Learning Coding and Math with RoboBlockly

*Teaching Resource for Coding Level 2 Activities*

Harry H. Cheng

UC Davis Center for Integrated Computing and STEM Education (C-STEM)

[http://c-stem.ucdavis.edu](http://c-stem.ucdavis.edu)
Learning Coding and Math with RoboBlockly

Teaching Resource for Coding Level 2 Activities

Harry H. Cheng
UC Davis Center for Integrated Computing and STEM Education (C-STEM)
University of California-Davis

Copyright © 2015, UC Davis C-STEM Center, All rights reserved. No part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written consent of the author, including, but not limited to, in any network or other electronic storage or transmission, or broadcast for distance learning.

Acknowledgements:
This work was supported in part by the National Science Foundation under grant numbers CNS-1132709, IIS-1208690, IIS-1256780, and by the CaMSP Program of the California Department of Education.

Additional Contributor(s):
Kayce Mastrup
How to Use RoboBlockly Activities in your Classroom

What is the purpose of RoboBlockly?
In RoboBlockly, students program a robot using a user-friendly drag-and-drop method. Students will complete basic problem solving to move a robot or multiple robots.

RoboBlockly is built to allow students to work at their own pace, but in general each problem should take approximately 5 minutes to complete each activity. We encourage you to give students additional time if needed or make it clear that they don’t need to finish the entire set of activities during one class session.

General Computer Usage Requirements
Technology Requirements: Any modern browser on computers, laptops, tables, or smartphones with any type of operating system.

It is important to know that every browser functions differently. We encourage you to test RoboBlockly on the computers you will have students using before implementation. Please test out the following: audio and video streaming quality, default browser specific mechanisms for saving blocks and saving Ch code, etc. all so you are better able to support your students. Make sure that pop ups have been enabled on all computers.

You may wish to provide headphones or ask students to bring headphones to allow students to independently watch tutorial videos.

Prepare yourself
Go through the activities yourself so that you are familiar with what your students will be experiencing. The Teacher Resource Packet contains all the activities and solutions for the pathway. Please note that the activities build on previous activities in each pathways such that students may need to complete some or all activities prior to the activity selected.

1) Determine the purpose for students using RoboBlockly:
   • To support student learning in Math,
   • To support student learning in Computer Programming,
   • To support student learning in Robotics.

2) Based on your purpose, determine what additional resources your students will need for instruction. We do not recommend using RoboBlockly to introduce a mathematical concept but to rather use it for skill building or as a culminating performance task.
   Use as skill building: We recommend that you provide your students with a worksheet that includes important related definitions, work space, leading questions, etc. and encourage your students to refer to their class notes which cover these topics.
   Use as a culminating performance task: Carefully select which activity directly relates to the content you have taught, making note that the previous activities may be necessary to complete to build prior knowledge.

Prepare your students
Help students get excited about RoboBlockly by inspiring students and discussing how computer science impacts every part of our lives. As a class, list things that use code in everyday life, or discuss different ways technology impacts our lives etc.

When using RoboBlockly in class, first demonstrate to students how to navigate and use the RoboBlockly website. There are five Video Tutorials, along with a self-guided interactive non-video tutorial which should be used to help familiarize your students with the different functionalities of RoboBlockly. Helping students understand the functionality of RoboBlockly and which elements can be manipulated in which manners is very important to ensuring your students have full access to the content.
Pre-Requisite Skills

RoboBlockly Activities

Prior to completing the Coding Level 2 activities, students should complete Robotics Level 1 and Level 2 activities. The Coding Activities are intended to be self-guided, including video lessons – the Robotics Activities include lessons on topics that are the foundation for both levels of coding in RoboBlockly.

Math

- Number Sense – addition and subtraction with positive whole numbers.
- Algebra:
  - Solving basic equations.
- Geometry:
  - Coordinate plane, all problems are completed on a coordinate plane. Students need to be comfortable understanding how to count, graph, and locate coordinates on a grid.
  - Properties of basic Geometric shapes: circle, rectangle, squares and triangles.

Computer

- Basic computer skills:
  - Drag and drop using a mouse
  - Key boarding
  - Navigating a web browser
  - Zoom In/Out in a browser
  - Disabling or enabling pop-up windows
  - Adjusting volume for videos

Extension

Using hardwired robots, Linkbot Controller, RoboSim or Robot Controller to execute programs built in RoboBlockly.

All can be downloaded from the UC Davis C-STEM Center’s webpage: [http://c-stem.ucdavis.edu/downloads/](http://c-stem.ucdavis.edu/downloads/)
# Learning Coding and Math with RoboBlockly

*Teaching Resource for Grade > 7 Coding Activities*

<table>
<thead>
<tr>
<th>Activity</th>
<th>Blocks Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Drive Distance: Moving Forward</td>
<td>driveDistance</td>
</tr>
<tr>
<td>2. Turn Left/Right</td>
<td>driveDistance, turnLeft, turnRight</td>
</tr>
<tr>
<td>3. Debugging a Program: Running Step by Step to Find and Fix the Error</td>
<td>turnRight, driveDistance, turnLeft</td>
</tr>
<tr>
<td>4. Using User Input to Assign Variables in Order to Solve a Problem</td>
<td>driveDistance, variable, prompt for “”</td>
</tr>
<tr>
<td>5. Outputting Values to the User</td>
<td>driveDistance, turnRight, getxy, print “”</td>
</tr>
<tr>
<td>6. Get the Position of a Robot on the Coordinate Plane</td>
<td>getxy, driveDistance, turnRight, print</td>
</tr>
<tr>
<td>7. Loops: Moving the Robot a Distance Repeatedly</td>
<td>while loop, driveDistance</td>
</tr>
<tr>
<td>8. Loops: Drawing Shapes - Square</td>
<td>while loop, driveDistance, turnRight</td>
</tr>
<tr>
<td>9. Loops: Drawing Shapes Part II – Using a Variable</td>
<td>count with, driveDistance, turnRight</td>
</tr>
</tbody>
</table>
### Table of Contents: Activity to Block Alignment (cont.)

<table>
<thead>
<tr>
<th>Activity</th>
<th>Blocks Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>10. Using RobotBlockly to Solve Simple Mathematical Expression</td>
<td>variable, driveDistance, squareroot, +/- block, print “”</td>
</tr>
<tr>
<td>11. Drawing Geometric Shapes</td>
<td>fill color, rectangle, triangle, circle</td>
</tr>
<tr>
<td>12. Animations – Make the Ball Bounce</td>
<td>repeat, count with, background color, fill color, circle, delay</td>
</tr>
<tr>
<td>13. Assign, Store and Print Random Integers</td>
<td>variable, print, driveDistance</td>
</tr>
<tr>
<td>14. If Statements: Using Random Integers to Drive the Robot a Specified Distance</td>
<td>variable, print, if, driveDistance</td>
</tr>
<tr>
<td>15. If-else Statements: Using Random Integers to Drive the Robot a Specified Distance &amp; Direction</td>
<td>variable, print, if/do/else, driveDistance</td>
</tr>
<tr>
<td>16. Else-If Statements: Using Random Integers to Drive the Robot to a Specified xy Coordinate</td>
<td>variable, print, if/elseif, driveDistance, turnRight</td>
</tr>
</tbody>
</table>
### Table of Contents: Activity to Block Alignment (cont.)

