Locked-in

Age group successfully used with: 11 – adult

Abilities assumed: None

Time: 20-30 minutes, can be used within a longer activity

Size of group: anything from 2 to hundreds

Focus

What is an algorithm?
Search algorithms: linear search
Comparing algorithms
Computational Thinking

Syllabus Links

This activity can be used (for example)

- as a general introduction to computing topics such as what an algorithm is and how they can be compared from KS3 up.
- to introduce computational thinking problem solving from KS3 up,
- to teach specific syllabus topics such as:
  AQA A'level 3.1.1 Problem Solving: Linear Search
  KS3: understand several key algorithms (searching) that reflect computational thinking; use logical reasoning to compare the utility of alternative algorithms for the same problem

Summary

Explore the design of an algorithm to allow someone with locked-in syndrome to communicate. Locked-in syndrome is a condition resulting from a stroke where a person is totally paralysed. They can see, hear and think but cannot speak. How could a person with Locked-in syndrome write a book?

Aims

This activity aims to introduce computational thinking based problem solving, leading to an understanding of what an algorithm is, what linear search is and how algorithms can be compared on the basis of efficiency. It also illustrates how computational thinking is about more than just creating computer-based solutions. Computing is about solving problems for people.

Technical Terms

Search algorithm, linear search, protocol, efficiency analysis.

Materials

Nothing special required. A copy of the book 'The Diving Bell and the Butterfly' by Jean-Dominique Bauby is useful.

What to do

The Grab:
Explain you are going to look at how computer scientists solve problems. You are going to look at a very human problem. Describe life with locked-in syndrome:

> "One of the worst medical conditions I can imagine is locked-in syndrome. It leaves you with all your mental abilities intact, but totally paralyzed. It could happen to anyone, out of the blue, as a result of a stroke. If you are one of the lucky ones you can perhaps blink a single eyelid. Your intelligent mind is locked inside a useless body, able to sense everything but unable to communicate"

Explain that the activity is about how a computer scientist could help someone with locked-in syndrome…but it isn't about technology. It is about computational thinking. You are going to explore how to help a person with locked-in syndrome to communicate.

The Set-up:
Explain that a massively uplifting book is 'The Diving Bell and the Butterfly'. It is the autobiography of Jean-Dominique Bauby, written after he woke up in a hospital bed with locked-in syndrome.

Show the book to the class if you have one so that they can see it is a normal published book.

In the book, he describes life with locked-in syndrome. Bauby did have a way to communicate not only to write the book but also to talk with medical staff, friends and family. All he could do was blink one eyelid. He did it without any technological help at all. He just had a human helper to write down his words.

How did he do it?

The activity:
1) Get the class to suggest ways Bauby could have communicated with the helper by blinking.
   Likely suggestions include:
   a) blinking different numbers of times for each letter: 1 blink means A, 2 means B
   b) using morse code
   c) The helper reading through the alphabet A, B, C … He blinks when they get to the letter he is thinking of.

   Point out that the class are thinking like computer scientists – doing computational thinking – in coming up with such methods. Have the class discuss the advantages and disadvantages of each with you.

   Bauby used the last alternative where the helper goes through the alphabet. Illustrate this by talking though communicating the first few letters of the book:

"Through the frayed …" It involves the helper reading the alphabet A, B, C up to T, Bauby blinking and the helper writing down T. The helper starts again from A up to H, he blinks and they write it down, etc.

2) Get the class in pairs and have them communicate messages to each other this way.

3) Discuss with the class how well it works – are there any problems to solve. Can they think of any improvements? Can they come up with a solution that really works?
   Likely suggestions of problems include:
   a) the need to deal with extra characters: punctuation, digits, etc
   b) what you do when the person blinked by mistake?
   Improvements suggested might include
   a) guessing a word before it is finished (this is essentially doing the same as predictive texting), eg A-N-T-E guess antelope.
   b) changing the order of the letters ask to ask common ones first – E is most common so ask about it first.

4) Now get the class to work out how long it would take to write the book this way. Point out we can use the number of questions asked (letters the speaker has to say) as a way of measuring work done. Focus on communicating one letter. Have them think about which letter would be
   a) the best case (i.e., take fewest questions to work out)?
      the letter A and you get it in 1 question
   b) the worst case (i.e., take most questions to work out)?
      the letter Z and you get it in 26 questions

Now get the class to think about how many questions on average does it take over all the letters in the book? (13 – roughly for every A there is likely to be a Z for every B a Y and so on.

