Lesson 6: Concurrency with the Vacuum Gripper

Vocabulary

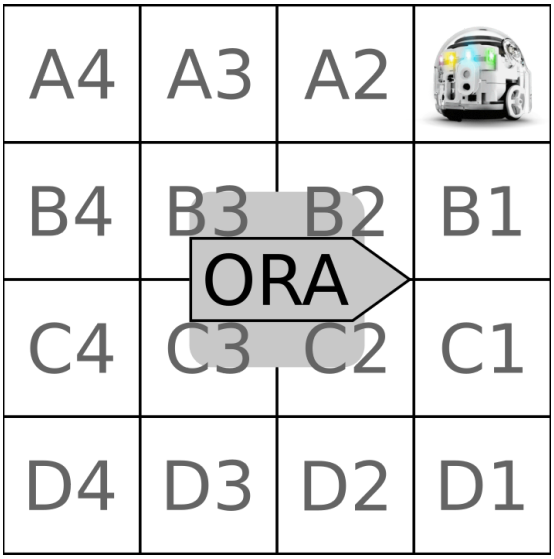
Concurrency - Two or more robots performing tasks simultaneously.
“Run Concurrently” Block - Code block that provides 2 frames for actions that will be run in parallel.
Porosity - Characteristic of a material, tiny spaces or holes through which liquids or air can pass.
Surface Texture - Characteristic of a material, the roughness/irregularity or smoothness/uniformity of the object surface.

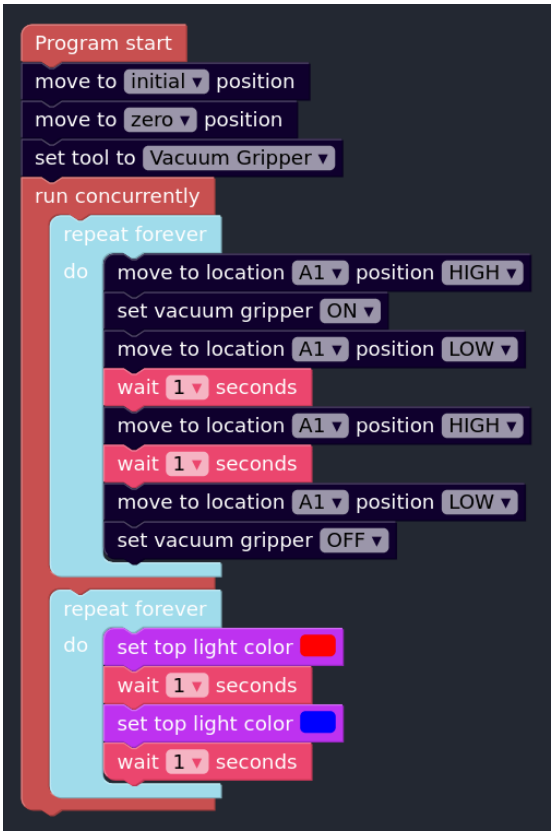
Direct Instruction

Engage: What is Concurrency?

On the connected display, open and show the **EngageL6Code** (Share Code: ayyt94t). Direct the students to the Engage section of the Student Activity Sheet. The code has the ORA arm pick up Evo and move across the mat while Evo continues to change colors. Both bots are running code operations concurrently to each other.

Setup: Ora is mounted in the center of the mat. Evo is centered on square A1.





Play Lesson 6 Video

Explore Vacuum Gripper Capabilities

Ask students what kind of objects are unlikely to work with the vacuum gripper. Why do those objects seem like they would not be compatible with a vacuum grip?

These common objects are to be evaluated for compatibility.

- Ping Pong Ball
- Textbook
- Paper Clip
- Sock
- Sponge
- LEGO structure (studs up)
- LEGO structure (smooth side up)
- Plastic Bottle Cap
- Gift Card (or other similar cards like ID or store loyalty card)
- DVD/Blu-Ray Case
- Pencil
- Golf Ball

Direct students to the Student Activity Sheet to mark objects they think cannot be picked up by the vacuum gripper. Students will identify characteristics of objects that are likely compatible with the vacuum gripper and objects that are likely NOT compatible with the vacuum gripper. Students will pick 3 of the choices they would like to see tested with the vacuum gripper. Find the 3 most popular choices for the class and test them. Some objects may need to be substituted for similar objects if you are unable to provide one for testing.

