Algorithmia
an introduction to
# problem solving

- A didactic unit about mathematics, programming, solutions, and robots -

- ## Introduction

  The objective of this one / two hours class for very young children is to perform a brief journey on the basic knowledge, principles and skills required to address and attempt to solve any kind of problem, pointing out collaboration and communication as main process cornerstones. By introducing basic concepts and techniques of mathematics and computer programming - and putting them into practice in a couple of simple collaborative games that need almost no materials (nor computers)- students play, experience and reflect on necessary and meaningful possibilities and values to address and solve problems.

- ## Objectives
  - Knowledge
    - Basic concepts of mathematics: Algorithms, problem, solution, groups, sequence, symbols, signs, operands, results, grids, maps, orientation, coordinate system.
    - Basic concepts of programming and computing: language, programming language, program, programmer, function, command, script, input, output, compiler, runtime, recursion, repetition, robot, etc.
    - Basic concepts and methodologies for problem-solving.
  - Skills
    - Graphic and oral expression, spatial orientation, autonomy, capacity of situational and contextual analysis, teamwork and participatory skills applied to problem solving.
  - Procedures
    - Group play, participating in assembly to evaluate processes and outcomes, spatial orientation, creation and use of graphic symbolic language applied to real situations, creation of maps, drawing, writing and executing programs-algorithms.
  - Values and attitudes
    - Appreciation of communication - collaboration – respect - solidarity - team-work as a set of fundamental tools for problem-solving and communal well-being.

- ## Activities

  Single Session (50 / 100 min, 2 handkerchiefs to blindfold the «robots», 2 balls or boxes, a blackboard, chalk, paper and pencil)

  - ### Hello world! (5 min)
    - Halo! (…) Hello world is the easiest program programmers use when they want to introduce a new programming language. And it's usually just a program that displays those words, a simple greeting. We will learn a little bit on programming today. Who can tell what programming is good for?
    - Yes, to solve problems, exactly. Today we are going to talk about problems and how to solve them... Anyone knows what is a problem? Someone has an example? And what is a solution?
    - A long time ago, a man called Muhammad ibn Mūsā al-Khwārizmī, -Al-Khuwarizmi for short- invented algebra, which means "change". He invented and described a set of easy operations, which consisted mainly on changing things -numbers, quantities- from one place to another, which helped enormously to solve math problems better than ever before. It was a Revolution. His new methods were so good and practical that today we call the series of steps / instructions to follow in order to solve a problem, an algorithm.
    - Al-Khuwarizmi's name means “from Khuarizmi” which is an east persian city in Uzbekistan called Khiva now, which is a bit of a labyrinth with some of the most beautiful walls, doors and towers you might ever find. But he lived in Baghdad, and some say he knew the thief of Baghdad, and he worked at The house of wisdom. And it's thanks to him that we use the numbers we use now here in Europe.
    - It's possible to extend a bit here on the two mathematical operations (reduction and balancing [Algebr wal muqabala (kan oversettes med «gjenopprettelse og forenkling»)]) that are core to his works, one of which gives the name to Algebra, adjusting it to the children's level. Though they are both easy enough they were indeed a massive advance for science.
  - ### Let's play!

    Introduction to basic computer concepts and problem solving, and explaining the first game (5 min)
    - Programming is used mainly to solve problems. And a program is an algorithm for solving a particular problem. Programmers write programs. They use programming languages, which are like human languages. These languages have words, commands or instructions and names or adjectives to define things, and express the actions (movements, relations), name the actors (people, things) or describe the context (time, space, mood), as any other language.
    - We are going to define a simple problem and we are going to use a chalk on the blackboard (or a pen in the whiteboard) to create a simple programming language and thus help us solve our problem.
    - We need to get out of the class, but to open the door we first need to find the key. The key is a ball that we have secretly placed in the floor somewhere. We need a few volunteers.