<table>
<thead>
<tr>
<th>Activity</th>
<th>Blocks Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>17. Logical “and” Statements: Using Random Integers to Drive the Robot to a Specified xy Coordinate</td>
<td>variable, and, print, if/elseif, driveDistance, turnRight</td>
</tr>
<tr>
<td>18. Logical “or” Statements: Using Random Integers to Drive the Robot to a Specified xy Coordinate</td>
<td>variable, or, print, if/else, driveDistance, turnRight</td>
</tr>
<tr>
<td>19. Function: Creating a Square</td>
<td>function, createsquare, driveDistance, turnRight, loop, traceOn/Off</td>
</tr>
<tr>
<td>20. Function: Creating a Circle</td>
<td>function, createcircle, driveDistance, turnRight, loop, traceOn/Off</td>
</tr>
<tr>
<td>21. Function: Creating a Rectangle</td>
<td>function, createsrectangle, driveDistance, turnRight, loop, traceOn/Off</td>
</tr>
<tr>
<td>22. Function: Calculating the Area of a Rectangle Using a Function with a Return Value</td>
<td>function, createsrectangle, driveDistance, turnRight, loop, traceOn/Off</td>
</tr>
</tbody>
</table>
Table of Contents: Textbook to Activity Alignment

<table>
<thead>
<tr>
<th>Section 5.1</th>
<th>Use Variables</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 5.2</td>
<td>The Output Function printf()</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 5.4</td>
<td>Move a Distance for a Two-Wheel Robot</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 5.6</td>
<td>Turn Left and Turn Right</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 10.1</td>
<td>Move a Linkbot-I in a Coordinate System</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Appendix A.1</td>
<td>Make a Decision Using and if and else if Statements</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Appendix A.2</td>
<td>Use a while-loop for Repeated Motions</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table of Contents: Textbook to Activity Alignment
### Learning Computer Programming with Ch for the Absolute Beginner

<table>
<thead>
<tr>
<th>Section</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 2.1 Commonly Used Data Types and Declaration of Variables</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 2.2.1 The Output Function printf()</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 2.2.2 The Input Function scanf()</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.2.1 The if Statement</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.2.2 The if-else Statement</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.3 Logical Operators</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.4 The else if Selection Statement</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 6.1 The while Loop</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 6.4 Control of Repetition</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 6.5 the for Loop</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section 6.8 Random Number Generation</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Chapter 10 Quick Animation</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Activity #1 Drive Distance: Moving Forward

Objective: Students will use the driveDistance block to move the robot a specified distance.

Math Skills: Addition of positive numbers. Familiar with coordinate plane.

RoboBlockly Student Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C1. Getting Started with Coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>Drive Distance</td>
</tr>
<tr>
<td></td>
<td><img src="driveDistance.png" alt="driveDistance(distance 5 in);" /></td>
</tr>
<tr>
<td></td>
<td>This block drives the robot forward with the distance specified by the first argument.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td><img src="driveDistance.png" alt="driveDistance(distance 5 in);" /></td>
</tr>
<tr>
<td>Problem Statement</td>
<td>Drive me to y = 5. Then, get me to y = 9.</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>I did not get to where I wanted to go. Please try again.</td>
</tr>
<tr>
<td>Hint</td>
<td><img src="driveDistance.png" alt="driveDistance(distance 5 in);" /></td>
</tr>
<tr>
<td></td>
<td>This problem is equal to 5 + ___ = 9.</td>
</tr>
<tr>
<td></td>
<td>You can uncheck Show Robot to view the distance on the coordinate system.</td>
</tr>
<tr>
<td>Possible Solution in C</td>
<td><code>#include &lt;linkbot.h&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>CLinkbotI robot;</code></td>
</tr>
<tr>
<td></td>
<td><code>double radius = 1.75;</code></td>
</tr>
<tr>
<td></td>
<td><code>robot.driveDistance(5, radius);</code></td>
</tr>
<tr>
<td></td>
<td><code>robot.driveDistance(4, radius);</code></td>
</tr>
</tbody>
</table>
Activity #1 Drive Distance: Moving Forward

Problem Statement:
Drive me to y=5. Then, get me to y = 9.

Location of solution for “Load Blocks” tab in RoboBlockly
C-STEM Studio -> Teaching Resources -> TeachCh -> RoboBlocklySolution -> CodingLevel2 -> c1.xml

C-STEM text alignment: *Robot Programming with Linkbot for the Absolute Beginner, 5th edition*

a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
Activity #2 Turn Left/Right

Objective: Students will drive the robot to an ordered pair in the coordinate plane using the driveDistance and turnLeft/Right block.

Math Skills: Familiar with coordinate plane, ordered pairs and plotting points.

RoboBlockly Activity:

Initial Student Prompt

Turn Left/Right

This block changes the direction of the robot by the amount of degrees specified by the first argument.

The track width is the distance between the two wheels as shown below.

Pre-Placed Blocks

Problem Statement
Drive me to (-4, 9).

Wrong Prompt
You did not get me to (-4, 9). Please try again.

Hint
This problem is equivalent to drive up 9 units and left 4 units.

Possible Solution 1 in C

```c
#include <linkbot.h>
CLinkbot1 robot;
double radius = 1.75;
double trackwidth = 3.69;

robot.driveDistance(9, radius);
robot.turnLeft(90, radius, trackwidth);
robot.driveDistance(4, radius);
```
Possible Solution 2 in C

```c
#include <linkbot.h>
CLinkbotI robot;
double radius = 1.75;
double trackwidth = 3.69;

robot.turnLeft(90, radius, trackwidth);
robot.driveDistance(4, radius);
robot.turnRight(90, radius, trackwidth);
robot.driveDistance(9, radius);
```
Activity #2 Turn Left/Right

Picture of solution in RoboBlockly

Problem Statement:
Drive me to (-4, 9).

- `driveDistance(distance 9 in);`
- `turn Left (angle 90°);`
- `driveDistance(distance 4 in);`

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh -> RoboBlocklySolution -> CodingLevel2 -> c2.xml

C-STEM text alignment: *Robot Programming with Linkbot for the Absolute Beginner, 5th edition*
- a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
- b) Section 5.6 Turn Left and Turn Right (turn block)
- c) Section 2.2 Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.
Activity #3 Debugging a Program: Running Step-by-Step to Find and Fix the Error

Objective: Students will apply their understanding of debugging to run the RoboBlockly code step-by-step to determine what blocks need to be changed in order to make the robot according to the problem statement.

Math Skills: Familiar with coordinate plane, ordered pairs and plotting points.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C2. Debugging a Program Step-by-Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>Debugging a Program: Step by Step Analysis</td>
</tr>
<tr>
<td></td>
<td>This button allows you to run your blockly code one block at a time. When the code lights up orange, the robot will perform the action related to that block. You can use this function to locate, analyze and correct errors “bugs” in programming.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td>![Image of pre-placed blocks]</td>
</tr>
<tr>
<td>Problem Statement</td>
<td>Use the Step function to help you “debug” the pre-placed blocks and move Linkbot/Lego Mindstorm along the colored path shown on the coordinate grid.</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>You have not fixed all the “bugs” in the pre-placed blocks. Remember to run the code one line at a time using the step button.</td>
</tr>
<tr>
<td>Hint</td>
<td>Remember to run the program step by step. The robot will turn right or left based on the way it is facing. The initial move of the robot is up, along the positive y axis.</td>
</tr>
</tbody>
</table>
| Possible Solution in C | ```c
#include <linkbot.h>
CLinkbotI robot;
double radius = 1.75;
double trackwidth = 3.69;

