Lesson 5: One-Way Selection Statements

45 minutes

Overview

How can I specify alternate outcomes based on the current state of the `Painter`?

Students incorporate `if` statements into their programs to make decisions based on the current state of a `Painter` object. Students learn about methods that return `boolean` values and how to execute a specific block of code based on the method call result. As the complexity of their programs increases, students identify debugging strategies to find and correct syntax and logic errors.

Standards

Full Course Alignment

CSA Conceptual Framework

- **CON-2** - Programmers incorporate iteration and selection into code as a way of providing instructions for the computer to process each of the many possible input values
- **MOD-1** - Some objects or concepts are so frequently represented that programmers can draw upon existing code that has already been tested, enabling them to write solutions more quickly and with a greater degree of confidence
- **MOD-2** - Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attirbutes and/or behaviors of the object or concept
- **VAR-1** - To find specific solutions to generalizable problems, programmers include variables in their code so that the same algorithm runs using different input values

Agenda

Warm Up (5 minutes)
CS in the Arts

Activity (30 minutes)
Selection Statements

Objectives

Students will be able to:

- Identify strategies for debugging syntax and logic errors
- Use methods that return information about the state of an object
- Write `if` statements that only execute if a specified condition is `true`

Preparation

- Create a Debugging Wall by either designating a physical space in your classroom or making a digital version using a presentation program or an interactive whiteboard as indicated on Setting Up the Debugging Wall resource
- If using a physical Debugging Wall, gather two sticky notes for each student
- Check the **Teacher's Lounge** for verified teachers on the CSA Forum to find additional strategies or resources shared by fellow teachers

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the students

- **Conditional Statements** - Video
- **U1L5 Extra Practice** - Handout

Vocabulary

- `boolean` - a primitive data type that can be either `true` or `false`
- `if` **statement** - a conditional statement that only executes when the condition is `true`
- **condition** - determines whether or not to execute a block of code
- **conditional statement** - a statement that only executes when a condition is `true`
- **logic error** - an error that occurs when the code runs but does not do what was expected
- **return** - to exit a method and go back to the point in the program that called it with the requested value or information

Teaching Guide

Warm Up (5 minutes)

CS in the Arts

Remarks

> The impacts of computer science can be seen in many different industries and interests, even as a form of creative expression. For example, art has expanded beyond paintings or sculptures to include digital media, animations, and computer-generated art.

Do This: Share examples of computer science in the arts.

Do This: Direct students to write down ideas and thoughts in response to the prompt on a sticky note or scrap piece of paper.

Teaching Tip

Students participate in a Give One, Get One activity during the wrap up to share their ideas with their peers.

Activity (30 minutes)

Selection Statements (10 minutes)

Remarks

> In the previous lesson, we learned how to use some of the methods in the `Painter` class to navigate

> and paint The Neighborhood. When we gave a `Painter` object instructions to move and paint, we had to tell Java each step to take. We could use conditional statements, or `if` statements, to check for things like if a `Painter` object is on a paint bucket or if a `Painter` object can move in a certain direction and execute specific blocks of code based on whether those things exist.

Do This: Review the lesson objectives.

Display: Show the video – *Conditional Statements*.

Do This: Direct students to Level 1 on Code Studio to predict the program's outcome, then run the program to compare their predictions to the actual outcome.

1 Predict: Conditionals and More Methods

Discuss: Click through the animated slide to display the prompts.

- *What do you notice about the code in this program?*
- *What do you wonder about the code in this program?*

Discussion Goal: Students identify the methods used in the program and share ideas about what each method does. Students also notice these methods are called in the `if` statements and that the block of code only executes if the condition is `true`. Students may wonder about checking these conditions repeatedly or about other conditions they might check about the state of the `Painter` object.

Remarks

> This program uses methods that the `Painter` class has that give us information about the current state of a `Painter` object. These methods tell us whether or not the `Painter` object is on paint, is on a paint bucket, or can move in a specific direction, which we can use with `if` statements to execute a block of code based on the result of these method calls.

Do This: Define `boolean` and *return*.

Discuss: Use the Hold That Thought strategy to discuss the prompt.