This means if we multiply the number of letters in the book by 13 we get a good estimate of how much work is needed to communicate the book. If we then know roughly how long it takes for the helper to ask a question we can get an estimate of the time taken (just multiply the time for a question by the number of questions).

Summing up:
We have looked at how a person can communicate when all they can do is blink an eyelid. What is needed is an algorithm both people agree on to communicate. The algorithm Bauby used is a variation on the search algorithm called linear search. In linear search you find things by notionally lining them up one after the other and checking each in turn until you find the thing you are looking for. If you get all the way to the end without finding it then you know it isn't there.

We saw a simple way to evaluate algorithms – to see how quick they are in terms of amount of work done. We can work out the best and worst cases. They give us limits on how good or bad it could possibly be. We can also work out the average case, which gives a good estimate of the actual work done.

For someone with locked-in syndrome to communicate in this way we needed an algorithm that was split in two – one part for the person communicating the letters and one for the person writing them down. Computer scientists call this a protocol. Agreed protocols are needed to get two computers to communicate over a network: it is a very similar problem. All they can do is send 1s and 0s rather than blink or no-blink.

We have also been doing computational thinking. We have used **algorithmic thinking** to devise an algorithm: a computational (i.e., algorithmic) solution to a problem. We have explored potential problems with it and thought about ways to fix them. We have thought about improvements to the basic algorithm. To do that we have **translated solutions** from one problem to another: predictive texting helps us write texts faster and a similar solution improved our algorithm.We have also applied a simple form of **analytical thinking** to the problem to determine how good our solution is. None of this has directly involved computers or any electronic technology at all. Computational thinking is not just about computers it is about **solving problems for people.**

Variations and Extensions

1) This activity was originally developed to be combined with the 20 Questions activity (see below) leading to much faster algorithmic solutions that solve the same problem. Having discussed Bauby's method and variations, point out that actually at worst only 5 questions are needed to work out any letter, not 13 on average. Explain that by switching problems you can show that everyone knows the right questions to ask. You need to play a game of 20 Questions… Once that activity is completed bring the class back to locked-in syndrome. Can they now devise a set of similar halving questions that would always get a letter of the alphabet in 5 questions.
2) Link this activity directly to linear search by describing searching an array using linear search. For classes that can program, have the class write a program to search an array.
3) Apply the efficiency analysis to other suggested solutions. For example how efficient in the best case, worst case and average case is the algorithm where you blink once for A, twice for B and so on. How does the number of blinks needed compare with the number of questions asked in Bauby's method?
4) For groups that can program, have them write programs to implement the algorithms and create a prototype for a program that could be used by someone with locked in syndrome. It should flash up letters on the screen. Blinking or not should be simulated by the space bar being pressed or not. It could thus be used by the helper to record letters.

Further Reading

Computational Thinking: Searching To Speak

This activity combined with the 20-Questions activity (below) written up in a booklet. It is available from http://teachinglondoncomputing.org/resources/

Links to other activities

The following activities are also available via teachinglondoncomputing.org

20 Questions

Play 20 questions and see you already know the key to efficient search algorithms

This activity introduces the idea of divide and conquer problem solving in the context of search algorithms. It also introduces the idea of efficiency analysis as a way of comparing algorithms.

The intelligent piece of paper

Take part in a test of intelligence against an intelligent piece of paper!

This is a good introduction to what an algorithms is and what a computer program is before looking at search algorithms. It can also be used to start a discussion on what it would mean for a computer to be intelligent.

Winning Games: the perfect Tic-tac-toe player

Create a set of instructions that would allow anyone to play Tic-tac-toe perfectly.

This is a good follow on activity from the intelligent piece of paper – now create your own! It introduces programming and explores how a computer is able to win at board games like chess. The emphasis is on programming being about solving the problem rather than about being able to write in a programming language.

Live demonstration of this activity

Teaching London Computing give live sessions for teachers demonstrating this and our other activities. See http://teachinglondoncomputing.org/ for details. Videos of some activities are also available or in preparation.

Department for Education
SUPPORTED BY MAYOR OF LONDON
COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE