1+1=3
The symbiosis between maths and computing in education

Simon Peyton Jones
Epic Games and Computing at School

ENSICO Conference
March 2022

Re-envisioning computing education

"Education should prepare young people
for jobs that do not yet exist,
using technologies that have not yet been invented,
to solve problems of which we are not yet aware."

Richard Riley

USA Secretary of Education
1997-2000

Disciplines

Skills

Disciplines

Ideas, knowledge, principles, techniques, methods

Maths, science, history, English

Skills

Artefacts, devices, programs, products, organisations, business

Presentation skills, metalwork, textiles, food technology, teamwork

Ideas

Write
Create
Understand
Knowledge

Knowledge gives you power over magic

Technology

Read
Consume
Use
Magic

Clarke's Third Law

Any sufficiently advanced technology is indistinguishable from magic

Computer Science

ICT

Information and Communication Technology

Spreadsheets, databases, Powerpoint, web, internet, audio, video, e-safety

This was the situation in the UK during the 2000's

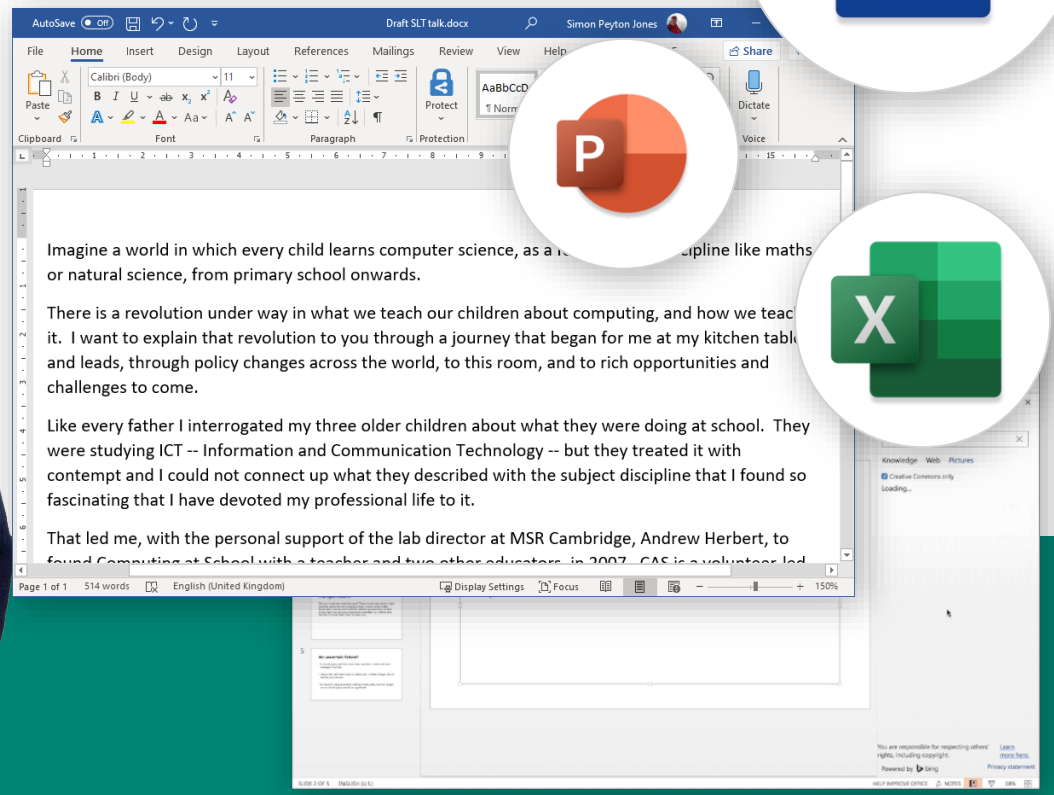
Discipline

Computer science
(Informatics)