- *Look at these methods in the UML diagram for the `Painter` class. What keyword is in front of the method name? What do you think this means?*

Discussion Goal: Students identify the `boolean` keyword before the method name. Students share ideas about what `boolean` means and connect to the program they saw. Students suggest that the methods return a `boolean` value.

Teaching Tip

Prompt students with examples of where they might have heard the word "boolean" before, such as from math class or from Computer Science Principles (CSP).

Using if Statements (10 minutes)

Remarks

> Let's practice using these new methods with `if` statements to move around and paint The Neighborhood!

Do This: Direct students to Level 2 on Code Studio to complete Levels 2 and 3. Students debug a program on Level 2, then continue to Level 3 to complete a choice level.

2-3 **Conditional Moving and Painting**

2 3

Debugging Strategies (10 minutes)

Remarks

> Sometimes, we may encounter syntax errors when we don't write an `if` statement correctly. When this happens, we can use the console to help identify the cause of these errors.

Do This: Click through the animated slide to explain a possible error and potential console error message that can occur if an `if` statement is not written correctly.

Remarks

> Syntax errors aren't the only errors we might run into with `if` statements! Sometimes we may encounter situations where the syntax is correct, but the program does not do what we expected.

Do This: Define *logic error*.

Do This: Click through the animated slide to explain the example of a logic error and strategies to identify the causes.

Teaching Tip

Have students check the condition and determine whether or not the code will execute. Ask students why the `if` statement does not execute and how they could fix the program to have the `Painter` move south instead.

Remarks

> Having bugs in your program is a normal part of being a software engineer – even professional software engineers discover bugs in their code every day. Even though bugs can be frustrating, they can always be found and fixed. Throughout this course, we will have a classroom Debugging Wall where you can share the strategies you used to find and fix your errors.

Do This: Introduce the Debugging Wall.

Distribute: Give each student one to two sticky notes.

Do This: Have students think about bugs they have encountered so far and write down a strategy they used to find and fix the error.

Teaching Tip

If students struggle to recall bugs they have encountered, ask them about errors they might have encountered in any of the programming activities from previous lessons. You can also share an example of a bug you encountered while learning Java and how you found and fixed the error.

Do This: Direct students to add their sticky notes to the Debugging Wall under the appropriate category and take a moment to read over other strategies.

Teaching Tip

Ask additional guiding questions to help students identify strategies, such as:

- **Describe the problem:** What do you expect it to do? What is actually happening? Does this always happen?
- **Use your tools:** Are there any warnings or errors when your code compiles? What lines does the compiler highlight as problematic?
- **Retrace your steps:** What did you change most recently? What code is related to the problem you've identified?
- **Enlist a partner:** Explain your code to someone else. Sometimes a second pair of eyes is all you need!
- **Try solutions:** Make small changes to your code - what happens?
- **Document as you go:** As you debug, make notes in your journal and add new strategies to the Debugging Wall.

Wrap Up (10 minutes)

Revisiting CS in the Arts

Remarks

> At the beginning of this lesson, we saw how computer science is used in the arts. Let's revisit the ideas and thoughts you had and share these with your peers. It's time to...

Do This: Play the music clip to cue the Give One, Get One activity and direct students to participate in a Give One, Get One.

Discuss: Click through the animated slide to display the prompts.

- *What did you learn from each other?*
- *How do these ideas and thoughts contribute to your identity as a software engineer?*

Discussion Goal: Students share the ideas they gathered and identify patterns or common thoughts. Students recognize how computer science can be used to express themselves and as a creative outlet and make personal connections to their identity as software engineers.

Do This: Review the concepts covered in this lesson.

Display: Key Vocabulary

Assessment: Check for Understanding

Check For Understanding Question(s) and solutions can be found in each lesson on Code Studio. You can use these questions as an exit ticket.

4 **Check for Understanding**

AP Classroom Topic Questions

To assign questions from the AP Classroom Question Bank that align with this lesson, create a custom quiz in AP Classroom by searching the Question Bank for the Essential Knowledge statements listed at the top of this lesson plan. You can find instructions and video demonstrations to do this on **AP Central**.

The following Topic Questions in AP Classroom can be assigned as a formative assessment for this lesson:

- Topic Questions 3.2

Note: *Some Learning Objectives and Essential Knowledge statements in the suggested Topic Questions are covered in later units.*