**Lesson 4** Else and Else if

**How can we write multiple sets of possible instructions in a program?**

| **Overview** | |
| --- | --- |
| This lesson builds on students’ knowledge of conditional statements by introducing “else” and “else if”. Students will use work in groups to . | |
| **Lesson Objectives** | |
| Students will be able to   * Write an “if...else” statement. * Use “else” statements to draw shapes and change attributes of shapes. * Use “else if” statements to draw a range of different shapes and attributes by creating thresholds for mouseX and mouseY. | |
| **Suggested Duration** | |
| Two periods (90 minutes) | |
| **Blueprint Foundations Student Outcomes (**https://blueprint.cs4all.nyc/outcomes/) | |
| Algorithms  Analyze | **Compare and contrast** instructions for completing a task with algorithms for completing types of tasks. |
| Algorithms  Prototype | **Demonstrate** the benefit of using an event, conditional or loop in my prototype. |
| Algorithms  Communicate | **Compare and contrast** how conditionals or loops were used in classmates’ prototypes. |
| Programming  Analyze | **Describe tools** and processes needed to collaborate on programming projects. |
| Programming  Communicate | **Present** the challenges, and benefits of using tools to program collaboratively. |
| **Vocabulary** | |
| * Conditional Statement: Specifies a block of code to be executed when a certain condition is met. The “else if” statement specifies a new condition to test if the previous condition is false. For “else” statements, the code is only executed when all previous conditions are false. | |
| **Planning Notes** | |
| * Assign groups of 3-4 for the traffic light activity (if this is not feasible, students may work in pairs). Provide the following materials:   + Chart paper (divided into four sections for pseudocode: see below)   + Markers (especially red, yellow, and green) | |
| **Resources** | |
| * Video tutorial: [Else, and else if, AND and OR](https://www.youtube.com/watch?v=r2S7j54I68c) (ignore AND and OR section) | |
| **Assessments** | |
| * Circulate during **Independent Practice**. Check for the ability to:   + Use “else” statements to control a p5 sketch   + Apply “else” statements in a new context using mouseY. * Assess written responses in the **Discussion**. Check for the ability to:   + Identify which thresholds are being tested   + Identify how the sketch will change when a given condition is true or false * Assess the **Wrap Up**. Check for the ability to:   + Explain the difference between “if” and “else if” | |

