Lesson 5: The Need for Programming Languages

Overview

In this lesson students explore the challenges of clearly communicating instructions. They build a small arrangement of blocks (LEGO® pieces or paper cutouts) and then create text instructions a classmate could follow to construct the same arrangement. Groups then trade instructions to see if they were clear enough to allow reconstruction of the original arrangement. The wrap-up discussion is used to highlight the inherent ambiguities of human language and call out the need for the creation of a programming language which leaves no room for interpretation.

Purpose

This lesson is students' first introduction to the concept of a programming language. It helps them understand why programming languages exist by giving them a first-hand experience with the problems that programming languages are designed to address. Natural language is usually too ambiguous for giving precise instructions that can be followed correctly 100% of the time. We need to create more structured and precise programming languages in order to accomplish this.

Standards

Full Course Alignment

CSTA K-12 Computer Science Standards (2017)

- **AP** - Algorithms & Programming

Agenda

Lesson Modifications

Warm Up (5 minutes)

Activity (30 minutes)

Wrap Up (10 minutes)

Assessment: Check For Understanding

Objectives

Students will be able to:

- Explain the qualities that differentiate natural languages and programming languages
- Justify the existence of programming languages to precisely communicate instructions

Preparation

- Prepare either a small set of LEGO blocks or paper cutouts for each pair of students

Links

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

For the teachers

- **CSP Unit 3 - Intro to App Design** - Slides

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)

Remarks

> Today we're going to look at what it takes to write "good" instructions. We all use instructions all the time, whether it's directions to get somewhere, instructions to fill out a form, or even instructions from teachers. To help us get started, let's brainstorm what "bad" instructions" look like.

Discuss: *Write down three different reasons you would call a set of instructions "bad". Be ready to share with a neighbor.*

Discussion Goal: Aim to hear a few different students share reasons that instructions are "bad". The point here is just to get students thinking and there's no specific answer you're driving towards. Some possible ideas, however, might include:

- Instructions are not clear on what to do
- Instructions use confusing words
- Instructions don't actually accomplish what they're supposed to

Remarks

> Alright, so we have an idea of what "bad" instructions look like. Let's see if we can use this brainstorm to help us write good instructions in the next activity!

Activity (30 minutes)

Group: Place students in groups of two, optionally forming a group of three.

Distribute: Give each group a small set of blocks (either LEGO or paper cutouts). Also have each group use a blank sheet of paper or a blank page in a journal.

Display: Use the lesson slides to guide this activity.

Step 1: Design: Give students a couple minutes to create their design. Keep this quick since the bigger focus should be on the instructions.

Step 2: Record: Either have students sketch their design or take a picture. Make sure the image is on a separate piece of paper or the back side of a sheet of paper.

Step 3: Write Instructions: This is the most important step and should be given about 5 minutes. Encourage students to be as clear as possible while only using words. Encourage them to think about their brainstorm of "bad" instructions this morning to see if it can inspire them to make "good" ones.

Step 4: Trade: Have students take apart their design, and then trade instructions with another group. Make sure they keep their recording of the design hidden.

Step 5: Build: Give students 3 minutes to follow the instructions from another group.

Step 6: Compare: Give students 2 minutes to compare what they built with the picture of the actual target. In their groups they should discuss what they think went wrong or anything they're surprised worked out.

Step 7: Repeat: Depending on how much time you have, encourage students to trade with one or two other groups. Make sure to save time for the wrap up.

Teaching Tip

Make it a Gallery Walk: Depending on your room set-up you may be able to have students run this activity in a more open-ended way. Students should leave their unconstructed blocks and instructions at their table with the target image face down. They can then make their way to a few different groups' instructions over the course of 10-15 minutes.

Mark What's Confusing: You may optionally have students leave suggestions on instructions when they use them explaining where they were confused or wanted more detail. This is a good way to practice giving and receiving feedback. It also forces groups to think more clearly about what is causing their confusion since they'll have to leave it in writing.

Wrap Up (10 minutes)

Discuss: *When you or your classmates made mistakes following instructions today what "went wrong"? Try to be as specific as possible.*

Discussion Goal: This wrap up includes two sets of prompts and should be run as back-to-back discussions. The first set helps students synthesize the challenges they encountered during the main activity. The second set prompts students to think about how they might design a new language that avoids these challenges.

When running the first discussion ask students to share specific experiences during the lesson and prompt them to think about what went wrong. They might say things like "if we'd just been more clear" or "we just need to include more detail". This might be true, but eventually you should point out that human language is often ambiguous. In other words, it's not their fault! It's hard to do a good job with bad tools!

When running the second discussion push students to think about what a language designed for giving instructions would look like. While specific answers aren't important, you should push them to build on their experiences in the main activity. For example, if they saw that words having two meanings was confusing in the main activity, they should suggest redesigning the language so each word has a single well-understood meaning.

Remarks

> Even trying to be as clear as possible we struggled to write clear instructions. There are things we could do to improve, but the core challenge here is that everyday human language is bad for giving clear instructions. Words can have two or three meanings! Instructions that seem clear turn out to be vague when we actually go follow them. If we want to give clear instructions, we need to fundamentally change the language we use. We need to create a new kind of language.

Discuss: *Imagine we were going to redesign human language to be really good for giving clear instructions. What types of changes would we need to make?*

Remarks

> Today we talked a lot about instructions because soon we're going start programming your apps. When you write a program you're just giving instructions to a computer for what it should do to run your app. As you'll see, the programming language we use to give these instructions sometimes looks like English, but then has a lot of weird (and sometimes confusing) differences. This is because it needs to be more precise and unambiguous than normal human language. We'll dive into this more next time we meet!

Assessment: Check For Understanding

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 is the difference between a programming language and natural (every-day) language?

1 **Check For Understanding**