Lesson 7: Conditionals Practice

Overview

In this lesson students spend most of their time practicing using the skills and processes they have learned about conditionals. At the conclusion of the lesson students discuss the main things they realized and still have questions about at the conclusion of this lesson.

Purpose

This lesson is students primary opportunity to get hands on with conditionals in code prior to the Make activity in the following lesson. Give students as much class time as you can to work through these. For this lesson it's recommended that you place students in pairs as a support and to encourage discussion about the challenges or concepts they're seeing. In the following lesson students are encouraged to work independently.

Standards

Full Course Alignment

CSP Conceptual Framework

- **AAP-2** - The way statements are sequenced and combined in a program determines the computed result. Programs incorporate iteration and selection constructs to represent repetition and make decisions to handle varied input values.
- **AAP-3** - Programmers break down problems into smaller and more manageable pieces. By creating procedures and leveraging parameters, programmers generalize processes that can be reused. Procedures allow programmers to draw upon existing code that has already been tested, allowing programmers to write programs more quickly and with more confidence.
- **CRD-2** - Developers create and innovate using an iterative design process that is user-focused, that incorporates implementation/feedback cycles, and that leaves ample room for experimentation and risk-taking.

CSTA K-12 Computer Science Standards (2017)

- **AP** - Algorithms & Programming

Agenda

Objectives

Students will be able to:

- Debug programs that use boolean expressions and conditional statements
- Write programs that use boolean expressions and conditional statements with the support of sample code.

Preparation

- Review the programming challenges students will be completing
- Review the Debugging Guide for ideas on how to support your students during the lesson

Links

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

For the teachers

- **CSP Debugging Guide**
- **CSP Unit 4 - Variables, Conditionals, and Functions** - Slides
- **Guide to Practice Lessons** - Video

Teaching Guide

Lesson Modifications

 **Attention, teachers!** If you are teaching virtually or in a socially-distanced classroom, please read the full lesson plan below, then click **here** to access the modifications.

Warm Up (5 minutes)

Quick Warm Up

Remarks

> Today we're going to have a chance to practice programming with a lot of the concepts and patterns we've explored over the last two lessons. I encourage you to go through these with a partner, but pay close attention to what each other is doing. In our next lesson you're going to have to use a lot of these on an independent project, and these activities are good practice for what you'll find there! Alright, let's get to it!

Teaching Tip

Move Quickly to the Activity: There's a lot in the main activity of today's lesson. You may optionally wish to do a quick vocabulary review or address any questions that came up in the last lesson. Otherwise, give students more time to get hands on with some code.

Activity (35 minutes)

Practice Time

Group: It is recommended that students work in pairs for this lesson and a number of the activities feature discussions. Consider using pair programming, having drivers and navigators switch every 3 minutes, not every level.

Do This: Direct students to Code Studio, Lesson 3 Level 2. Then briefly remind students about debugging skills that they will be using in today's activity.

Remarks

Today you're mostly going to practice what we've learned about programming with conditionals. As always you should be using the debugging process to help you as you work on issues. Today we're also going to be working on finding two types of errors

1. Syntax errors show up when you type code that breaks the rules of the programming language. You can check for errors and warnings
2. Logic errors show up when you type valid code but it works incorrectly. Today you're going to focus on testing your code to make sure you don't have logic errors.

Other errors you may encounter include:

- Run-time error - a mistake in the program that shows when running the program. These are defined by the programming language.
- Overflow error - an error that occurs when a computer tries to handle a number outside of the defined range of values.

Levels 1-3: These levels only use the `console.log()` command which prints commands in the debug console. Here are a few things to keep an eye out for

- Levels 1-2 ask students to write Boolean expressions using comparison operators. Students may need to quickly review the comparison operators `<`, `>`, `<=`, `>=`, `==`, `!=`
- Level 3 asks students to write Boolean expressions with logical operators `&&`, `||`, `!`

Teaching Tip

Providing Support: Circulate around the room through the lesson encouraging students to use the strategies introduced at the beginning of the lesson. Students have a number of supports at their fingertips, so a big part of your role is helping build their independence in using those resources.