robot.turnLeft(90, radius, trackwidth);
robot.driveDistance(8, radius);
robot.turnLeft(90, radius, trackwidth);
``` |
```javascript
robot.driveDistance(5, radius);
robot.turnLeft(90, radius, trackwidth);
robot.driveDistance(12, radius);
robot.turnLeft(90, radius, trackwidth);
robot.driveDistance(8, radius);
robot.turnRight(90, radius, trackwidth);
robot.driveDistance(6, radius);
```
Activity #3 Debugging a Program: Running Step-by-Step to Find and Fix the Error

Problem Statement:
Use the Step function to help you “debug” the pre-placed blocks and move Linkbot/Lego Mindstorm along the colored path shown on the coordinate grid.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c3.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
b) Section 5.5 Number Line for Distance. (driveDistance block)
c) Section 5.6 Turn Left and Turn Right (turn block)

Section 2.2 Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.
Activity #4 Using User Input to Assign Variables in Order to Solve a Problem

Objective: Students will use the prompt and variable blocks to assign inputted values to make the robot move to a location on the coordinate plane.

Math Skills: Can identify distance in the coordinate plane from the origin to a preplaced ordered pair.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C3. Using Variables to Hold Different Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>If you want to use variables in Roboblockly use the [variable] block. You can also ask the user to input values for your variable by using [prompt for variable ] with message [“”]. Let’s try it!</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td></td>
</tr>
<tr>
<td>Problem Statement</td>
<td>The first move of the robot will not reach the green dot. You will be prompted to enter in the remaining distance the robot needs to travel to reach the green dot, once the program begins running. Before you run the program, you will need to add in one purple variable blocks and set the value to ‘distance’.</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>You did not get the robot to the green dot. Please try again.</td>
</tr>
<tr>
<td>Hint</td>
<td>You will need to add in one purple variable blocks. Make sure to change the value to ‘distance’</td>
</tr>
<tr>
<td>Possible Solution in Ch</td>
<td>#include &lt;linkbot.h&gt; double distance; CLinkbotI robot; double radius = 1.75; robot.driveDistance(5, radius); printf(&quot;How far should the robot drive in inches?\n&quot;); scanf(&quot;%lf&quot;, &amp;distance); robot.driveDistance(distance, radius);</td>
</tr>
</tbody>
</table>
Activity #4 Using User Input to Assign Variables in Order to Solve a Problem

Problem Statement:
The first move of the robot will not reach the green dot. You will be prompted to enter in the remaining distance the robot needs to travel to reach the green dot, once the program begins running. Before you run the program, you will need to add in one purple variable blocks and set the value to ‘distance’.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c4.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
a) Section 5.1 Use Variables (set variable block)
b) Section 5.2 The Output Function printf() (print variable block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
b) Section 2.22 The input Function scanf() (prompt for “” block)
**Activity #5 Outputting Values to the User**

**Objective:** Students will drive the robot to a specified point on the coordinate plane and output the x- & y-values to the user.

**Math Skills:** Geometry – ability to locate an ordered pair on the coordinate plane, determine distance in the coordinate plane and understand the meaning of perimeter.

**RoboBlockly Activity:**

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C4. Displaying the Values of Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initial Student Prompt</strong></td>
<td>Print, variables, and random numbers</td>
</tr>
<tr>
<td></td>
<td>is a block that outputs what is passed into it. It can be used with integers, decimal numbers, expressions, strings, and variables.</td>
</tr>
<tr>
<td><strong>Pre-Placed Blocks</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Problem Statement</strong></td>
<td>Insert additional driveDistance and turn blocks to drive the robot to the blue dot. You must move around the perimeter of the pink rectangles. Use the print variable &quot;&quot; blocks to output the x and y values for the blue dot.</td>
</tr>
<tr>
<td><strong>Wrong Prompt</strong></td>
<td>You did not output the correct coordinates for the blue dot. Please try again.</td>
</tr>
<tr>
<td><strong>Hint</strong></td>
<td>After programming the robot to drive to the blue dot you need to change the “variable” in the print block to say x and y, the values you want to print out.</td>
</tr>
<tr>
<td><strong>Possible Solution in C</strong></td>
<td></td>
</tr>
</tbody>
</table>
  ```
  #include <linkbot.h>
  double x;
  double y;
  CLinkbotI robot;
  double radius = 1.75;
  double trackwidth = 3.69;

  robot.driveDistance(4, radius);
  robot.turnRight(90, radius, trackwidth);
  robot.driveDistance(6, radius);
  robot.turnLeft(90, radius, trackwidth);
  robot.driveDistance(3, radius);
  robot.turnRight(90, radius, trackwidth);
  ``` |
<table>
<thead>
<tr>
<th>robot.driveDistance(3, radius);</th>
</tr>
</thead>
<tbody>
<tr>
<td>robot.turnRight(90, radius, trackwidth);</td>
</tr>
<tr>
<td>robot.driveDistance(3, radius);</td>
</tr>
<tr>
<td>robot.turnLeft(90, radius, trackwidth);</td>
</tr>
<tr>
<td>robot.driveDistance(4, radius);</td>
</tr>
<tr>
<td>robot.turnRight(90, radius, trackwidth);</td>
</tr>
<tr>
<td>robot.driveDistance(1, radius);</td>
</tr>
<tr>
<td>robot.getxy(x, y);</td>
</tr>
<tr>
<td>printf(&quot;x is %g \n&quot;, x);</td>
</tr>
<tr>
<td>printf(&quot;y is %g \n&quot;, y);</td>
</tr>
</tbody>
</table>
Activity #5 Outputting Values to the User

Problem Statement:
Insert additional driveDistance and turn blocks to drive the robot to the blue dot. You must move around the perimeter of the pink rectangles. Use the print variable "" blocks to output the x and y values for the blue dot.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c5.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
  a) Section 5.1 Use Variables (set variable block)
  b) Section 5.2 The Output Function printf() (print variable block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
  a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
  b) Section 2.2.1 The Output Function printf() (print variable and "" block)
Coding Level 2 Activities with RoboBlockly

Activity #6 Get the Position of a Robot in the Coordinate Plane & String Outputs

Objective: Students will drive the robot to a specified coordinate and output the location of the robot \( x = \) and \( y = \).

Math Skills: Basic understanding of the coordinate plane and graphing and locating coordinates.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C5. Formatting the Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>Get ( xy )/Position of the Robot and Combine Inputs into a String</td>
</tr>
<tr>
<td>[ \text{get } { xy }; \text{ get } { \text{Position} }; ]</td>
<td>Both blocks are used to obtain information about the robot's position and/or orientation.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td>To combine several inputs into a string, use:</td>
</tr>
<tr>
<td>Problem Statement</td>
<td>Drive the robot 7 units to the right and 9 units up. Get the ( x ) and ( y ) coordinate of the robot and print them.</td>
</tr>
<tr>
<td>Check Your Answer</td>
<td>Did you get a pop up box with the position of the robot printed out as ( x = 7 ) and ( y = 9 )?</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>You did not correctly move the robot. Please try again.</td>
</tr>
<tr>
<td>Hint</td>
<td>To define more than 2 inputs, click ( ) to the left of the block and drag</td>
</tr>
</tbody>
</table>
them into the input blocks as shown:

Possible Solution in C

```c
#include <linkbot.h>
double x;
double y;
CLinkbotI robot;
double radius = 1.75;
double trackwidth = 3.69;
robot.driveDistance(9, radius);
robot.turnRight(90, radius, trackwidth);
robot.driveDistance(7, radius);
robot.getxy(x, y);
printf("x = %s, y = %s\n", x, y);
```

```c
#include <linkbot.h>
double x;
double y;
CLinkbotI robot;
double radius = 1.75;
double trackwidth = 3.69;
robot.driveDistance(9, radius);
robot.turnRight(90, radius, trackwidth);
robot.driveDistance(7, radius);
robot.getxy(x, y);
printf("x = %s, y = %s\n", x, y);
```
Activity #6 Get the Position of a Robot in the Coordinate Plane & String Outputs

Picture of solution in RoboBlockly

The page at roboBlockly.ucdavis.edu says:

\[ x = 7, \ y = 9 \]

Pop up window students will see.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c6.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
   a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
   b) Section 5.6 Turn Left and Turn Right (turn block)
   c) Section 5.1 Use Variables (x & y variable blocks)
   d) Section 10.1 Move a Linkbot-I in a Coordinate System (getxy block)
   e) Section 5.2 The Output Function printf() (print variable block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
   a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
   b) Section 2.2.1 The Output Function printf() (print variable and “” block)
**Activity #7** Loops: Moving the Robot a Specified Distance Repeatedly

**Objective:** Students will move the robot backwards and forwards 10 times using a while loop by inserting the driveDistance blocks.

**Math Skills:** Distance forward is a positive number, distance backwards is a negative number. Familiar with coordinate plane.

**RoboBlockly Activity:**

<table>
<thead>
<tr>
<th><strong>Video Lesson</strong></th>
<th>C6. Repeating with a Loop</th>
</tr>
</thead>
</table>

**Initial Student Prompt**

Loops  
A loop is a sequence of statements that may be carried out several times in succession.

**Pre-Placed Blocks**

![Repeat Block]

**Problem Statement**

Add blocks within the repeat loop to move 10 units forward and backward three times.

**Wrong Prompt**

You did not move me forward and backward 10 units. Please try again.

**Hint**

The blocks inserted should be the same except in opposite directions.

**Possible Solution in C#**