    - A Programmer writes the commands: move forward, turn right, turn left, look down on the floor for the key, open the door.
    - Then she will decide which one to use as necessary by pointing at it.
    - A "real-time" "compiler" will read aloud our program.
    - A "prisoner" or "robot" will follow the instructions she receives.
    - We do a simple test so everyone knows how it works, and then we choose volunteers to make a race where two teams will try to find the ball and get out first.
- Out of the Maze - take 1 (10 min)
    - Two teams compete to get the prisoner out of the labyrinth as soon as possible. The prisoners go this time blindfolded. There should be a level of confusion. And that is good, best if there is a bit of chaos. The two teams at least talk at the same time, disorienting the prisoners as they receive multiple instructions, the other students in the class, who have not been instructed to abstain from participation, will probably try to help or confuse the robots, they'll shout (there!, there!, to the right!, No! The other way! Etc.).
    - If chaos doesn't occur spontaneously, it should be encouraged, (using synonyms for each command (move and walk, turn and turn around, maybe doing gestures of impatience to "motivate" the more restless, giving vague instructions, cheering loud yourselves or even “cheating”). The exercise should be interrupted before any team can get out of the class if possible.
- Sharing the experience so far (5 / 10 minutes)
    - Now the class must share their thoughts on the game and think together how to improve it.
    - Three main key concepts should surface. Noise, synchronicity, and names.
    - Alumni must debate and find solutions to limit the noise and to avoid the overlapping yelling of arbitrary commands, to unspecified targets, in order to avoid confusion.
    - Also we can shorten the programs, using numbers next to commands, instead of repeating each many times. (5 --> instead of --> --> --> --> -->)
- Out of the Maze - take 2 (5 / 10 minutes)
    - We repeat the former game, but now only participants play while the rest of the class pays attention in silence.
    - Teams play taking turns, each issuing only one shorter, better defined command, calling the robot that must perform the command by name. We must see that now it all works in a much more fluid way.
    - If time is not an issue, they will probably want to play again, since most of them didn't get a chance to play an active role in the game. But if so, we might do a couple of things here. One, introduce some new concept in the pause -like the origin of the word robot, and two, maybe change the rules a bit, but letting them actually change them after some debate, with their own ideas, maybe suggest that instead of racing one team against the

other, the game be this time around about collaborating all together to do something?

- Robots (5 / 10 minutes)
  - Who knows what a robot is?
  - In 1920, Karel Capek wrote and took to the theater with great succes a play about men and women that are made in a factory, and look exactly like normal people, except they are not borned from a mother, but fabricated in the factory, the “Rossum's Universal Robots (R.U.R.)” factory, which was also the name of the play, and (supposedly) have no feelings, and (supposedly) can not have children, and then they were (not supposedly) sold as slaves to work and work and work and nothing more than work. Anyone guess how it ends? But it was Karel's brother, Josef, who suggested the word robot, and also used it in a short story, but never mind.
  - Robot was short for robotnik, from Czech robotnik "forced worker," from robota "forced labor, compulsory service, drudgery," from robotiti "to work, drudge," from an Old Czech source akin to Old Church Slavonic rabota "servitude," from rabu "slave," from Old Slavic orbu -, from PIE orbh- "pass from one status to another" (see orphan). The Slavic word thus is a cousin to German Arbeit "work" (Old High German arabeit).
- Out of the Maze - take 3 (10 min)
  - Now we will use the commands we have learned / created to plan each of us our own escape.
  - Using paper and pencil we will each -or in couples, better- write the algorithm to get the ball (key / treasure) in the middle of the classroom and get out through the door.
  - Some will write the program before you have had time to explain the task and run to show it to you.
  - Others will want to stand up and follow their way to the key and door and write the program as they perform it so they can count the necessary steps precisely.
  - Some others might think of drawing an overhead plan of the class and drawing the program commands in the plotted maze itself.
  - They should be encouraged to help each other and share their tactics to tackle the task.
- Conclusions

  Closing up, conclusions and goodbye (5 / 10 minutes)
  - First off, we share the different approaches the class has found to perform the task: direct writing, walking along, plotting a map, assigning tasks, finding out the shortest route, easing up the writing of the program, maybe using new commands, etc...
  - Then, we ask what was it all like and if they think what they did today and what they learned will be useful in the future and how.
  - They should express freely before we summ it all up, possibly using their inputs.
  - We have followed a series of steps that are common to almost all situations in which we need to solve a problem or perform a task, both individually and in group.