Skills

Digital skills

Around the kitchen table in 2007…
ICT was a school subject



Computing at School

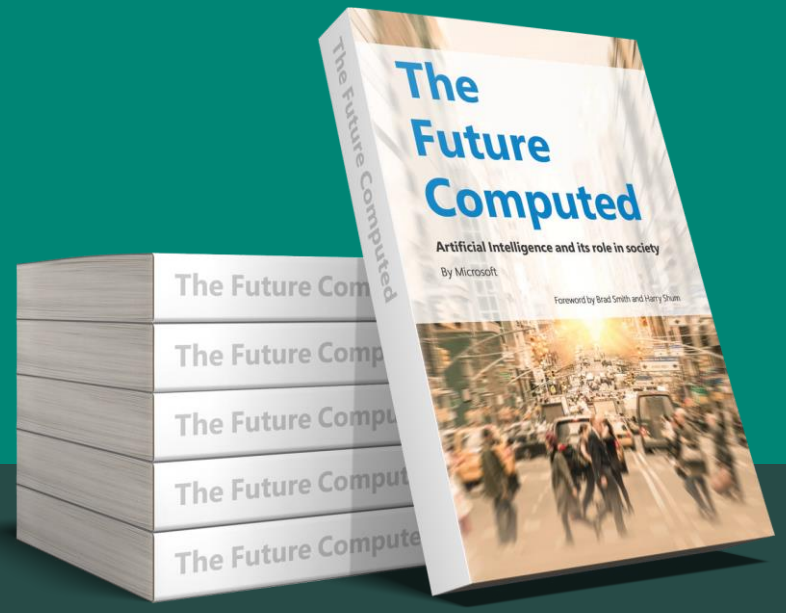
A grass roots, volunteer-led community of practice for computing educators

MEMBERS	RESOURCES	CAS LOCAL COMMUNITIES
32,585	4,842	339

SCHOOLS	LEAD SCHOOLS	MASTER TEACHERS
1,791	578	512

Vision

Every child should learn computer science, as a foundational discipline, just as they do maths and natural science, from primary school onwards.



Ideas, not technology

Not even primarily about computers.

Every child

Not just future software developers, not just boys, not just middle class.

From primary school

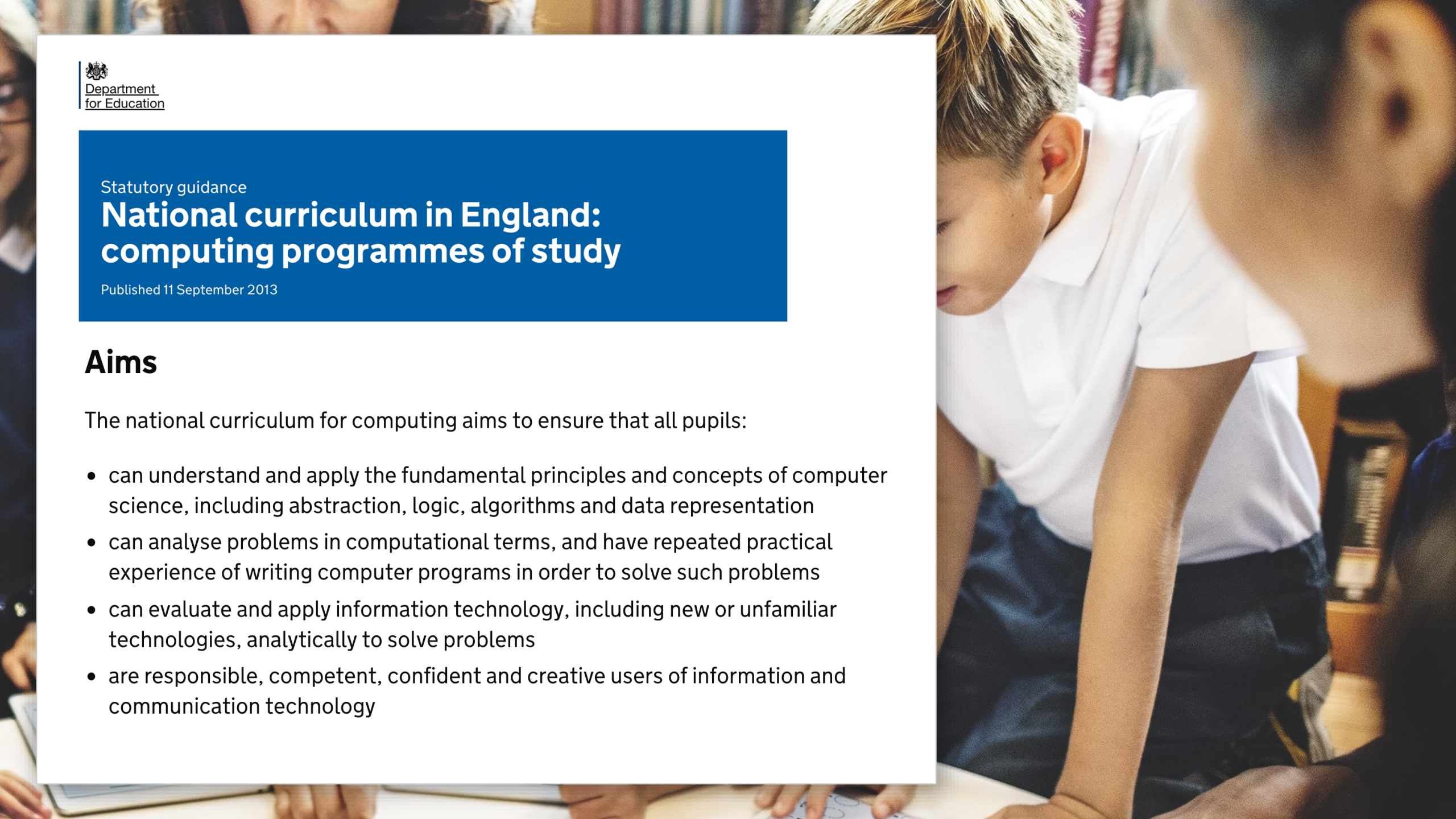
Underpin confidence, self-efficacy, and informed judgement to navigate a ubiquitously digitised world.

Discipline, not just skill

In particular, not just coding.

Authors / informed critics

Not mere passive consumers



Department for Education

Statutory guidance

National curriculum in England: computing programmes of study

Published 11 September 2013

Aims

The national curriculum for computing aims to ensure that all pupils:

- can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
- can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems
- can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems
- are responsible, competent, confident and creative users of information and communication technology

The UK journey

Computing at School

CAS is formed

2008

CAS curriculum
35 pages

2009

2010

2011

Shut down or restart

2012

New curriculum published
2 pages

2013

New curriculum launches

2014

NETWORK OF EXCELLENCE COMPUTER SCIENCE TEACHING

2015

2016

After the Reboot

2017

National Centre for Computing Education

2018

Review of the National Curriculum of England

National Centre for Computing Education

Teach Computing

https://teachcomputing.org

Inspire, train, support, equip, encourage teachers to teach the most fascinating, creative, and economically important subject on the planet

From age 6 through to 18 (i.e. K-12)

£82m over four years	**3,500** secondary schools	**16,000** primary schools

Department for Education

The situation in Portugal

Pilot project, started 2019

Now running with 1000 students age 8-15, 1 hr/week – lots of enthusiasm!

Close cooperation with Ministry of Education

Starting by **adding computational thinking to maths**,
using Jupyter notebooks, unplugged activities, and Haskell

Strong points

- Run a pilot, and scale up
- Teachers playing a leading role
- Ministry of Education as a partner

Maths and computing: the opportunity

England: Computing is now positioned as a foundational subject that, alongside maths and natural science, every child learns from primary school onwards.

Portugal: introducing the subject discipline of computer science through mathematics, again at all ages.

How can we take the potential symbiosis between maths and computing from aspiration to reality?

Symbiosis (from Greek συμβίωσις, *sumbíōsis*, "living together", from σύν, *sún*, "together", and βίωσις, bíōsis, "living")[2] is any type of a close and long-term biological interaction between two different biological organisms. In 1879, Heinrich Anton de Bary defined it as "the living together of unlike organisms". [Wikipedia]

