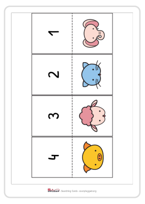
Divide and conquer

- **Duration:** 30 minutes
- **Ages 8 to 10:** Lesson 2

Printables



Searching Cards
Includes helper sheet for teacher.

Classroom resources

- Paper
- Payment system such as tokens or marbles
- Pens

Learning outcomes

Students will be able to:

- Describe how the time taken grows with the size of the input, and most importantly how it grows in different ways for two different algorithms.
  Computational Thinking: Generalising and Patterns
- Describe how to compare number values for equality and inequality (greater than, less than).
  Mathematics: Numeracy
- Explain how they used decomposition to divide and conquer when doing a binary search.
  Computational Thinking: Decomposition
- Explain the range of the number of guesses for unsorted lists compared to sorted lists.
  Mathematics: Statistics
- Identify search algorithms for sorted and unsorted lists (sequential and binary search).
  Computer Science: Algorithms

Key questions

Imagine 31 numbers have been organised in ascending order in a list by a computer program. Now the program has to find a number in the list, but it can only look at one number at a time. Is it easier to find the number now, than if they were in a random order?

Potential answers could include:

- The information is organised in a way that makes it more efficient to find. By guessing one number and checking it, you can use logical thinking to eliminate the numbers below or above the number guessed because you know the numbers are in order.

- If they are in random order you can't use a strategy to find them quickly.

Lesson starter

We have 31 different numbers, one on each card. You can't see them but this time they are in order from the lowest number to the highest number. The numbers range from 0 to 1000. Can you find number 302

⊕ Teaching observations

You can adapt the range to suit what your students are working on in their mathematics lessons. This lesson focuses on sorted lists and we are using a range of numbers from 0 - 999. This activity works best if the numbers **aren't** sequential because, for example, if there are 50 cards in the range 1 to 50 and they are sequential and you ask a student to find the number 10 they will probably just look at the 10th card straight away! You can generate different sets of cards with various ranges of numbers here. It's best if the numbers aren't spread evenly so that it's very hard to guess where a particular value might be.

Lesson activities

Set up a line of cards, with the animal facing upwards. Have a payment system ready such as tokens for your classroom, counters, sweets, or marbles.

The game is even better if you have some real stakes - for example: I have 10 marbles each is worth 2 minutes of game time. For every token you use to find the number I'm thinking of, you will lose a marble.

Let's see how many guesses it takes to find the number: 302

Who would like the first guess? (Choose a student). Which animal should I turn over? Tell us why you chose that guess. (They should be selecting the card that is exactly half way. If they didn't, check with the class if they agree with the choice or could they add to the student's thinking to select the middle card. If they decide not to, that's fine; they will learn the hard way if they use a less efficient approach.)

Turn over the chosen card to show the number under it. If it's the correct one, you can stop, otherwise *remove that card and all other cards that the number can't be* (which will either be all cards to the right or all cards to the left of the chosen one) and take away one token from your pile. Repeat this process until a student chooses the card with your number on it; if they use all ten tokens then the teacher "wins".

How many guesses did it take to find the number?

With each guess, how many cards were eliminated from being a possibility? (Answer: half the cards could be eliminated with each guess if you picked the middle card.)

Did the students win because they guessed within 10 guesses or did you win because it took them longer?

Repeat this game until the students have won 3 times or you have won 3 times.

⊕ Teaching observations

The number of guesses required can be anything from 1 (if you are lucky the first time), to 5 (if you have chosen the middle number each time). Of course, they may use more than 5 guesses if they use a poor strategy. Most of the time they will need close to 5 guesses. This also means that students will always have tokens left if they use a good strategy, since the maximum number of guesses to find the number is 5.