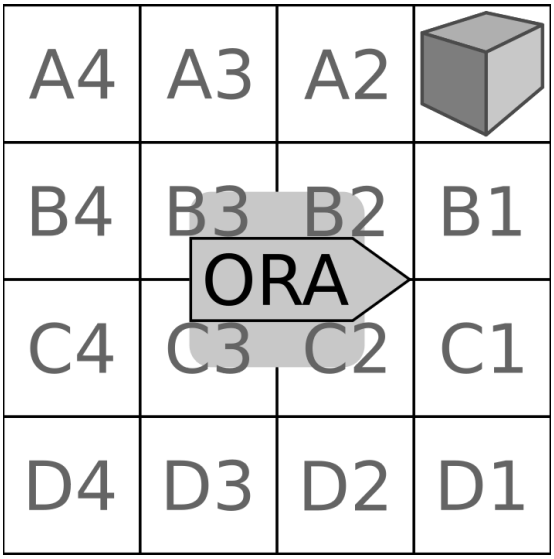
Good for vacuum gripper:
Ping pong ball, LEGO structure (smooth side up), plastic bottle cap, gift card

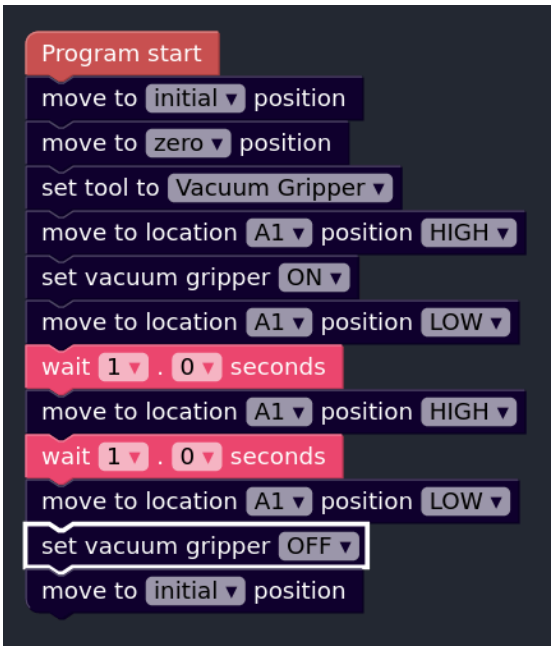
Bad for vacuum gripper:

Textbook	Will lift cover only if anything. Too Heavy
Paper Clip	Thin surface that air travels around
Sock	Porous and air travels through it
Sponge	Porous and air travels through it
LEGO structure (studs up)	Irregular surface that air travels around
DVD/Blu-Ray Case	Too heavy, must be grabbed perfectly in center
Pencil	Thin surface that air travels around
Golf Ball	Surface is dimpled, doesn’t allow for consistent airtight seal

Select 3 objects as a class to test the objects students are most curious about testing with the vacuum gripper. Have students compare their predictions to the results of these objects when tested with the vacuum gripper in Quadrant A1 using the **ExploreL6Code** (Share Code: pmkf7bw).

Setup: Ora is mounted in the center of the mat. Test object is centered on A1. For very low profile objects, place an object underneath to elevate the test object to reach the vacuum gripper. A deck of cards box is enough elevation for most objects. Objects that roll may need held in place by bracing them with another object.



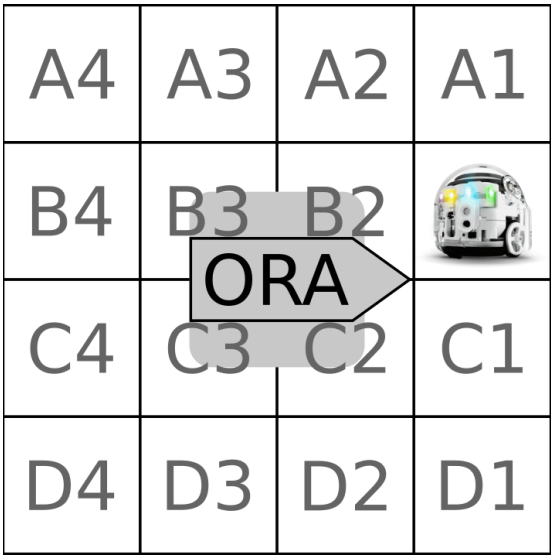


Ask students to share their expectations and how they compare to what they observed when testing these objects.