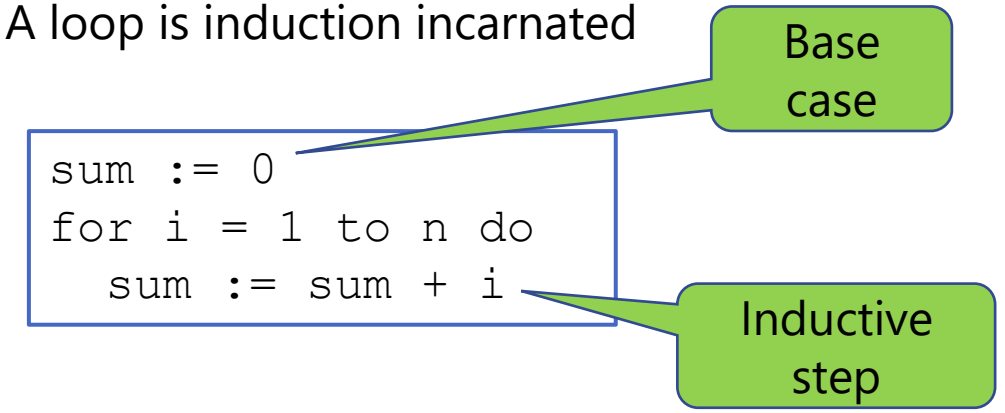
Unifying idea

Computing is maths made incarnate

Maths made

- Tangible
- Executable
- Concrete
- Experimental, exploratory
- Motivated, by a context

A loop is induction incarnated



Theorem: $\Sigma_1^{\mathrm{N}}\, i \;=\; N * (N+1)/2$

What we want to achieve

‘M’ realised she could replace the repeated steps by this for loop:

```
function main ()
    turtle → forward(100)
    turtle → right turn(90)
    turtle → forward(100)
    turtle → right turn(90)
    turtle → forward(100)
    turtle → right turn(90)
    turtle → forward(100)
end function
```

```
function main ()
    for 0 ≤ i < 4 do
        turtle → forward(100)
        turtle → right turn(90)
    end for
end function
```

Computing in action

M's schoolteacher:

Here I have M, self esteem going through the roof and she has associated this computing success with maths.

Over the last couple of weeks, she has solved [maths] problem after problem, met target after target – she is truly flying.

M is going great because of a positive experience in a computing workshop in London.

Danger 1

Maths and computing perceived as being in competition for limited timetable slots.

- Zero sum game.
- Silos

Opportunity 1

Maths and computing enrich and inform each other

- 1+1 = 3

Danger 2

Computing is perceived as a useful, even trendy, decoration to Real Mathematics

Need to go beyond
“look good”

Opportunity 2

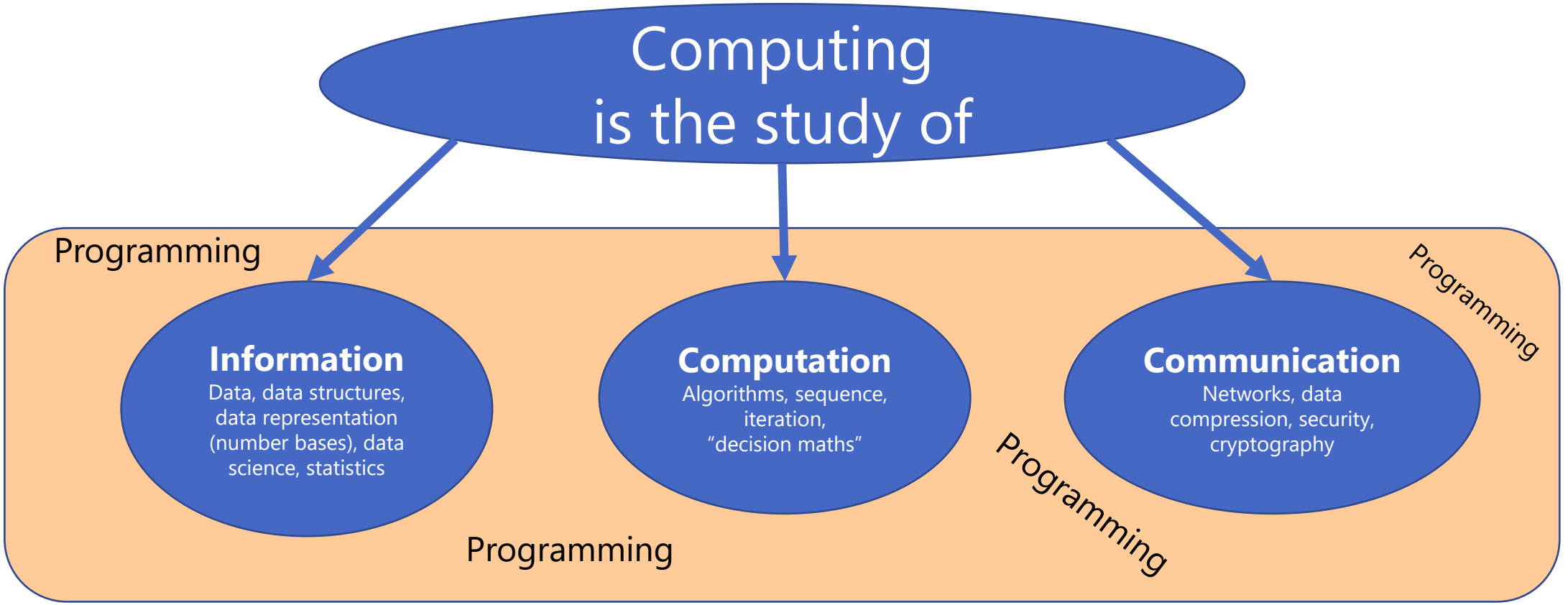
- Computing and Maths are genuinely symbiotic
- Not just a way to do **maths better**, but a way to do **better maths**.
- (and better computing too)

