Code.org Puzzle Solving Recipe

(Based on Polya's Four Step Problem Solving Process)

These tips will help you get unstuck when solving Code.org puzzles!

Step 1: Understand the Puzzle

- What does the puzzle want you to do?
- Can you talk about the problem in your own words?
- Were you given any code to start?
  - What does it do?
  - Why do you think it's there?
- What is the goal of the puzzle?
- Have you solved any other puzzles that are like this one?

Step 2: Create a Plan *(Pick one or more)*

- Write an algorithm.
- Guess and check as you go.
- Draw a picture of what you want to do.
- Try working backward.
- Solve one small piece at a time.
- Compare to a puzzle that you've already solved.

Step 3: Perform and Perfect the Plan

- Did you solve the puzzle?
- If not, hunt for one error at a time.
- Retest your plan after every change.
- If you start to get frustrated, take a deep breath, or leave your screen for a minute. When you come back, you may see what was causing the trouble!
- Ask questions. Maybe one of your friends can help you figure out where your plan goes awry.

Step 4: Check Your Work

- Does your answer solve the puzzle?
- Did you hit all of the goals of this puzzle?
- Now that you have one way to solve the puzzle, is there an easier way to do it?
- If you change this solution a little, will it work for any other puzzles?
- Could you explain your solution to help

H Code.org Debugging Recipe

These debugging tips will help you keep moving when you get stuck!

Work to Avoid Mistakes

- Read the directions.
- What is the goal of the puzzle?
- Take it slow and go one step at a time.
- Can you talk about the problem in your own words?
- Were you given any code to start?
  - What does it do?
  - Why do you think it's there?

Debugging

- Look for problems each step of the way.
- Describe what was supposed to happen.
- Describe what is going wrong.
- Does the difference between what was supposed to happen and what did happen give you any clues?

- Fix one thing at a time, then describe how the result changed.
- Try leaving "breadcrumbs" in your program. You can put clues inside your code (like having your program "say" something) to let you know when each chunk runs.
- Try doing each task as its own chunk, then put all of the pieces together at the end so it is easier to see what each thing does.
- Try at least three ways of fixing problems before you ask for help.
- Talk to a friend. Maybe one of your classmates can help you figure out where your plan goes awry.