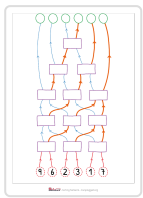


Investigating variations using the Sorting Network

- **Duration:** 30 minutes
- **Ages 5 to 7:** Lesson 2

Printables



Sorting Network

Use blank copy as template for creating Sorting Network.



Sorting Network Cards

Cards for students to use with the Sorting Network.

Classroom resources

- Chalk

Learning outcomes



Students will be able to:

- Demonstrate how your alphabet knowledge supports ordering words or books.
Literacy: Reading
- Explain how a Sorting Network parallel algorithm works.
Computational Thinking: Algorithmic Thinking
- Recognise that a sorting method can be applied to any values that can be ordered.
Computational Thinking: Generalising and Patterns

Preparatory knowledge

Students should have completed lesson 1 to introduce Sorting Networks.

Key questions

- In the Sorting Network, do you think that we can sort things other than numbers? What do you think we could sort? (Potential answers could include sorting things by colour, size, age, and height).

Lesson starter

Show the students the Sorting Network again (if the network needs redrawing then students often enjoy doing this, and drawing it accurately from the diagram is a useful exercise). Tell them that they will be trying it with some variations this time.

Mathematical links

Predicting outcomes: By understanding how the Sorting Network works students will be investigating different ways of using the Sorting Network and exploring how the lowest and highest number always ends up in the correct output position.

Variations

This part of the lesson explores changing the way the numbers are used.

Variation 1: Identical value



In this variation, students try the Sorting Network with a set of cards where some cards have an identical value, such as 1, 2, 3, 3, 4, 5. They will probably ask what to do when comparing the identical cards - ask them what they think, and they are likely to realise that it won't make any difference (if 3 and 3 meet, then it won't matter which one goes left and which goes right!). Ask them to predict what will happen at the end of the network (they may realise that the identical values will end up adjacent).

Now run the numbers through the network to check. Here's a brief reminder of the Sorting Network instructions; full details are in lesson 1.

1. Six students start in the input circles, each holding a card with one of the numbers on it.
2. They all step forward at the same time, and when they meet someone in a box, they compare their cards.
3. The person with the smaller card follows the line out to the left, and the larger card to the right (this is reversed in the second variation for this lesson).
4. This continues until all the students reach the output circles, at which point they should be in sorted order.

Variation 2: Larger to the left

This time, the person with the larger number goes to the left instead of the right and follows the line to the next square, while the person with the lower number goes to the right instead of the left and follows the line to the next square.

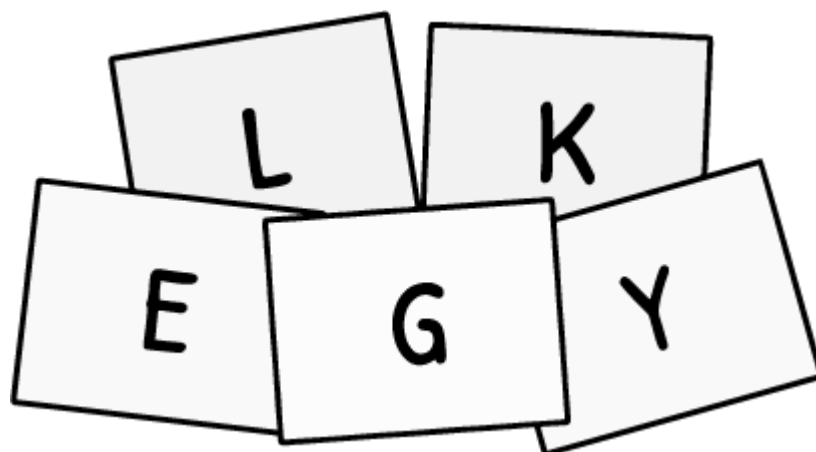
Ask the students to predict what will happen (they should be able to work out that the values will come out in reverse sorted order i.e. from largest to smallest instead of smallest to largest).

Have them try it out with some numbers to check it.

⊖ Teaching observations

By reversing the left/right decision, the final result will be in the reverse order to how it would have been in lesson 1.

Variation 3: Letters of the alphabet



Give the students cards with letters on them. Ask how we could compare these (students should observe that they could be in alphabetical order). Have them test this by sorting the cards.

Using the network backwards

This is an experiment that addresses a question that students may have asked: does the Sorting Network correctly sort the values if we start at the other end?