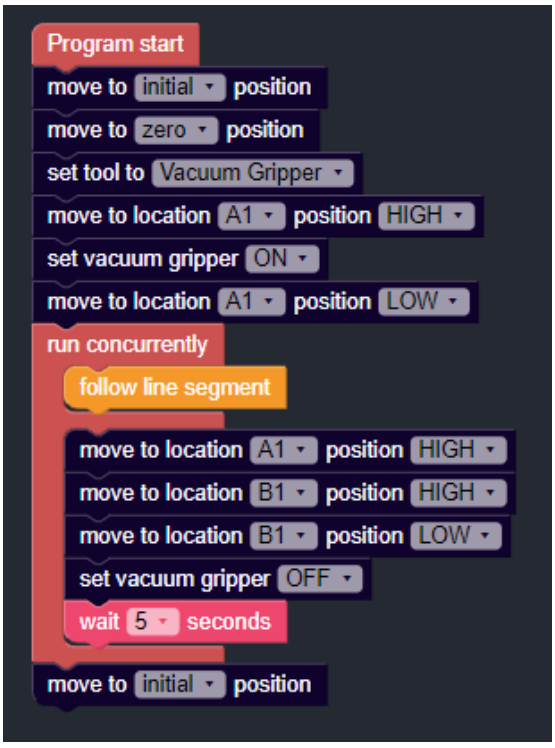
Explain: Concurrency and Timing

Align the printable Evo paths so that Evo can travel from B1 to A1 when the code executes. Show students both sample codes (**ExplainL6ConcurrencyTest1** Share Code: w5ne7gk and **ExplainL6ConcurrencyTest2** Share Code: 8wrut95) for concurrency, but do not run either one. Students analyze the code to determine which code will successfully pick up Evo at the appropriate location and which code will not. Run the code the majority of the class believes will properly pick up and return Evo to the start of the line.

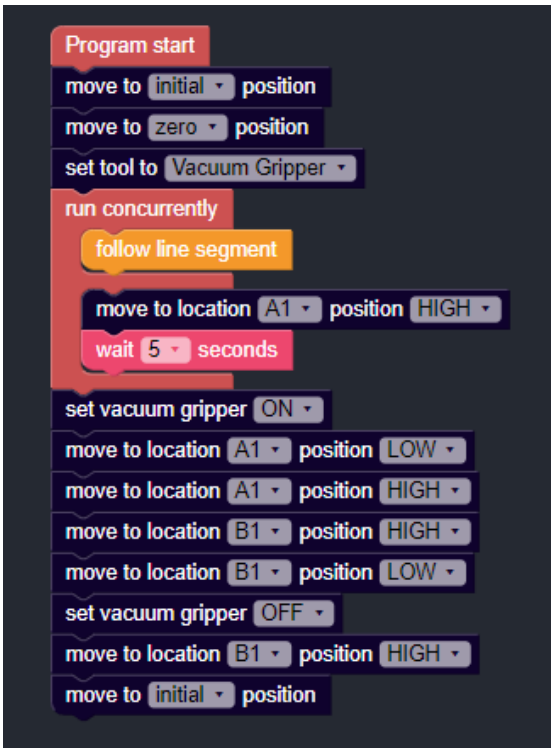
Setup: ORA is mounted in the center of the mat. Evo is centered on square B1. A single Evo line cutout spans from the center of B1 to the center of A1



