6.AP.3 Algorithms and Programming

> *The student will use the iterative design process to create, test, and debug programs using a block-based or text-based programming language. (a) Create and test programs that use multiple conditional control structures. (b) Incorporate existing code, media, or libraries into original programs. (c) Trace and predict outcomes of programs. (d) Analyze and describe program results to assess validity of outcomes. (e) Analyze the outcomes of programs to identify logic and syntax errors. (f) Incorporate feedback from others to refine programs. (g) Revise and improve programs to resolve errors and produce desired outcomes.*

Understanding the Standard

This standard outlines the basic coding skills students should use as they address other standards in this strand. As students write code, they will make mistakes or write instructions that lead to unexpected outcomes. When this happens, they will need to use their debugging skills to correct the code and make the computer do what they intended it to do. It's pretty difficult for students to make any progress coding without addressing this standard, especially given that 6.AP.2 requires them to use conditionals.

Term	Definition
conditional	See 6.AP.2
library	A built-in set of commands to use in a program
code tracing	Reading a program line-by-line to predict what the computer will do when it executes the program
logic error	An error in human reasoning, leading to a program that doesn't work as intended
syntax error	An error in spelling, creating a program that the computer doesn't know how to execute
debugging	Identifying and addressing errors in code

Prerequisite Knowledge

The only element of this standard that requires prerequisite knowledge is 6.AP.3a. Before using conditional control structures, students will need to understand variables and relational expressions.

Integration Opportunities

Math 6.MG.4 d Develop a tool that allows users to input side lengths and angles of two polygons, uses multiple conditional control structures to assess congruence, and refine the program based on test results to ensure it accurately identifies congruent and noncongruent shapes.

History Skills USII e,f,i Create, test, and debug programs to compare and contrast, explain cause-effect relationships, and demonstrate understanding of content.

Science 6.2b Create, test, and debug programs using nested conditionals to identify planets in the solar system including sizes, order, distance from the sun, and other characteristics.

English 6.W.1a Use a story mapping tool to outline various decisions a character could make in a story. Trace the possible outcomes to assess their validity then revise and improve the outline to solve issues.

Summary of a Lesson

Any lesson that involves coding should also address this standard by incorporating activities that prompt students to read and predict the results of code. Have students read a program written in a familiar programming language and ask them to respond to questions like "what does this program cause the computer to do?" and "what mistakes are in this program?". You can even have students write their own code examples and trade programs, allowing students to practice their code tracing skills on a wide variety of examples. When students test their programs, ask them questions like "did the program do what you expected? Why or why not?" and "what errors did the computer find in the code?". Incorporating libraries, etc. is easy—almost all programming language involve using libraries, so refer to your chosen programming tool for details on this topic.