Have students try this with some simple values (such as the numbers 1 to 6). Chances are that it will work for many starting orders of the values. However, encourage them to keep trying until they find an initial order for which it doesn't work. This will require considerable reasoning to achieve.

If they struggle to find an example, you could give the one below, and then challenge them to find a different one that doesn't come out sorted.

⊖ Teaching observations

The Sorting Network is designed to work consistently one way, rather than working both ways. For example, the first image below shows an input that happens to come out sorted when going through the network backwards, while the second one doesn't. If it fails on just one input (the second one) then we can't rely on it, even though it sometimes works. In the other direction, it will always sort correctly.

The diagram illustrates a sorting network with two examples. Each example consists of six columns of nodes. The top nodes are dashed red circles, and the bottom nodes are solid green circles. The nodes are connected by red arrows, representing comparisons and swaps. The first example shows an input sequence of 1, 2, 3, 4, 5, 6 (dashed red circles) at the top and an output sequence of 6, 5, 4, 3, 2, 1 (solid green circles) at the bottom. The second example shows an input sequence of 1, 4, 2, 3, 5, 6 (dashed red circles) at the top and an output sequence of 5, 1, 2, 3, 6, 4 (solid green circles) at the bottom. The nodes are connected by red arrows, representing comparisons and swaps between nodes in adjacent columns.

Applying what we have just learnt

This kind of algorithm needs to run on special hardware to take advantage of doing multiple comparisons at the same time. It is only used for specialist applications at present, for example it is sometimes done on the graphics processor (GPU) of a computer, because these processors are good at doing parallel computation. Sorting Networks were invented long before powerful GPUs came along; this is an exciting thing about Computer Science - some of our discoveries are ahead of the hardware that is available, so we're ready to make use of the hardware when it does become commonly available! Note that this is *not* a conventional sorting algorithm, as the sorting that is done on a conventional system can make only one comparison at a time; conventional sorting algorithms are explored in another lesson.

Lesson reflection

What did you notice happen with each variation of using the Sorting Network?

Was it what you had expected?

Seeing the Computational Thinking connections


Throughout the lessons there are links to computational thinking. Below we've noted some general links that apply to this content.

Teaching computational thinking through CSUnplugged activities supports students to learn how to describe a problem, identify what are the important details they need to solve this problem, break it down into small logical steps so that they can then create a process which solves the problem, and then evaluate this process. These skills are transferable to any other curriculum area, but are particularly relevant to developing digital systems and solving problems using the capabilities of computers.

These Computational Thinking concepts are all connected to each other and support each other, but it's important to note that not all aspects of Computational Thinking happen in every unit or lesson. We've highlighted the important connections for you to observe your students in action. For more background information on what our definition of Computational Thinking is [see our notes about computational thinking](#).

Many of the connections are covered in the unit plan and lesson 1. Here are some additional connections:

⊖ Algorithmic thinking




When comparing words for alphabetical order, the algorithm involves comparing the two words letter by letter, and basing the decision of the first pair of letters that differ.

What to look for

Were students able to systematically compare words? Can they articulate the algorithm, particularly when comparing words with a long prefix in common (such as "computer" and "computing")?

⊖ Abstraction




Sorting Networks can work for any type of data that can be compared. This means that we do not need to know what the data is, we just need to know how to compare it and order it.

What to look for

Did students recognise that different types of data could be compared with the same Sorting Network and same process? Can they come up with new types of data to sort?

⊖ Generalising and patterns




In this lesson we moved from comparing numbers to the idea of comparing information in general. This meant we were able to compare other things like letters.

What to look for

Did students recognise that comparing other types of information, such as letters, would sort them into a relevant order? For example according to alphabetical order.

⊖ Evaluation



We evaluated if the Sorting Network would work backwards. If we find one example that fails, then that establishes that it can't be used in that way.

What to look for

Did students recognise that the one example of a Sorting Network not working backwards was enough to show that it isn't a valid Sorting Network?

⊖ Logic



As described in the unit plan, if the data being sorted have a transitive relation then the Sorting Network will be able to sort them, and each of the types of data we used in this lesson has this transitive relation.

What to look for

Are students able to recognise that each of the sets of items compared in this exercise have a transitive relation? Could they identify the most logical comparison to use (such as alphabetical order)?

Can they think of any types of data that don't have a transitive relation, and that we can't sort with the Sorting Network?

One answer could be putting food into order of tastiness - If you like pies more than spaghetti, and your friend likes lasagne more than spaghetti, that doesn't necessarily mean you like lasagne more than pies, and your friend might not like spaghetti more than pies!