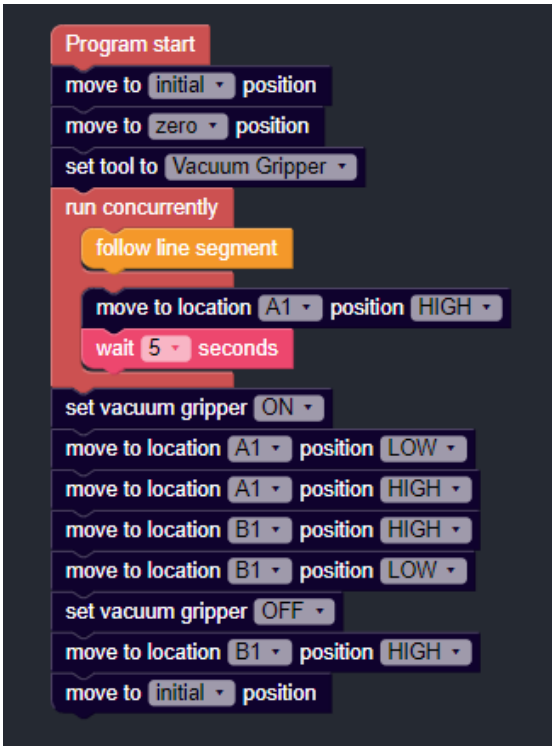
A



B



Run the correct code, **ExplainL6ConcurrencyTest2** (shown right), for students to observe.

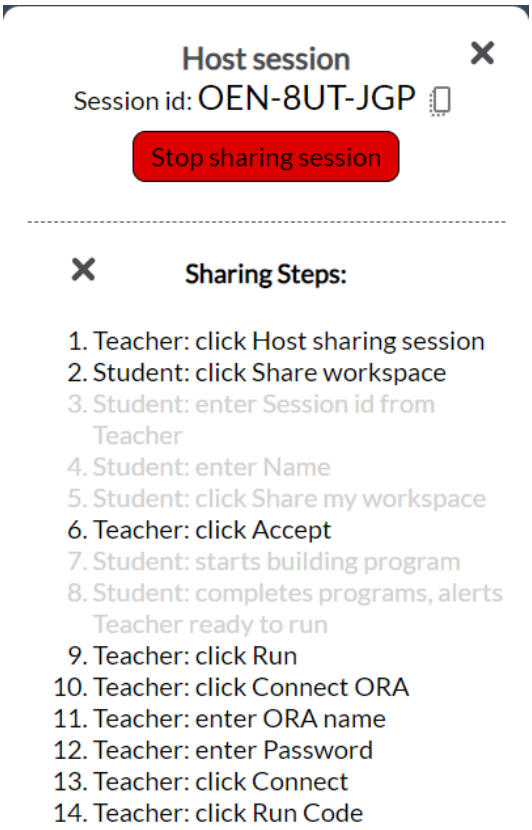


ORA will travel to the initial position, then the zero position, and then it will move to A1 High.
ORA will wait at A1 High for 5 seconds to ensure that Evo has enough time to arrive at the A1 location.
If your path cutout is misplaced, Evo may not be successfully lifted by ORA. Realign the sheet and try again.
ORA will pick up Evo at A1 and deliver Evo to B1. ORA will then return to its initial position and stop.
Code

How quickly does Evo arrive at A1? Is this code waiting too long? Measure the travel time of Evo using a stopwatch or timer smartphone app. Demonstrate how to modify the code (by changing the value Wait block in the ORA concurrency code) to use the least amount of wait time before the ORA picks up Evo, as students will want to know the rate Evo crosses 1 square when writing their code for the next activity.

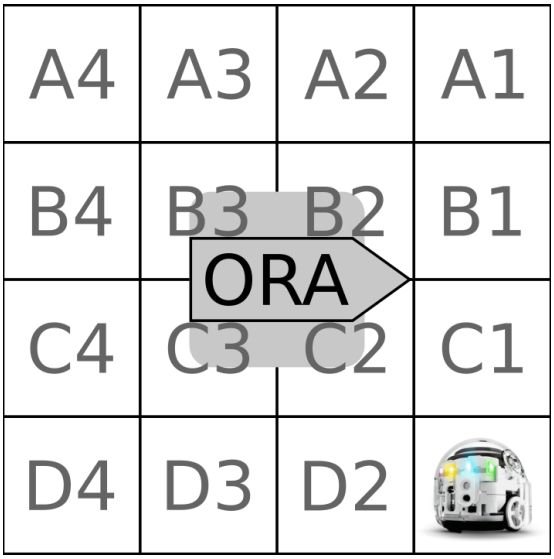
Elaborate: Looping Evo and ORA currently

In this challenge, students will use ORA and Evo concurrently to create a loop of motions where Evo travels from D1 to A1, ORA picks up Evo and delivers Evo to D1, then Evo travels back to A1 to be picked up again. The initial code is incomplete and students must complete the code. The wait is insufficient for Evo to arrive, and the ORA movements to pick up and deliver Evo are missing.