Digression: I am not talking about

- How to do all subjects better using computers
  - e.g. *"DEFI maps the educational logic of the new Internet Age through research that explores the pedagogical affordances of new technologies, which means the ways of teaching and learning that they make possible for education, ways which were either not possible or more difficult to implement before"*

- How to do maths better using computers
  - E.g. Conrad Wolfram *"in real world maths computers do almost all the calculating, but in education maths people do almost all the calculating"*

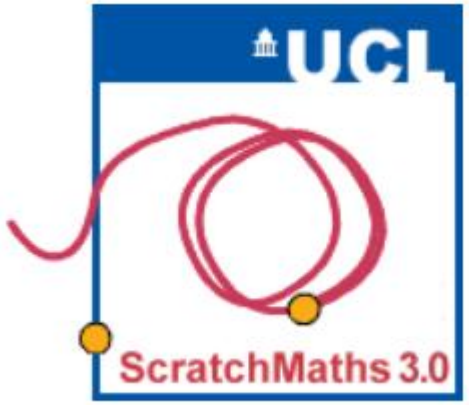
Both these are very important, but they aren't my focus



Common themes between computing and mathematics

- Abstraction
  - From real world to what you model
  - Doing it once: algebra, functions
- Precision in thinking
- Logic

UCL Scratch Maths

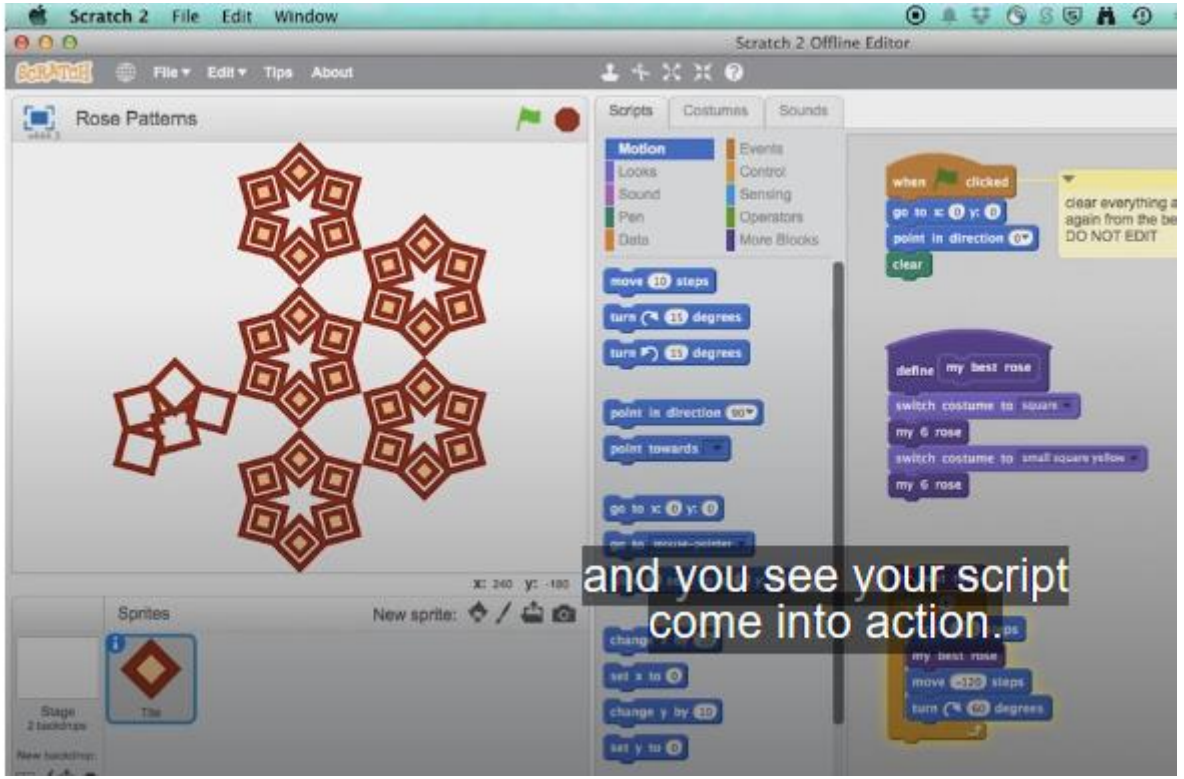


Maths is difficult in part because of the language in which it is expressed. *Can we find a different language - and set of ideas and approaches - that is more open, more accessible and more learnable*. And can we find it without sacrificing what makes mathematics work?

Our tentative answer is 'yes' - the language of programming might - if we design it right - be just such a language.

Celia Hoyles and Richard Noss

https://www.ucl.ac.uk/ioe/research/projects/ucl-scratchmaths



…it’s like magic in front of your eyes.

Bootstrap World

https://bootstrapworld.org/

BOOTSTRAP

Equity • Scale • Rigor

The Bootstrap:Algebra Pathway applies mathematical concepts and rigorous programming principles to creating a simple video game

In this series of 19 lessons, students create a simple, 3-character game involving a player, a target and a danger. They design what each character looks like, and use mathematical concepts such as coordinate planes, order of operations, ratio and proportion, domain and range, function composition, word problems and the distance formula to detect collisions, handle keystrokes, and determine how they move and interact.

Algebra **Reactive** **Data Science** **Physics**

Paul Curzon (of cs4fn fame)

https://teachinglondoncomputing.org/interdisciplinary-computational-thinking/computer-science-and-maths/

