Algorithms & Programming: Program Development (3) Grade: 1

Standard 1.AP.PD.03
Identify and correct errors (**debug**) in **programs** which include sequencing and repetition to accomplish a task, through a variety of techniques and strategies that could include an **unplugged** activity (e.g., changing order or sequence, following steps, trial and error).

Essential Skills
Analyze a simple **algorithm** to find **bugs**.
Suggest solutions to the bugs in the algorithm using a variety of strategies.

Essential Questions
How do you identify an error in an **algorithm**?
What strategies can you use to correct an error in an algorithm?

Explanation
Students will try a variety of strategies to analyze, correct, and improve (**debug**) their **algorithms**, such as changing the sequence of steps, following the algorithm in a step-by-step manner, and trial and error. The algorithms should be a set of ordered steps that include at least one repeated sequence. Students should be able to describe what they are doing and why.

Think of this as similar to….
When you play a musical instrument, you have to keep practicing and correcting what does not sound right.

Implementation Examples—What would this look like in the classroom?

Title	Description	Link	Content Connection & Notes
The Very Hungry Bee-Bot	**Grade K**--Class uses a paper version of the robot to find a path from the starting point to the target. Following the algorithm the class created, students program a robot to reach a specific target. Students determine if the robot reached the target as expected. If it does not, students will identify that there was an error and discuss how it could be corrected. **Grade 1**--Given different algorithms to follow, students program their robot and report where the robot ends up; Students should compare where their robot ended up with where the robot of another group that was given the same algorithm ended up and determine if there were bugs in their robot's program. They should brainstorm possible causes for the differences and suggest ways to determine how to correct errors.	The Very Hungry Bee-Bot	This lesson also aligns with **CS** AP.A.01 and K.AP.V.01. and can be used with any robot, despite the title. The lesson assumes students have a basic competence with the robots. An anchor chart with the functionality of the buttons created in a previous lesson is referenced in this lesson.
Kidbot: Little Red Riding Hood	**Grade K--**Using a grid with the symbols for Little Red Riding Hood and a house, students create algorithms to move Little Red Riding Hood to the house. When the students follow the algorithm, they will determine if the algorithm worked as expected and Little Red Riding Hood arrived home safely. **Grade 1--**For the algorithms that don't work as expected students analyze it step by step to find out why Little Red Riding Hood did not get home. They can fix the bug by trying a number of strategies, such as guess and check. **Grade 2--**Students identify where the algorithm is not working as desired and make the changes they feel are needed. They then test the new algorithm and make corrections as necessary. (a different story can be used as appropriate and relevant to what students are studying)	Kidbot	This lesson also aligns with **ELA**.K.SL.6

Title	Description	Link	Content Connection & Notes
Dancing Alone	Grade K--Students are given a computer program in Scratch Jr. that programs the Scratch Cat to do a dance using motion blocks. The program has a ""bug"" and does not work as students are told that it will. Students identify if and where the cat does not do what they expected it to do. Grade 1--Students should identify the areas of code that did not work as expected and try to fix them. Grade 2: Students will create a computer program to make the Scratch Cat dance and identify anything that is not working as planned. They should determine what in the code is not working correctly and revise the code until it works as planned.	Dancing Alone	This lesson also aligns with **CS** AP.V.01, AP.C.01, AP.PD.01, and AP.PD.04 and is similar to Getting Loopy

Standard: AP.PD.03 Grade: 1

These annotations are a collaboration between Maryland Center for Computing Education and the Maryland State Department of Education.