```csharp
#include <linkbot.h>
CLinkbot1 robot;
int count;
double radius = 1.75;

count = 0;
while(count < 3) {
    robot.driveDistance(10, radius);
    robot.driveDistance(-10, radius);
    count = count + 1;
}
Activity #7 Loops: Moving the Robot a Specified Distance Repeatedly

Picture of solution in RoboBlockly

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c7.xml

C-STEM text alignment: *Robot Programming with Linkbot for the Absolute Beginner, 5th edition*
  a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
  b) Appendix A.2 Use a while-loop for Repeated Motions (repeat block)
  c) Section 2.2 Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: *Learning Computer Programming with Ch for the Absolute Beginner*
  a) Section 6.1 The while Loop (repeat block)
Activity #8 Loops with Shapes Part I – Drawing a Square

Objective: Students will use a while loop and given distances to have the robot draw a square.

Math Skills: Knowledge of the basic features of Geometric shapes – square. Understand how to plot and locate points in the coordinate plane.

RoboBlockly Activity:

| Initial Student Prompt | Loops with Shapes Part 1  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A square is a shape with four equal sides and four equal angles.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pre-Placed Blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>repeat 4 times do</td>
</tr>
<tr>
<td>driveDistance(distance 7 in);</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a square with sides of length 7 using the coordinates (0,0), (7,0), (7,7), and (0, 7).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wrong Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>I did not get a square. Please try again.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hint</th>
</tr>
</thead>
<tbody>
<tr>
<td>A square contains 4 sides. Each of the angles is 90 degrees.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Possible Solution in C</th>
</tr>
</thead>
<tbody>
<tr>
<td>#include &lt;linkbot.h&gt;</td>
</tr>
<tr>
<td>CLinkbot robot;</td>
</tr>
<tr>
<td>int count;</td>
</tr>
<tr>
<td>double radius = 1.75;</td>
</tr>
<tr>
<td>double trackwidth = 3.69;</td>
</tr>
<tr>
<td>count = 0;</td>
</tr>
<tr>
<td>while(count &lt; 4) {</td>
</tr>
<tr>
<td>robot.driveDistance(7, radius);</td>
</tr>
<tr>
<td>robot.turnRight(90, radius, trackwidth);</td>
</tr>
<tr>
<td>count = count + 1;</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
Activity #8 Loops with Shapes Part I – Drawing a Square

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c8.xml

C-STEM text alignment: *Robot Programming with Linkbot for the Absolute Beginner, 5th edition*
   a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
   b) Section 5.6 Turn Left and Turn Right (turn block)
   c) Appendix A.2 Use a while-loop for Repeated Motions (repeat block)
   d) Section 2.2 Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: *Learning Computer Programming with Ch for the Absolute Beginner*
   a) Section 6.1 The while Loop (repeat block)
### Activity #9 Loops with Shapes Part II – Using a Variable

**Objective:** Students will define variables in a loop to draw a shape.

**Math Skills:** Knowledge of the basic features of Geometric shapes – rectangle. Understand how to plot and locate points in the coordinate plane.

**RoboBlockly Activity:**

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C7. Using a Variable with a Loop</th>
</tr>
</thead>
</table>
| **Initial Student Prompt** | Loops with Shapes Part 2  
A rectangle is similar to a square except that all 4 sides do not have to be equal, only opposite ones. |
| **Pre-Placed Blocks** | ![Pre-Placed Blocks](image) |
| **Problem Statement** | Create a rectangle of side lengths 5 and 10 using the count loop. |
| **Wrong Prompt** | I did not get a 5 by 10 rectangle. Please try again. |
| **Hint** | You need to add four blocks into the count loop, this will determine the width and height of your rectangle. You will also need to change the number of times the loop will run. Think about how many sets of congruent sides a rectangle has. |
| **Possible Solution in C** | ```
#include <linkbot.h>

double d;
CLinkbotI robot;
double radius = 1.75;
double trackwidth = 3.69;

for(d = 1; d <= 2; d++) {
    robot.driveDistance(10, radius);
    robot.turnRight(90, radius, trackwidth);
    robot.driveDistance(5, radius);
    robot.turnRight(90, radius, trackwidth);
} ``` |
Activity #9 Loops with Shapes Part II – Using a Variable

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c9.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
   a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
   b) Section 5.6 Turn Left and Turn Right (turn block)
   c) Section 2.2 Connect Linkbots from a Computer – have students move a hardwired robot by
      generating the Ch code using “Save Ch” on RoboBlockly.

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating
the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
   a) Section 6.5 the for Loop (count with block)
Activity #10 Using RoboBlockly to Solve Simple Mathematical Expressions

Objective: Students will use their understanding of assigning and formatting output from previous activities to have the robot drive a distance computed using math functions in Ch.

Math Skills: Understand how to calculate square root and multi-digit addition.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C8. Using Arithmetic Operators and Mathematical Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>You can use the math tab to access a variety of math operations. We will use these blocks to calculate the square root and perform arithmetic on specific values.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td></td>
</tr>
<tr>
<td>Problem Statement</td>
<td>Use the preplaced blocks to set a variable to the sum of the square root of 225 + 5. Drive the robot the distance of the solution by inserting a variable block for the value of driveDistance. Print out the answer in the following format: variable is 20 which is the sum of the sqrt of 225 plus 5. You will need to use the Math, Text and Variable tabs to find all the missing blocks.</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>The robot did not drive the distance (sqrt 225 + 5) and print out the correct solution.</td>
</tr>
<tr>
<td>Hint</td>
<td>You will need to add in 4 blocks to solve this problem. Two number blocks from the Math tab – you will need to set their values as specified in the problem. You will need to add a purple variable block into the driveDistance block and finally, you will need to add the print block with the text “which is the sum of the sqrt of 225 plus 5.”</td>
</tr>
</tbody>
</table>
| Possible Solution in Ch | #include <linkbot.h>  
double variable;  
CLinkbot robot;  
double radius = 1.75;  
variable = sqrt(225) + 5;  
robot.driveDistance(variable, radius);  
printf("variable is %g which is the sum of the sqrt of 225 and 5. \n", variable); |
Activity #10 Using RoboBlockly to Solve Simple Mathematical Expressions

Picture of solution in RoboBlockly

Problem Statement:
Use the preplaced blocks to set a variable to the sum of the square root of 225 + 5. Drive the robot the distance of the solution by using the variable block and print out the answer in the following format: variable is 20 which is the sum of the sqrt of 225 plus 5.

You will need to use the Math, Text and Variable tabs to find all the missing blocks.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c10.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
   b) Section 5.1 Use Variables (variable block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
   b) Section 2.2.1 The Output Function printf() (print variable and “” block)
Activity #11 Drawing Geometric Shapes

Objective: Students will use the drawing tab to draw a gray house with two windows, a red door and a blue roof. The robot will not move in this activity.