| **Do Now: Real Life Conditionals** |
| --- |
| In their notebooks, have students write conditional statements based on their lives. They should follow the structure below:   * *If \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, then \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. Otherwise \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.* * Ex. 1: *If* it is sunny outside, *then* I’ll go to the beach. *Otherwise* I’ll play video games. * Ex. 2: *If* it is a weeknight, *then* I’ll set my alarm to 6:30 am. *Otherwise* I’ll set my alarm to 10 am.   Call on students to share their conditionals with the class. |
| **Discussion: Else and Else if Statements** |
| * “Else” Statements:   + Explain to students that they will be able to use the concept of “otherwise” in their code to create more possible outcomes for their conditional statements. In code, this is done by adding an “else” statement onto an “if” statement.      * Open up a new sketch and type out the following code. Before you run it, ask students to predict what will happen when mouseX is less than 300, and greater than 300:      * Students should know from their work with if statements that if the condition is ***true***, the background will be black. Explain to students that if the condition is ***false***, (in this case, if mouseX is less than 300) the program will run the code block inside the else statement, making the background white. * Share the following starter code with students ([link here](https://editor.p5js.org/mparker/sketches/Y30bmfNgX)):      * Run the code, and ask why it is that no shapes have appeared on the screen. Students should be able to explain that an ellipse will only appear when the condition is true, and the mouse is on the right side of the canvas. * Check for Understanding:   + Tell your students to add an “else” statement to this conditional which draws a square in the same position as the ellipse when the condition is ***false***. Have students duplicate the sketch and work with a partner to construct the else statement. Circulate, and choose a student to read their statement out loud while you type it out. * Independent Practice   + Have students expand on the previous sketch by writing a new “if...else” statement based on the position of mouseY. [Here is an example](https://editor.p5js.org/mparker/sketches/y4PjyQ_Mj). Students should follow the prompts below:  1. Choose a condition that tests the position of mouseY 2. Write an if statement that draws an oval when the condition is true 3. Write an else statement that draws a rectangle when the condition is false  * Turn and Talk   + After students have written their own “if...else” statements, ask them if there is some mouse position where there will be more or less than two shapes on the screen at a time. If yes, where? If no, why not? Have them discuss with a partner.   + Open up the discussion to the whole class. Students should begin to recognize the logic behind the “if...else” statement. At any given time, a condition **must** be either true *or* false. This means that one of the two code blocks is *always* running, but never at the same time. * “Else if” Statements:   + Reiterate to students that when we use an “if” statement by itself, we create **one set of instructions** for the condition. When we add an “else” statement, we create **two sets of instructions**--a code block that runs when the condition is true, and a code block that runs when the condition is false.   + Now explain that there may be situations in which we want to have multiple sets of instructions for a whole **range of conditions**. You can connect this back to the “real life” conditionals from the Do Now by saying the following: “Another real life example of an “if...else” statement might be to tell someone that *if* it’s cold they should wear a coat, *otherwise* wear a t-shirt. But if I wanted to give someone more than two instructions I could say,   If it’s freezing, wear a coat,  Or else if it’s NOT freezing but it’s cold, wear a jacket,  Or else if it’s NOT cold but it’s a little chilly, wear a sweater.  Otherwise, wear a t-shirt.  To do this in code, we can add an “**else if**” statement to our conditional *between* if and else.”   * Project the pseudocode below so students can see the anatomy of an “else if” statement        * Explain to students that the program will test each condition in order, top to bottom. As soon as one condition evaluates to **true**, the instructions inside those brackets run and the program **skips** any “else if” or “else” statements that come after. This means that even if a person writes 100 sets of instructions inside 100 “else if” statements, each set of instructions can only happen **one at a time**. * Note: An “else if” statement does not not need to end in an “else” in order to work. |
| **Teacher Demo: Code Along** |
| * Tell students that you are going to use a conditional to recreate the following p5 sketch:      * Before building out the example, model how you would use pseudocode to create a plan for your code (students will need to use pseudocode in the next activity). It may be helpful to include a diagram of the circle with lines that show each threshold like the one below:      1. Create a color variable at the top of the sketch 2. Draw a circle 3. if mouseX is less than the left edge of the circle, make the color variable gray 4. if mouseX is past the left edge but less than the center of the circle, make the color variable pink 5. if mouseX is past the center but less than the right edge of the circle, make the color variable blue 6. else make the color variable purple  * Now build out [the example](https://editor.p5js.org/mparker/sketches/0lInPRezC) below based on your pseudocode as your students code along. Point out that:   + Conditions go inside parentheses, and each open parenthesis needs a “partner” to close it.   + Instructions go inside curly brackets and each open brace needs a “buddy” to close it.   + You can see if a bracket has a matching buddy by clicking next to it--pairs will have gray boxes drawn around them (see image below).   + Only “if” and “else if” statements take specific conditions. “else” **does not** take a condition inside parentheses because that code runs in all other scenarios, not given a particular scenario.          * Before running the program, instruct students to tent their laptops. Without moving the mouse around the canvas, give students a moment to look at the code. Call out different values for mouseX (e.g., 400, 150, 5, 300), ask students to predict what the output will look like for each value, and move the mouse to test each prediction. * Lastly, add comments to the code to further clarify the logic ([example here](https://editor.p5js.org/mparker/sketches/I5b5hKDgE)). * You may use the following language to further explain this sketch: “When I move the mouse so that the x position is greater than 100 pixels, the ellipse is pink. To find out why, let’s look at the full conditional. Our first condition tests if mouseX is less than 100. That’s false, so we skip to the next condition. Is the second condition true or false? Call it out. It’s **true**, so the fill changes to pink. Is the *third* condition true or false? It’s *also* true. But remember, only the *first* true condition will be seen by the code. Everything that comes after is **ignored**.” |
| **Student Activity: Traffic Light** |
| Tell students that they will work in groups to create a p5 sketch of a traffic light. Instructions are below. Here is an example [solution](https://editor.p5js.org/mparker/sketches/W9KzClrbs) for teacher reference.     1. Use conditionals to change colors based on **mouseY** 2. Use three **variables** for each color 3. Only one light can be on at one time   Group Pseudocode:   * For groups of 3 or 4, distribute chart paper with the sections seen below. For groups of two, students may write their pseudocode in theirdesign journals. * Instruct students to base their pseudocode on the steps you modeled earlier. Students should not get bogged down deciding on every value for the size and location of shapes. * When it’s time for students to combine their individual pseudocode into one group plan, they may include a diagram with some numerical estimates (like thresholds for mouseY), but should only write down the main steps and avoid JavaScript syntax.      | **Step 1** | Individually (without group discussion) write pseudocode in your area with each step you think you need to make the traffic light work. | | --- | --- | | **Step 2** | Take turns presenting each person’s pseudocode and giving feedback. | | **Step 3** | Group makes a decision about the most effective pseudocode to solve the problem and writes it in the center. |   Pair Programming the Traffic Light:   * Once a group has shown you their plan, they may begin coding in p5. For groups of 4, split them up into two pairs and assign initial driver-navigator roles. For groups of 3, there may be one driver with two navigators. Make sure that students switch roles every 5 to 7 minutes. * Remind students to save frequently and comment their code. |
| **Wrap Up** |
| Share Out:   * [**Design Journal**] First have students reflect on the following:   + What was the most challenging part of making this?   + What strategies did you use to get unstuck?   + How might you change or add onto this project in the future? * Invite a few groups to come up and share their process:   + What did you discuss when deciding on the group pseudocode?   + Describe your project and how it works.   + Share your responses to the reflection questions. * Make sure students share their code with each other and with you before they leave. |
| **Extension: Debug “Else if”** |
| * First show students [this sketch](https://editor.p5js.org/cs4all/present/SJ2Oz4aVZ) withoutshowing them the code.     Now share the [starter code](https://editor.p5js.org/mparker/sketches/1OqaEB9uE) that students need to debug:   * Give students time to work through it on their own. For students who are stuck, encourage them to move the mouse to a point on the screen and go through the “if...else” statement line by line, asking themselves if the condition is true or false and what instructions get run. Give students further support by asking them to compare the “if...else” statement in this exercise to the previous example with the changing fill and try to notice any key differences in the order that conditions are tested. * Circulate the room, and pick a student who can explain their process for figuring out the “bug”--that the first “if” statement is **true** for any value of mouseX greater than 100, so all of the following “else if” statements are **skipped**. * Pick another student who can explain how they fixed this bug in their code. Students may compare this exercise to the previous example to explain how they realized they needed to test the highest value first (see example below).   + Note: Be sure to clarify that this **doesn’t** mean to always start by testing the highest value. For other examples, students may need to first test the lowest value. The important thing to stress to students is that when using “else if” statements, **order matters**. |