Computational Thinking
Tangram Activity

Objectives

The goal of this activity is to explore how challenging it can be to break a problem down, and communicate the solution to that problem. This mirrors two major aspects of computational thinking: problem solving and communicating information (with a computer). This activity can be extended in several ways to focus on certain aspects of the problem solving and communication.

Materials

- Sets of tangrams, at least one per team. Two per team if you have enough.
- Sample tangram solutions.
- Divider or other mechanism for two team members to not be able to see each other's tangrams (or solution) during the activity.

Activity

Introduction

Computer science is a discipline where we use computers to solve problems. This first requires the computer scientist to practice solving problems, and secondly communicating that solution to the computer so it can do the work. This activity allows us to explore both aspects of computer science and you will also find that it allows us to explore ideas in a wide range of other disciplines.

Setup

Break the students up into groups of two or three. One student will be the *computer*, the other will be the *programmer*. If you have groups of three, the third person will be the *observer*. Each group gets one or two sets of tangrams and a divider. The *programmer* is given a sheet with one or more tangram solutions and sets up the divider such that the *computer* cannot see the solution, and the *programmer* cannot see the *computer's* workspace. The *computer* is given a set of tangrams.

The *programmer* has one minute to prepare for communicating with the *computer*. If multiple options for the pattern to communicate, the *programmer* uses this time to pick one.

Each team is given five minutes to communicate. The goal is for the *programmer* to successfully communicate with the *computer* such that the *computer* produces the correct pattern. Remember that the two team members cannot see what the other is doing. The observer, if present, will observe the communication and progress of the *computer* silently and consider what goes well and what does not.

After the time is up, or if the team completes the activity earlier, they are given two more minutes to discuss what aspects of the activity were easier or harder than others, how the *programmer* could have communicated better with the *computer*, and what would help the *computer* get to the goal more quickly.

If time allows, switch roles and do the activity again using a different pattern. Discuss the same issues and compare how differences in the problem impact how well the activity went.

Discussion

This activity can lead to many different threads of conversation, but today we are focusing on the computational thinking skills. To start the discussion we will ask some broad, open-ended questions, and then lead the conversation to the computational nature of the activity.

General Questions:

- What went well? What strategies seemed to work for your group?
- What did not go well? What was frustrating?
- Are there ways to simplify the activity to make it easier/faster/better suited to younger students?
- Are there ways to extend the activity to make it more challenging/longer/better suited to older students?

Computational Thinking Discussion:

- How did naming the roles *programmer* and *computer* impact how you thought about the activity?
- How was the programmer role similar to what a programmer does?
- How was the computer role similar to what a computer does?
- How did your choice of language impact the task? How does that relate to computational thinking?
- What problem solving strategies did you use to accomplish the goal?

Variations

There are many opportunities for variations for this activity. Varying the materials used (tangrams, legos, duplos, shapes, particular mix or number of blocks) and complexity of the task to complete (size, number of dimensions, number of blocks to complete). The amount of space and time would also need to be scaled as appropriate for those variations. Groups of students can be used for both roles as well.

Variations in language: communication frequency, vocabulary that is allowed, the amount of words used to communicate can all be restricted or not to increase or decrease the level of difficulty. To emphasize computational thinking, it would be good to consider the set of words used and what is important information (orientation, size) and what is not as important (color, texture). This is a nice way to highlight abstraction, an important concept in CT.

Similar style activities with the roles of programmer and computer can also be done with tasks in other disciplines to show how problem solving is used in that discipline, and how abstraction is used to communicate that to a computer.

Other Resources

If you liked this activity and are looking for more, here are just a few of our favorite resources and a link to where you can contact us for more information.

- Code.org - lots of great curricular materials, videos, and tools. https://code.org/
- CS Unplugged - computational thinking activities typically for K-8 students without a computer. https://csunplugged.org/en/
- Exploring Computer Science - a computational thinking curriculum for K-12. http://www.exploringcs.org/

Please check out some more resources we have compiled and presented at previous workshops on our github page: https://github.com/CS4HS-UWL/cs4hs-uwl.