Math Skills: Geometric shapes: rectangle, triangle and circle. Coordinate Plane: locate points in the coordinate plane and apply these locations to simple Geometric shapes.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C9. Drawing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>Use the drawing tab to easily create simple geometric shapes or complex designs. For this activity the default to “show robot” has been turned off.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td>![Pre-Placed Blocks Diagram]</td>
</tr>
<tr>
<td>Problem Statement</td>
<td>Use the drawing tab to create a: Rectangular gray house with bottom left corner starting at the origin that is 10 ft tall and 15 feet wide. The roof will be a blue triangle. There needs to be two yellow windows and one red door. The only requirement is that the door is in between the two windows. Use this</td>
</tr>
</tbody>
</table>
**Hint**
Remember to place items you want to appear in the “background” first, if you draw the windows before the house you will not see them.

Place the blocks for color directly before the item you want to color.

**Possible Solution in Ch**

```c
#include <chplot.h>
CPlot plot;

plot.fillColor("cyan");
plot.rectangle(0, 0, 8.5, 9.5);
plot.fillColor("cyan");
plot.triangle(-1, 10, 7.5, 12, 16, 10);
plot.fillColor("red");
plot.rectangle(6, 0, 3, 5);
plot.fillColor("cyan");
plot.circle(3, 7, 1);
plot.circle(0, 0, 1);

plot.label(PLOT_AXIS_XY, "");
plot.grid(PLOT_OFF);
plot.tics(PLOT_AXIS_XY, PLOT_OFF);
plot.axis(PLOT_AXIS_XY, PLOT_OFF);
plot.axisRange(PLOT_AXIS_X, -12, 24);
plot.axisRange(PLOT_AXIS_Y, -12, 24);
plot.ticsRange(PLOT_AXIS_X, 6);
plot.ticsRange(PLOT_AXIS_Y, 6);
plot.sizeRatio(1);
plot.plotting();
```
Activity #11 Drawing Geometric Shapes

Picture of solution in RoboBlockly

Problem Statement:
Use the drawing tab to create a:
- Rectangular gray house with bottom left corner starting at the origin that is 10 ft tall and 15 feet wide.
- The roof will be a blue triangle.
- There needs to be two yellow windows and one red door. The only requirement is that the door is in between the two windows.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c11.xml

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
  a) Chapter 10 Quick Animation
Activity #12 Animations – Make the Ball Bounce

Objective: Students will combine their understanding of the drawing tab and loops to have a ball bounce. They will modify the pre-placed code according to the problem specifications.

Math Skills: Understands location in the coordinate plane and time.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C10. Animations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>You can create fun animations by combining your drawings with loops and functions you’ve already learned. For this activity you will bounce a ball using preplaced code – Let’s try it!</td>
</tr>
<tr>
<td>Pre-Placed Blocks [Diagram]</td>
<td></td>
</tr>
</tbody>
</table>
| Problem Statement | Modify the pre-placed blocks to produce the following animation:
- Have the ball bounce 8 times.
- Use a different color for the ball when it bounces up and for when it bounces down.
- Change the size of the ball – make it smaller.
- Change the starting position of the ball so it begins bouncing from the ground and reaches a height of 18.
- Change the speed of the bounce. |
**Hint** You are only modifying the preplaced blocks. You do not need to add or remove any additional blocks.

To change the speed of the bounce you change how fast the count loop is counting by. To change the number of times the ball bounces you change the repeat loop. The first count loop moves the ball up the second count loop moves the ball down.

**Possible Solution in Ch**

```c
#include <chplot.h>
double d;
CPlot plot;
int count;

count = 0;
while(count < 3) {
    // Move ball up
    for(d = 4; d <= 18; d += 0.3) {
        plot.backgroundColor("#999999");
        plot.fillColor("#ff9900");
        plot.strokeColor("orange");
        plot.circle(0, d, 4);
        delaySeconds(0.03);
    }
    // Move ball down
    for(d = 18; d >= 4; d -= 0.3) {
        plot.backgroundColor("#999999");
        plot.fillColor("#ff9900");
        plot.strokeColor("orange");
        plot.circle(0, d, 4);
        delaySeconds(0.03);
    }
    count = count + 1;
}

plot.label(PLOT_AXIS_XY, "");
plot.grid(PLOT_OFF);
plot.tics(PLOT_AXIS_XY, PLOT_OFF);
plot.axis(PLOT_AXIS_XY, PLOT_OFF);
plot.axisRange(PLOT_AXIS_X, -12, 24);
plot.axisRange(PLOT_AXIS_Y, -12, 24);
plot.ticsRange(PLOT_AXIS_X, 6);
plot.ticsRange(PLOT_AXIS_Y, 6);
plot.sizeRatio(1);
plot.plotting();
```
**Activity #12 Animations – Bounce a Ball**

**Problem Statement:**
Modify the pre-placed blocks to produce the following animation:

- Have the ball bounce 8 times.
- Use a different color for the ball when it bounces up and for when it bounces down.
- Change the size of the ball – make it smaller.
- Change the starting position of the ball so it begins bouncing from the ground and reaches a height of 18.
- Change the speed of the bounce.

**Location of Solution for “Load Blocks”**
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c12.xml

**C-STEM text alignment:** *Robot Programming with Linkbot for the Absolute Beginner, 5th edition*
  a) Appendix A.2 Use a while-loop for Repeated Motions (repeat block)

**Extension:** Connect Linkbots from a Computer – have students move a hardwired robot by generating
the Ch code using “Save Ch” on RoboBlockly.

**C-STEM text alignment:** *Learning Computer Programming with Ch for the Absolute Beginner*
  b) Section 6.1 The while Loop (repeat block)
  c) Section 6.4 Control of Repetition (count with block)
  d) Chapter 10 Quick Animation
**Activity #13 Assign, Store and Print, Random Integers**

**Objective:** Students will use the print and set blocks to define a variable value to be a random integer from 1 to 100 and output the value in a pop-up window.

**Math Skills:** Understands what it means to generate a number at random from 1 to 100.

**RoboBlockly Activity:**

<table>
<thead>
<tr>
<th><strong>Video Lesson</strong></th>
<th>C11. Generating Random Numbers</th>
</tr>
</thead>
</table>
| **Initial Student Prompt** | Print, variables, and random numbers.  
  [print](variable) and [“”] is a block that outputs what is passed into it. It can be used with integers, decimal numbers, expressions, strings, and variables.  
  [random integer from (1 to 100)] returns a random integer bounded by its two arguments. |
| **Pre-Placed Blocks**  | [set](variable) to [ ]  
  [print](variable) and [“”] |
| **Problem Statement**  | Store a random integer from 1 to 100 into a variable and then print that variable. (Your robot will not move.) |
| **Check Your Answer**  | Did you get a popup window with a random integer from 1 to 100? |
| **Hint**               | None. |
| **Possible Solution in C** | double variable;  
variable = randInt(1, 100);  
printf("variable is %g \n", variable); |
Activity #13 Assign, Store and Print, Random Integers

Picture of solution in RoboBlockly
No Grid as the robot will not move.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c13.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
b) Section 5.1 Use Variables (set variable block)
c) Section 5.2 The Output Function printf() (print variable block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
e) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
f) Section 2.2.1 The Output Function printf() (print variable and “” block)
g) Section 6.8 Random Number Generation (random integer from 1 to 100 block)
Activity #14 If Statements: Driving the Robot a Random Integer

Objective: Students will assign a variable a random integer value from 1 to 100, and if the integer generated is even, the robot will drive forward 5 units – otherwise the robot will not move.