Applying what we have just learnt

If any data is organised in order and a binary search is applied, then you can eliminate a lot of data quickly - cutting the number of items in half each time. As a slightly different example, if we were trying to guess a number between 1 and 1,000,000, then asking if the number is over 500,000 would eliminate 500,000 options in one question, the second question eliminates 250,000, the third question 125,000, and so on. So in 3 questions you have eliminated 875,000 numbers. With just 20 questions you can find the one number between 1 and 1,000,000. It's the same searching for objects that are sorted in descending order - each value that is checked halves the number of possible locations. Dividing problems in half makes them very small very quickly.

This general process is called "divide and conquer" - you break the problem into (two) parts, and deal with each part separately, in turn break them into two parts. Very soon you end up with very easy tasks, such as dealing with just one item. It's a great strategy for reducing any big task or challenge to achievable goals!

Lesson reflection

What is the algorithm for a binary search? Here is a possible answer:

- Ask to see the middle card
- Repeat until the correct number is found:
- Is the number greater than the number I want to find?
  - If yes, then keep the cards above that number,
  - Else, keep the cards below that number

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.

Algorithmic thinking

The divide and conquer process of repeatedly checking the centre card and deducing which cards can be eliminated, and which ones could still contain the number you are searching for, can be written as an algorithm. When you ask students to say which card to check each time they are actually articulating an algorithm and instructing you on how to follow it.

By describing this method with the following algorithm a computer or person can follow it without needing to know how it works, they can just follow the instructions and not have to think about how to actually do the task. It's important that algorithms are written like this, because computers can't figure out how to solve problems by themselves! A possible version of the algorithm is written under the lesson reflection.

Examples of what you could look for:

Who are the students who not only can explain the exact process to find the number, but are also the students who don't deviate from that process?

Abstraction

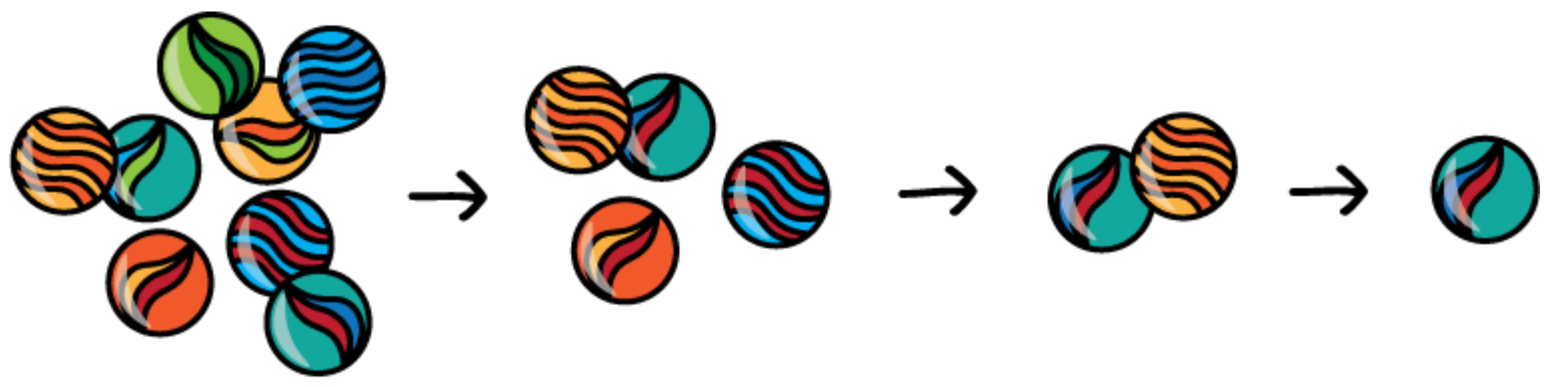
We can use the divide and conquer approach for more problems than just searching through an ordered list. We can use it to search through any set of objects that have identifying features.

We can also use it to help us sort things into order, which will be explored in the Sorting Algorithms unit.

