel, or spin the wheels in opposite directions. Explain the difference between these two approaches.

```c
task main()
{
    motor[motorA] = 50;
    motor[motorC] = 0;
    wait1Msec(5000);
}
```

```
```

```
```
E. Sensory input

Just like with motors, we have to initialize all sensors in the Motor and Sensor Setup menu. For now, setup a light sensor as shown below.

Once the light sensor is initialized, you can use the “SensorValue” function to take a sensor reading. Once you take a reading, you can compare it to the threshold white/black value, as shown below.

```plaintext
while(SensorValue[lightSensor] > 45)   // While the reading is greater than 45 (a light surface)
{
    motor[motorB] = 100;    // Motor C is run at a 100 power level.
    motor[motorC] = 100;    // Motor B is run at a 100 power level.
}
```

**Programming Challenge**

Write a program that moves forward until the robot senses black.