1-3 **Boolean expressions**

1 2 3

Levels 4-8: These levels practice if-statements while working with a star color-changing app.

- Levels 4-5 involve setting up an if-statement that becomes an if-else statement.
- In Level 6 students follow a pattern to create a lengthy if-else-if statement.
- For Level 7, make sure students slow down the running of the code to understand what's happening. It's suggested that students use the slider to slow down the code.
- Level 8 demonstrates that Boolean expressions can be written as conditional statements, and vice versa

4-8 **If-Statements**

4 5 6 7 8

Levels 9-10: The levels return to the "Can I Adopt a Cat?" flowchart from the Conditionals Explore activity. Students will use the flowchart to work out the logic of the if-statements in a their program.

- A new block appears in these levels: `getNumber()`. This is different than `getText()`. `getNumber()` gets a number from a user input that can be used mathematically.

- Level 10 can be completed many different ways. There are different combinations of Boolean expressions using `&&` and `||`. Students should regularly test their apps to see if their Boolean expressions are working properly.

9-10 **Logical Operators**

9 10

- Level 11 Students do a quick practice with the MOD operator.

11 **MOD Practice**

Extension Opportunities:

- Level 3: Students can add more variables and create complex Boolean expressions. One challenge might be to assign a String to a variable and compare that string to another.
- Level 9: There are multiple solutions. If students build their if-statement using only `&&` encourage them to figure out how to build it using only `||`. They may need to switch the content of the if and else branches.
- Level 10: Create another input (i.e. How many cats do you already own?). Students use this information to craft more complex if-statements.

Wrap Up (5 minutes)

Discuss: *What aspects of working with conditionals do you feel like clicked today? What do you still feel like you have trouble with?*

Discussion Goal: Use this opportunity to address any lingering questions or misconceptions in the room. You can also use this as a source of discussion topics to kick off the following lesson. As you lead the discussion, call out the many resources students have access to help when they're getting stuck.

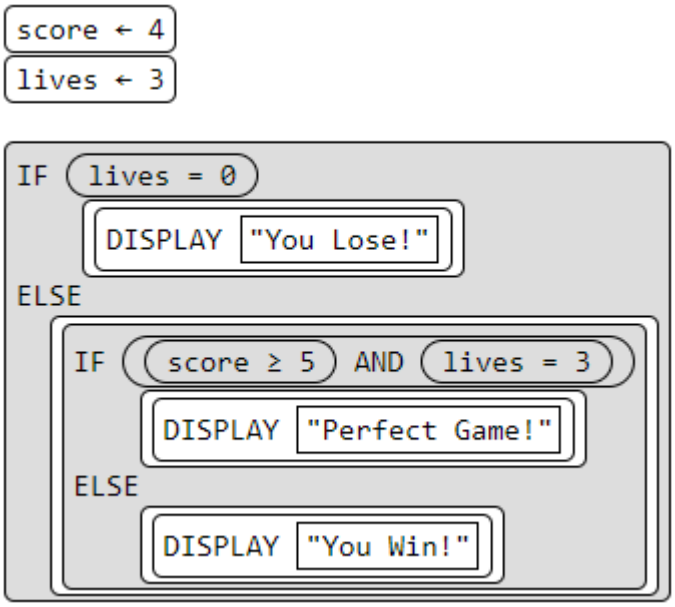
Remarks

> Conditionals can be a little bit tricky, but I saw a lot of good progress today in nailing down this concept. We may have a few lingering questions, but you also have a lot of resources available. Next time you'll have a chance to put all this together by programming an app that starts with "the blank screen"!

Assessment: Check For Understanding: AP Practice

Check For Understanding Question(s) and solutions can be found in each lesson on Code Studio. These questions can be used for an exit ticket.

Question: What will be displayed after this code segment is run?



Question: The program below asks a user to type in a number and then will output a message. What number will a user need to input for the message `"COLD"` to be displayed?

```
number <- INPUT()

IF (number >= 10)
{
  IF (number <= 20)
  {
    DISPLAY("MEDIUM")
  }
  ELSE
  {
    DISPLAY("HOT")
  }
}
ELSE
{
  DISPLAY("COLD")
}
```

12-13 Check For Understanding: AP Practice

12 13