Examples of what you could look for:

If you repeat this exercise but with the numbers underneath different objects, or maybe use different letters of the alphabet or different coloured discs to search for, who are the students that can see that these differences don't actually matter and they are still solving the same problem?

Decomposition



The Divide and Conquer method is entirely about decomposition. When we use divide and conquer to solve a problem, we are breaking the problem in half repeatedly, which soon decomposes it to a very simple case: a list of one item, which is very easy to search!

Examples of what you could look for:

Who are the students who are able to break the problem down into steps and then explain why each step is important?

Generalising and patterns

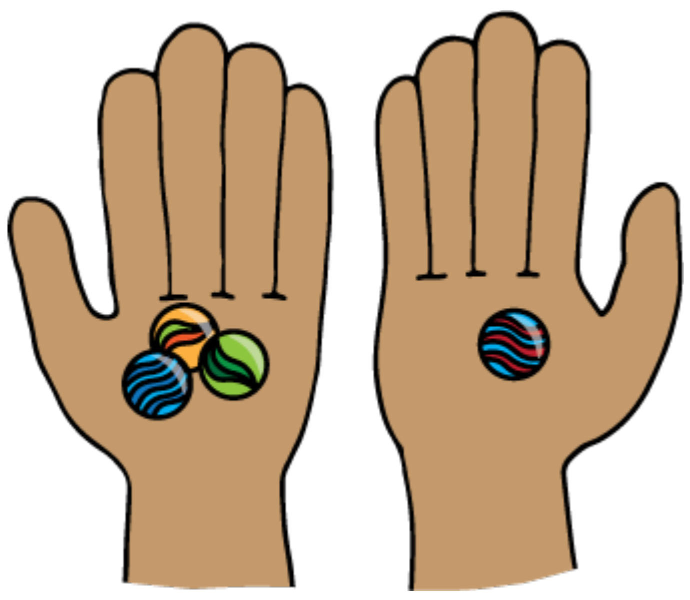
The key pattern to recognise in this activity is the process of eliminating half the possible cards by only looking in one, and that this is repeated over and over to accomplish the task.

Like we talked about in the lesson plan, the divide and conquer strategy is a pattern that appears frequently in computer science, and also in real life! It is an efficient and logical way of attacking many different problems where you are searching for something in a group of objects that have different identifying features.

Examples of what you could look for:

Who are the students who quickly identified the pattern?

Evaluation



Students can evaluate how well the divide and conquer method works by looking at how many marbles (or whatever payment you decide to use) they have left at the end of the activity. Older students can further examine the efficiency of this algorithm by calculating the maximum number of checks it would make for a different numbers of cards. You could compare this to the number of checks that a sequential search would need, and how these numbers change as you increase the number of cards.

Examples of what you could look for:

Who are the students who can explain the strengths and potential problems of using a binary search to find data? Can they explain why the maximum number of checks a binary search will make is much smaller than the maximum for sequential search?

Logic

To retain as many marbles (or whatever payment you decide to use) as possible it makes sense to try and eliminate as many cards as possible with each guess. That way you can cut down the number of cards to 1 as fast as possible. Students will generally be able to logically reason and recognise that the best way to do this is to check the centre card each time. If we check that card and compare it to the number we are searching for what does that tell us about all the cards to the left of that card? All the cards to the right of that card? Students can deduce which cards they can now eliminate based on the card they have checked.

Asking students to explain how they came to this conclusion is a great way to exercise their thinking skills, by getting them to articulate the logical steps they followed to come to this conclusion, and why it makes sense that doubling the number of cards only needs 1 more check. Understanding why divide and conquer will only ever require a specific small number of steps at most (for example it will never take more than 5 checks for 19 cards, or 20 checks for 1,000,000 cards) also requires a high level of logical reasoning.

Examples of what you could look for:

Which students instinctively go for the middle square when searching? They are likely logical thinkers who can deduce that since the numbers are sorted then the middle square will tell them the most useful information.