18

CS in Algebra | Lesson 18

Conditionals and Update Player

Lesson time: 30-60 Minutes

LESSON OVERVIEW

Using conditionals, students will write functions and programs that change their behavior based on logical evaluation of input values.

LESSON OBJECTIVES

Students will:

- Use Boolean operators to compare values.
- Apply Boolean logic, such as AND, OR, and NOT, to compose complex Boolean comparisons.
- Write conditional statements that evaluate differently based on their input values.

ANCHOR STANDARD

Common Core Math Standards

- **6.NS.8**: Solve real-world and mathematical problems by graphing points in all four quadrants of the coordinate plane. Include use of coordinates and absolute value to find distances between points with the same first coordinate or the same second coordinate.

Additional standards alignment can be found at the end of this lesson

TEACHING SUMMARY

Getting Started

1) Introduction

Activity: Conditionals

2) Online Puzzles

Extension Activities

3) Improving Luigi's Pizza

4) Update Player

TEACHING GUIDE

MATERIALS, RESOURCES, AND PREP

For the Student

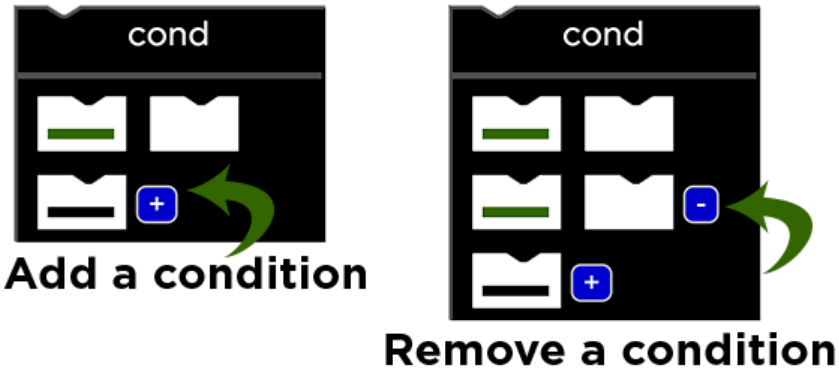
- Cost Design Recipe (in the student workbook)
- Update-player Design Recipe (in the student workbook)
- Key Code Reference (in the student workbook)

GETTING STARTED

1) Introduction

Remind students of the game they played in the last stage. What were some of the tricky elements of constructing a good conditional statement?

- Order matters (the first condition in the list to return true wins).
- Write clear and explicit conditions.
- Use the else clause as a catch-all for conditions that you don't expect or can't write explicit conditions for.
- All conditionals must have at least one condition and an else statement, you can add or remove further conditions using the blue buttons.



At the end of this stage, students will return to their Big Game to complete the update-player function. This function contains a conditional that will check which key was pressed (using key codes), and move the player up or down accordingly. We've provided a key code reference for students in case they wish to use keys other than the default up (38) and down (40) arrows.

LESSON TIP

Be sure to check students' Contracts and Examples during this exercise, especially when it's time for them to circle and label what changes between examples. This is the crucial step in the Design Recipe where they should discover the need for `cond`.

ACTIVITY: CONDITIONALS

2) Online Puzzles

Head to CS in Algebra stage 18 in Code Studio to get started programming.

EXTENSION ACTIVITIES

3) Improving Luigi's Pizza

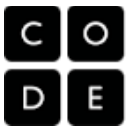
The final puzzle in the Luigi's Pizza sequence is a Free Play puzzle that allows for students to extend the program in a number of different ways. While some of the potential extensions seem simple, they can be deceptively challenging to get working. Allow students to explore extensions individually, or choose one to work through as a whole class.

- **Coupon Code**: Write a function `coupon` that takes in a topping and a coupon code and returns the price of a pizza with that topping, with %40 off is the code is correct.
- **Multiple Toppings**: Write a function `two-toppings` that takes in two toppings and returns the price of a pizza with those toppings.
- **Picture Menu**: Write a function `pizza-pic` that takes in a topping and returns a simple image representing a pizza with that topping.

3) Update Player

The `update-player` function is one of the most extensible in the Big Game. Here's a brief list of potential challenge extensions to give students:

- **Warping**: instead of having the player's y-coordinate change by adding or subtracting, replace it with a Number to have the player suddenly appear at that location. (For example, hitting the "c" key causes the player to warp back to the center of the screen, at y=200.)
- **Boundary-detection**: Keep the player on screen by changing the condition for moving up so that the player only moves up if the up ke was pressed AND player-y is below the top border. Likewise, change the condition for down to also check that player-y is above the bottom.
- **Wrapping**: Add a condition (before any of the keys) that checks to see if the player's y-coordinate is above the screen. If it is, have the player warp to the bottom. Add another condition so that the player warps back up to the top of the screen if it moves below the bottom.
- **Dissapear/Reappear**: Have the player hide when the "h" key is pressed, only to re-appear when it is pressed again!



Derived from