- First we choose or define the problem.
- We use examples to clarify. We shape them together. Clear up the doubts before starting off.
- We practice the problem, we get into matters, to get out of the labyrinth, in our simple case. Using the tools we have agreed and created.
- A programming language that we have agreed upon and we all understand, with some codes, symbols that indicate actions.
- We fail.
- We realize some things don’t work out well, they can be improved.
- Noise, chaotic simultaneity, unnecessary repetition of commands, robots that don’t know the commands are for them, etc.
- Then we stop and think as a group, and talk on what went wrong and what went right, and find solutions to each smaller problem.
- We ask doubts. Very important, asking.
- If there’s something we don’t know about the problem, if maybe we even need to redefine the problem, or if we lack data, or need to explore the rules of the game and see if we should maybe change them.
- We ask. Ourselves and others. And if we still need more, then we ask, who knows more about this?
- Are there any books on labyrinths and how to get out of them, maybe we should go to the library? Somebody who is an expert on escaping labyrinths?
- Then we try to get out of the labyrinth again, this time with new ideas and new methods.
- We have perfected our programming language, and the way we run the programs, so it all goes better this time. We are all a bit better at escaping labyrinths.
- Define, try, share, redefine if necessary, ask, ask who knows more, plan, create tools, symbols, or discover or choose new concepts and methods, try again, share your knowledge and explore again, find new, even more exciting problems. This is one possible algorithm to create or find new algorithms to solve about any problem you might ever face.
- Any questions? Did everyone have fun?
- One conclusion that should come up is that it’s easier to solve problems together, sharing information and experiences, experimenting the problem together and talking about it and asking and questioning and redefining as necessary, working in teams.
- Then maybe the teachers should get out of the classroom reading out loud the program they use to get out. «Turn left, turn left, walk, walk, walk, shake hands, walk, turn right, shake hands, smile, walk, walk, walk, walk, walk, walk, walk, walk, turn left, open door, turn around, wave hand, say goodbye, smile, walk backwards, close door, keep smiling...»

The leap from mathematics to computers

- We started talking about a man whose name is used to define the solutions of problems. So we should end with another person. Do you know who is considered the first person to ever imagine a "computing machine" and recognise its full potential, to solve not only mathematical problems, but any kind of problems, actually, and is also considered the first person that ever wrote a computer programme, and thus, the first ever programmer? Ada Lovelace!

- **We want to know more!**
  - On Al-Khuwarizmi
    - https://en.wikipedia.org/wiki/Muhammad_ibn_Musa_al-Khwarizmi
    - https://snl.no/Mohamed_Ibn_Musa_al-_Khwarizmî
    - https://no.wikipedia.org/wiki/Al-Khwârizmî
  - On Algorithms
    - https://no.wikipedia.org/wiki/Algoritme
    - http://ordbok.uib.no/perl/ordbok.cgi?OPP=+algoritme&ant_bokmaal=5&ant_nynorsk=5&begge=+&ordbok=begge
  - On Algebra
    - https://en.wikipedia.org/wiki/History_of_algebra#Al-jabr_wa'l_muqabalah
    - https://en.wikipedia.org/wiki/Algebra
    - https://no.wikipedia.org/wiki/Algebra
  - On Programming
    - http://oppgaver.kidsakoder.no/index.html
  - On Ada Lovelace
    - https://codepen.io/mi-mina/full/dXwjvk
    - https://en.wikipedia.org/wiki/Ada_Lovelace
    - https://snl.no/Ada_Lovelace
  - On Robots
    - http://ordbok.uib.no/perl/ordbok.cgi?OPP=robot&ant_bokmaal=5&ant_nynorsk=5&begge=+&ordbok=begge (the difference between bokmål and nynorsk is remarkable)
    - https://snl.no/robot
    - https://snl.no/Karel_Čapek

- **Credits**
  - An idea of Sergio Daroca, performed the first time at the Huerta Santa Marina school in Sevilla with the support of Raquel Sancho and Ara Urdambidelus and the fantastic children in 2A and 2B.
  - Some more at http://www.flashdance.es
  - This is set by the author on the public domain, you can do with it whatever you want.
  - So please use it, share it, improve it, share it again!