Math Skills: Number sense – knows the difference between odd and even numbers from 1-100.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C12. Making Decisions Using if-Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>The if statement executes a sequence of statements depending on whether the condition is true.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td>![Pre-Placed Blocks Image]</td>
</tr>
<tr>
<td>Problem Statement</td>
<td>Insert a condition so that the robot only drives when the random number between 1 and 100 is an even number.</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>You did not place the right condition in the if statement. Please try again.</td>
</tr>
<tr>
<td>Hint</td>
<td>There are two blocks that can help you with this problem: both are located in the Math tab.</td>
</tr>
</tbody>
</table>
| Possible Solution in C | #include <linkbot.h>
double variable;
CLinkbotI robot;
double radius = 1.75;

variable = randint(1, 100);
printf("variable is %g \n", variable);
if (variable % 2 == 0) {
    robot.driveDistance(5, radius);
} |
Activity #14 If Statements: Driving the Robot a Random Integer

Picture of solution in RoboBlockly
If the random integer is odd, the robot will not move. If the random integer is even, the robot will move.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c14.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
  a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
  b) Section 5.1 Use Variables (set variable block)
  c) Section 5.2 The Output Function printf() (print variable block)
  d) Appendix A.1 Make a Decision Using and if and else if Statements (if/do block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating
the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
  a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
  b) Section 2.2.1 The Output Function printf() (print variable and “” block)
  c) 5.2.1 The if Statement (if do block)
  d) Section 6.8 Random Number Generation (random integer from 1 to 100 block)
Activity #15 If-Else Statements: Driving a Robot Based on a Random Integer

Objective: Students will assign a variable a random integer value from 1 to 100, and if the integer generated is greater than 50, the robot will drive forward 5 units – otherwise the robot will move backwards.

Math Skills: Number sense – knows how to determine if a number is larger than 50.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C13. Making Decision Using if-else if else Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>The else statement else must be after an if statement and it is only executed if the corresponding if statement fails.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td>Insert blocks so that the robot moves forward 5 units when a random number between 1 and 100 is greater than 50. Otherwise, the robot moves backwards.</td>
</tr>
<tr>
<td>Problem Statement</td>
<td></td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>You did not insert the statements correctly. Please try again.</td>
</tr>
<tr>
<td>Hint</td>
<td>The else statement takes care of all other cases.</td>
</tr>
</tbody>
</table>
| Possible Solution in C | #include <linkbot.h>  
double variable;  
CLinkbotI robot;  
double radius = 1.75;  
variable = randin(1, 100);  
printf("variable is %g \n", variable);  
if (variable > 50) {  
  robot.driveDistance(5, radius);  
} else {  
  robot.driveDistance(-5, radius);  
} |
Activity #15 If-Else Statements: Driving a Robot Based on a Random Integer

Picture of solution in RoboBlockly

Since the number is even, the robot moves forward 5 units.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh -> RoboBlocklySolution -> CodingLevel2 -> c15.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
   a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
   b) Section 5.1 Use Variables (set variable block)
   c) Section 5.2 The Output Function printf() (print variable block)
   d) Appendix A.1 Make a Decision Using and if and else if Statements (if/do/else block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
   a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
   b) Section 2.2.1 The Output Function printf() (print variable and ‘”’ block)
   c) 5.2.2 The if-else Statement (if-else block)
   d) Section 6.8 Random Number Generation (random integer from 1 to 100 block)
Activity #16 Else-If Statements: Driving a Robot to Coordinates Based on a Random Integer

Objective: Students will assign a variable a random integer value from 1 to 100, and if the integer generated is less than 30, the robot will drive to the coordinate (0, 5), if is equal or greater than 70, the robot will drive to the coordinate (5, 5) – otherwise the robot will move to (5, 0).

Math Skills: Number sense – knows how to compare numbers and determine which is greater or smaller. Can graph, and locate points on the coordinate plane.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Initial Student Prompt</th>
<th>The else if statement expands conditional statements by executing the statements after the first condition is found to be true.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-Placed Blocks</td>
<td>Pre-Placed Blocks</td>
</tr>
<tr>
<td>Problem Statement</td>
<td>Problem Statement: Modify the if statement so that the robot travels to (0, 5) if a random number between 1 and 100 is less than 30, to (5, 5) if it is equal or greater than 70, otherwise, to (5, 0).</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>Wrong Prompt: You did not modify the if statement correctly. Please try again.</td>
</tr>
<tr>
<td>Hint</td>
<td>Hint: Read the conditions carefully and implement them using the Logic and Math tabs.</td>
</tr>
<tr>
<td>Possible Solution in C</td>
<td>Possible Solution in C</td>
</tr>
</tbody>
</table>

```c
#include <linkbot.h>
double variable;
CLinkbot robot;
double radius = 1.75;
double trackwidth = 3.69;

variable = randint(1, 100);
printf("variable is %g \n", variable);
if (variable < 30) {
    robot.driveDistance(5, radius);
} else if (variable >= 70) {
    robot.turnRight(45, radius, trackwidth);
    robot.driveDistance(7.07, radius);
} else {
    robot.turnRight(90, radius, trackwidth);
    robot.driveDistance(5, radius);
}
```
Activity #16 Else-If Statements: Driving a Robot to Coordinates Based on a Random Integer

Picture of solution in RoboBlockly

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh -> RoboBlocklySolution -> CodingLevel2 -> c16.xml

C-STEM text alignment: *Robot Programming with Linkbot for the Absolute Beginner, 5th edition*
   a) Section 5.4 Move a Distance for a Two-Wheel Robot (driveDistance block)
   b) Section 5.6 Turn Left and Turn Right (turn block)
   c) Section 5.1 Use Variables (set variable block)
   d) Section 5.2 The Output Function printf() (print variable block)
   e) Appendix A.1 Make a Decision Using if and else if Statements (if/do block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: *Learning Computer Programming with Ch for the Absolute Beginner*
   a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
   b) Section 2.2.1 The Output Function printf() (print variable and “” block)
   c) 5.2.1 The if Statement (if do block)
   d) 5.4 The else if Selection Statement (else if block)
   e) Section 6.8 Random Number Generation (random integer from 1 to 100 block)
Activity #17 If Statements Combined with Logical Arguments

Objective: Students will assign a variable a random integer value from 1 to 100, and if the integer generated is greater than 50 and even, the robot will drive to the coordinate (0, 5), if is odd and less than 50, the robot will drive to the coordinate (5, 5) – otherwise the robot will move to (5, 0).

Math Skills: Number sense – knows how to compare numbers and determine which is greater or smaller and if a number is even or odd. Can graph, and locate points on the coordinate plane.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Lesson Video</th>
<th>C14. Making Decisions Using Logical Operations AND and OR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Student Prompt</td>
<td>The logical and takes in two boolean values and returns true if both values are true, otherwise false.</td>
</tr>
<tr>
<td>Pre-Placed Blocks</td>
<td>Modify the if statement so that the robot travels to (0, 5) if a random number between 1 and 100 is even and greater than 50, to (5, 5) if the variable is odd and less than 50, otherwise, to (5,0).</td>
</tr>
<tr>
<td>Wrong Prompt</td>
<td>You did not modify the if statement correctly. Please try again.</td>
</tr>
<tr>
<td>Hint</td>
<td>Read the conditions carefully and implement them using the Logic and Math tabs.</td>
</tr>
<tr>
<td>Possible Solution in Ch</td>
<td>#include &lt;linkbot.h&gt;</td>
</tr>
<tr>
<td></td>
<td>double variable;</td>
</tr>
<tr>
<td></td>
<td>CLinkbot robot;</td>
</tr>
<tr>
<td></td>
<td>double radius = 1.75;</td>
</tr>
<tr>
<td></td>
<td>double trackwidth = 3.69;</td>
</tr>
<tr>
<td></td>
<td>variable = randint(1, 100);</td>
</tr>
<tr>
<td></td>
<td>printf(&quot;variable is %g \n&quot;, variable);</td>
</tr>
<tr>
<td></td>
<td>if (variable % 2 == 0 &amp;&amp; variable &gt; 50) {</td>
</tr>
<tr>
<td></td>
<td>robot.driveDistance(5, radius);</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>else if (variable % 2 == 1 &amp;&amp; variable &lt; 50) {</td>
</tr>
<tr>
<td></td>
<td>robot.turnRight(45, radius, trackwidth);</td>
</tr>
<tr>
<td></td>
<td>robot.driveDistance(7.07, radius);</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>else {</td>
</tr>
<tr>
<td></td>
<td>robot.turnRight(90, radius, trackwidth);</td>
</tr>
<tr>
<td></td>
<td>robot.driveDistance(5, radius);</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
Activity #17 If Statements Combined with Logical Arguments

Picture of solution in RoboBlockly

Since the random integer generated is 64, the robot drove to (0, 5) because 64 is even and larger than 50.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh -> RoboBlocklySolution -> CodingLevel2 -> c17.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
- Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
- Section 5.6 Turn Left and Turn Right (turn block)
- Section 5.1 Use Variables (set variable block)
- Section 5.2 The Output Function printf() (print variable block)
- Appendix A.1 Make a Decision Using if and else if Statements (if/do block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
- Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
- Section 2.2.1 The Output Function printf() (print variable and “” block)
- 5.2.1 The if Statement (if do block)
- 5.3 Logical Operators (logical block)
- 5.4 The else if Selection Statement (else if block)
- Section 6.8 Random Number Generation (random integer from 1 to 100 block)
### Activity #18 If Statements Combined with Logical Arguments

**Objective:** Students will assign a variable a random integer value from 1 to 100, and if the integer generated is even or less than 50, otherwise the robot will move to (5, 0).

**Math Skills:** Number sense – knows how to compare numbers and determine which is greater or smaller and if a number is even or odd. Can graph, and locate points on the coordinate plane.

**RoboBlockly Activity:**

<table>
<thead>
<tr>
<th>Initial Student Prompt</th>
<th>The logical or [\text{or} ] takes in two boolean values and returns true if either values are true, otherwise false.</th>
</tr>
</thead>
</table>
| **Pre-Placed Blocks**  | \begin{tabular}{|c|c|}
|                        | \begin{tabular}{|c|c|}
| set variable to       | \begin{tabular}{|c|}
| variable from         | \begin{tabular}{|c|}
| 1 to 100             | \end{tabular}
| \end{tabular}
| \end{tabular}
| if variable > 50     | \begin{tabular}{|c|}
| do                    | \end{tabular}
| \end{tabular}
| \end{tabular}
| \end{tabular} |
| **Problem Statement** | Modify the if statement so that the robot travels to (0, 5) if a random number between 1 and 100 is even or less than 50, otherwise, to (5, 0). |
| **Wrong Prompt**      | You did not modify the if statement correctly. Please try again. |
| **Hint**              | Read the conditions carefully and implement them using the Logic and Math tabs. |
| **Possible Solution in Ch** | \begin{tabular}{|c|}
| \#include <linkbot.h> | \end{tabular}
| \begin{tabular}{|c|}
| double variable; \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| CLinkbot robot; \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| double radius = 1.75; \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| double trackwidth = 3.69; \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| variable = randint(1, 100); \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| printf("variable is %g \n", variable); \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| if (variable % 2 == 0 || variable < 50) { \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| robot.driveDistance(5, radius); \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| } \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| else { \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| robot.turnRight(90, radius, trackwidth); \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| robot.driveDistance(5, radius); \end{tabular} | \end{tabular}
| \begin{tabular}{|c|}
| } \end{tabular} | \end{tabular} |
**Activity #18 If Statements Combined with Logical Arguments**

Since the random integer generated is 16, the robot drove to (0, 5) because 16 is even or less than 50.

**Location of Solution for “Load Blocks”**
C-STEM Studio -> Teaching Resources -> TeachCh-> RoboBlocklySolution-> CodingLevel2->c18.xml

**C-STEM text alignment:** *Robot Programming with Linkbot for the Absolute Beginner, 5th edition*
- a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
- b) Section 5.6 Turn Left and Turn Right (turn block)
- c) Section 5.1 Use Variables (set variable block)
- d) Section 5.2 The Output Function printf() (print variable block)
- e) Appendix A.1 Make a Decision Using and if and else if Statements (if/do block)

**Extension:** Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

**C-STEM text alignment:** *Learning Computer Programming with Ch for the Absolute Beginner*
- a) Section 2.1 Commonly Use Data Types and Declaration of Variables (set variable block)
- b) Section 2.2.1 The Output Function printf() (print variable and “” block)
- c) 5.2.2 The if-else Statement (if-else block)
- d) 5.3 Logical Operators (logical block)
- e) Section 6.8 Random Number Generation (random integer from 1 to 100 block)
Activity #19 Functions: Creating a Square

Objective: Students will use a function to create a square with given side lengths, centered at a specified location in the coordinate plane.

Math Skills: Knowledge of the basic features of Geometric shapes – square. Understand how to plot and locate points in the coordinate plane.

RoboBlockly Activity:

<table>
<thead>
<tr>
<th>Video Lesson</th>
<th>C15. Creating Functions</th>
</tr>
</thead>
</table>
| Initial Student Prompt | Function
| Pre-Placed Blocks | createSquare |

Problem Statement

The createSquare function will draw a square with side lengths of 4, centered on a specific ordered pair. Use the createSquare function to draw two squares one centered at (3, 3) and the other centered at (9, 12).

For a challenge add trace blocks to only have the robot draw the two squares without any other marks on the grid.

Wrong Prompt

I did not get two squares at those places. Please try again.

Hint

The createSquare function will automatically create a square with a side length of 4 units. You need to use the driveToxy block to tell the program the center for each square, then place the createSquare function block after.

Possible Solution in C

```c
#include <linkbot.h>

CLinkbotI robot;

double radius = 1.75;
```
double trackwidth = 3.69;
int count;

void createSquare() {
    robot.traceOff();
    robot.driveDistance(2, radius);
    robot.turnRight(90, radius, trackwidth);
    robot.traceOn();
    count = 0;
    while(count < 4) {
        robot.driveDistance(2, radius);
        robot.turnRight(90, radius, trackwidth);
        robot.driveDistance(2, radius);
        count = count + 1;
    }
    robot.traceOff();
    robot.drivexyTo(3, 3, radius, trackwidth);
    createSquare();
    robot.traceOff();
    robot.drivexyTo(9, 12, radius, trackwidth);
    createSquare();
}
Activity #19 Functions: Creating a Square

Problem Statement:
The createSquare function will draw a square with side lengths of 4, centered on a specific ordered pair. Use the createSquare function to draw two squares one centered at (3, 3) and the other centered at (9, 12). For a challenge add trace blocks to only have the robot draw the two squares without any other marks on the grid.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources ->TeachCh->RoboBlocklySolution-> CodingLevel2->c19.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
  a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
  b) Section 5.6 Turn Left and Turn Right (turn block)
  c) Appendix A.2 Use a while-loop for Repeated Motions (repeat block)
  d) Section 10.1 Move a Linkbot-I in a Coordinate System (driveTo block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
  a) Section 6.1 The while Loop (repeat block)
**Activity #20 Functions: Creating a Circle**

**Objective:** Students will use a function to create a circle, centered at a specified location in the coordinate plane.

**Math Skills:** Knowledge of the basic features of Geometric shapes – circle. Understand how to plot and locate points in the coordinate plane.

**RoboBlockly Activity:**

<table>
<thead>
<tr>
<th>Initial Student Prompt</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>draw a shape here</td>
<td>do something</td>
</tr>
<tr>
<td></td>
<td>do something</td>
</tr>
</tbody>
</table>

when called, execute the statements inside the function.

<table>
<thead>
<tr>
<th>Pre-Placed Blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>createCircle</td>
</tr>
<tr>
<td>createCircle</td>
</tr>
<tr>
<td>createCircle</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem Statement</th>
</tr>
</thead>
</table>
| The `createCircle` function will draw a circle with a radius of 7, centered on a specific ordered pair. Use the `createCircle` function to draw two circles centered at: (0,0) and (12,0).

For a challenge add trace blocks to only have the robot draw the four circles without any other marks on the grid.

<table>
<thead>
<tr>
<th>Wrong Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>I did not get two circles at those places. Please try again.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hint</th>
</tr>
</thead>
<tbody>
<tr>
<td>The function creates a circle of radius 7. Its center is determined by the coordinates of the robot before the function is called.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Possible Solution in C</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#include &lt;linkbot.h&gt;</code></td>
</tr>
<tr>
<td><code>CLinkbotI robot;</code></td>
</tr>
<tr>
<td><code>double radius = 1.75;</code></td>
</tr>
<tr>
<td><code>double trackwidth = 3.69;</code></td>
</tr>
<tr>
<td><code>int count;</code></td>
</tr>
<tr>
<td><code>void createCircle() {</code></td>
</tr>
</tbody>
</table>
```
robot.traceOff();
robot.driveDistance(7, radius);
robot.turnRight(90, radius, trackwidth);
robot.traceOn();
count = 0;
while(count < 36) {
  robot.driveDistance(0.6101, radius);
  robot.turnRight(10, radius, trackwidth);
  robot.driveDistance(0.6101, radius);
  count = count + 1;
}
}

robot.setSpeed(8, radius);
circle();
robot.traceOff();
robot.drivexyTo(12, 0, radius, trackwidth);
circle();
```
Activity #20 Functions: Creating a Circle

Problem Statement:
The createCircle function will draw a circle with a radius of 7, centered on a specific ordered pair. Use the createCircle function to draw two circles centered at: (0,0) and (12,0). For a challenge add trace blocks to only have the robot draw the four circles without any other marks on the grid.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources ->TeachCh->RoboBlocklySolution-> CodingLevel2->c20.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
b) Section 5.6 Turn Left and Turn Right (turn block)
c) Appendix A.2 Use a while-loop for Repeated Motions (repeat block)
d) Section 10.1 Move a Linkbot-I in a Coordinate System (driveTo block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
a) Section 6.1 The while Loop (repeat block)
Activity #21 Functions: Creating a Rectangle using Arguments

Objective: Students will use a function with arguments to create a rectangle by assigning values in the function called arguments.

Math Skills: Knowledge of the basic features of Geometric shapes – rectangle. Understand how to plot and locate points in the coordinate plane.

RoboBlockly Activity:

**Initial Student Prompt**

Defining Arguments in a Function

When using a function you can assign values to the arguments by clicking on the blue icon and moving inputs into the block as shown above. Let’s try this to draw a rectangle.

**Pre-Placed Blocks**

Use the function block to create a three rectangles.

Rectangle 1: height = 12 units and width = 20 units.
Rectangle 2: height = 6 units and width = 10 units.
Rectangle 3: height = 3 units and width = 5 units.

You will first begin by assigning arguments in the create rectangle function.

Then you will need to assign values to the dimensions of your rectangles.
### Wrong Prompt
You did not draw the three rectangles correctly.

### Hint
Click on the blue icon on the create rectangle function. Drag two new inputs into the block and assign them values of height and width. Then you will need to open the variable tab and drag a block for height and for width into the driveDistance blocks in your loop. Finally you will need to assign values to your width and height using the math tab.

### Possible Solution in C
```c
#include <linkbot.h>
CLinkbotI robot;

double radius = 1.75;
double trackwidth = 3.69;
int count;

void create_rectangle(double height, double width) {
    count = 0;
    while(count < 2) {
        robot.driveDistance(height, radius);
        robot.turnRight(90, radius, trackwidth);
        robot.driveDistance(width, radius);
        robot.turnRight(90, radius, trackwidth);
        count = count + 1;
    }
}

create_rectangle(6, 10);
```
Activity #21 Functions: Creating a Rectangle

Problem Statement:
Use the function block to create a three rectangles.
Rectangle 1: height = 12 units and width = 20 units.
Rectangle 2: height = 6 units and width = 10 units.
Rectangle 3: height = 3 units and width = 5 units.
You will first begin by assigning arguments in the create rectangle function.
Then you will need to assign values to the dimensions of your rectangles.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh -> RoboBlocklySolution -> CodingLevel2 -> c21.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
b) Section 5.6 Turn Left and Turn Right (turn block)
c) Appendix A.2 Use a while-loop for Repeated Motions (repeat block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
a) Section 6.1 The while Loop (repeat block)
Activity #22 Functions: Calculating the Area of a Rectangle Using a Function with a Return Value

**Objective:** Students will use a function with arguments to create a rectangle by assigning values in the function called arguments. They will then prompt the user to enter values and get a return value.

**Math Skills:** Knowledge of the basic features of Geometric shapes – rectangle. Understand how to plot and locate points in the coordinate plane.

**RoboBlockly Activity:**

<table>
<thead>
<tr>
<th><strong>Initial Student Prompt</strong></th>
<th><strong>Defining Arguments in a Function with a Return Value</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Function with Arguments" /></td>
<td><img src="image2" alt="Function with Arguments" /></td>
</tr>
</tbody>
</table>

Now that you have learned how to assign values to the arguments in a function you can now have the function return a value to the user. Let’s try this to calculate the area of a rectangle.

<table>
<thead>
<tr>
<th><strong>Pre-Placed Blocks</strong></th>
<th><strong>Problem Statement</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Pre-Placed Blocks" /></td>
<td><img src="image4" alt="Problem Statement" /></td>
</tr>
</tbody>
</table>

Use the function block to a rectangle with an area of 36 square units.

You will need to fill in the variables needed to calculate the area of the rectangle – these blocks are under the “variable” tab.

Then you will need to assign values to the dimensions of your rectangle so the area equals 36 square units.
Finally, you need to enter in a complete sentence that tells the user the area of their rectangle.

<table>
<thead>
<tr>
<th>Wrong Prompt</th>
<th>You did not draw the rectangle with an area of 36 square units.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hint</td>
<td>To calculate the area of a rectangle you need to multiply the height and width. To find the dimensions of a rectangle with area 36 units squared think about numbers that have a product of 36.</td>
</tr>
</tbody>
</table>

**Possible Solution in C**

```c
#include <linkbot.h>

double variable;
CLinkbotI robot;
double radius = 1.75;
double trackwidth = 3.69;
int count;

double rectangleArea(double height, double width) {
    double area;
count = 0;
while(count < 2) {
    robot.driveDistance(height, radius);
    robot.turnRight(90, radius, trackwidth);
    robot.driveDistance(width, radius);
    robot.turnRight(90, radius, trackwidth);
count = count + 1;
}
area = height * width;
return area;
}

variable = rectangleArea(12, 3);
printf("A rectangle with a height of 12 and width of 3 has a total area of \%g units squared.\n", variable);
```
Activity #22 Functions: Calculating the Area of a Rectangle Using a Function with a Return Value

Problem Statement:
Use the function block to a rectangle with an area of 36 square units.
You will need to fill in the variables needed to calculate the area of the rectangle – these blocks are under the “variable” tab.
Then you will need to assign values to the dimensions of your rectangle so the area equals 36 square units.
Finally, you need to enter in a complete sentence that tells the user the area of their rectangle.

Location of Solution for “Load Blocks”
C-STEM Studio -> Teaching Resources -> TeachCh->RoboBlocklySolution-> CodingLevel2->c22.xml

C-STEM text alignment: Robot Programming with Linkbot for the Absolute Beginner, 5th edition
a) Section 5.4 Move a Distance for a Two-Wheel Robot. (driveDistance block)
b) Section 5.6 Turn Left and Turn Right (turn block)
c) Appendix A.2 Use a while-loop for Repeated Motions (repeat block)

Extension: Connect Linkbots from a Computer – have students move a hardwired robot by generating the Ch code using “Save Ch” on RoboBlockly.

C-STEM text alignment: Learning Computer Programming with Ch for the Absolute Beginner
b) Section 6.1 The while Loop (repeat block)