Create a shared workspace to view and run student code. From the Ozobot editor, select "Share" on the right-hand side, and provide students the Session ID to join. Students join via the session id from their own devices. For additional information on initiating and running a shared workspace, refer to Lesson 3.

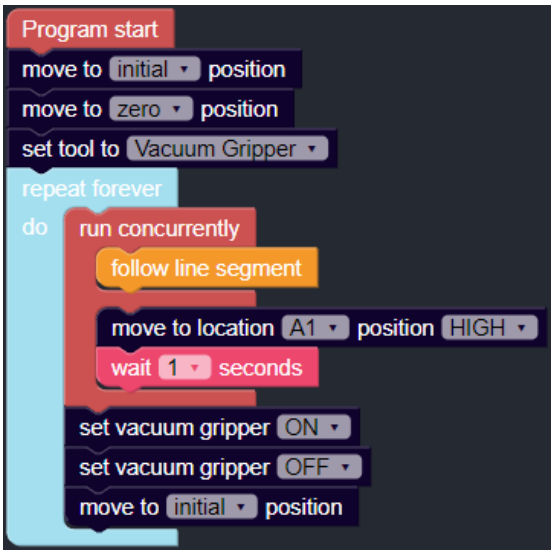
Setup: ORA is mounted in the center of the mat. Evo is centered on square D1. There is a line from D1 to A1.



Criteria:

- ORA loops picking up and relocating Evo
- Evo line follows from D1 to A1 before ORA attempts a pickup at A1
- ORA carries Evo from A1 to D1
- Evo loops following the line segment
- The wait block is the shortest amount of seconds possible

Starter Code with Missing Blocks (ElaborateL6StarterCode Share Code: 968b9h4)



Solutions will vary for fixing the sample code (ElaborateL6Solution Share Code: szuruxg)

Run each group's code and observe if they met the goals from the student activity sheet.

- ORA loops picking up and relocating Evo
- Evo line follows from D1 to A1 before ORA attempts a pickup at A1
- ORA carries Evo from A1 to D1
- Evo loops following the line segment
- The wait block is the shortest amount of seconds possible

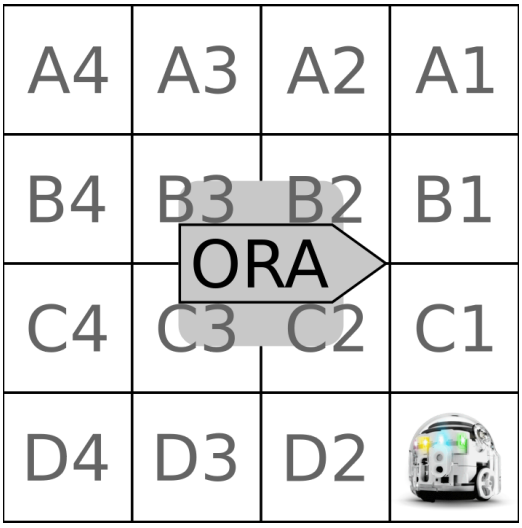
Evaluate: Exit Ticket

Students will fill out and submit the Evaluate section of the Activity Sheets as an exit ticket. (The last page of the Student Activity Sheet)

Optional Extension: Color and Deliver

In this activity, students send ORA to pick up Evo after a similar line following route from D1 to A1, with the addition that ORA picks up Evo, moves through the Painting Station of square A4 High, waits for 3s, and then delivers Evo to D1. Evo changes color after arriving at the painting station and returns to its previous color when the loop starts again.

Setup: ORA is mounted in the center of the mat. Evo is centered on square D1.



Criteria:

- Start with Evo showing red lights and ORA at the initial position.
- ORA and Evo loop a pickup, paint, and deliver sequence outlined below
  - Evo line follows from D1 to A1 before ORA attempts a pickup at A1
  - ORA carries Evo from A1 to to A4 High, then waits for 3 seconds at the "painting station" of A4.
  - Evo changes from red to green after reaching the A4 painting station.
  - ORA delivers the green light Evo to D1, and Evo goes back to a red light
- Minimize time spent to complete the loop through any improvements possible