COMPUTER SCIENCE, MATHS AND NUMERACY

Computational Thinking overlaps a lot with the way mathematicians think. Both are ultimately about solving problems. Computer Science focusses a lot more on the idea of algorithms as solutions as well as the importance of logic. Computer Scientists rely on maths a lot, and increasingly mathematicians are relying on computational thinking. Algorithms are even being used to prove unsolved mathematical problems that human mathematicians have struggled with for years.

Computer Science topics can also be used to support numeracy, by giving an exciting, fun real world context for the importance of numbers and mathematics.

Numeracy: numbers, counting, practical use of numbers

- Pixel Puzzles
  - These simple colour by number puzzles show how computer images are represented as numbers and come in several variations. In the simplest each number tells you how to colour a square. In others you are given a number telling you how many consecutive squares to colour.

Numeracy: multiplication tables

- Multiplication Table Pixel Puzzles
  - These simple colour by number puzzles are a fun way to learn about the patterns in the times tables, and practice your times tables based on a deeper understanding, while also exploring how patterns can lead to new algorithms.
- 9 Times table Algorithms
  - Explore algorithms that allow you to work out the 9 times table, exploring what an algorithm is, efficiency, patterns and using algebra to prove algorithms always work.
- Tudor Multiplication
  - Learn a cunning Tudor algorithm for doing those tricky times tables you can never remember, and prove it does always work.
- Happy Times maths card game
  - Happy Times is a fun Happy families / Rummy style card game where you collect sets of cards in the same times table to win. Playing helps you learn about the fun of numbers with a focus on times tables, multiplication, division and prime numbers. Start to gain an understanding of the patterns in numbers.
- Ten maths card game
  - Ten is a fun maths card game that involves

Ontario curriculum (George Gadanides)

http://www.edu.gov.on.ca/eng/curriculum/elementary/math.html

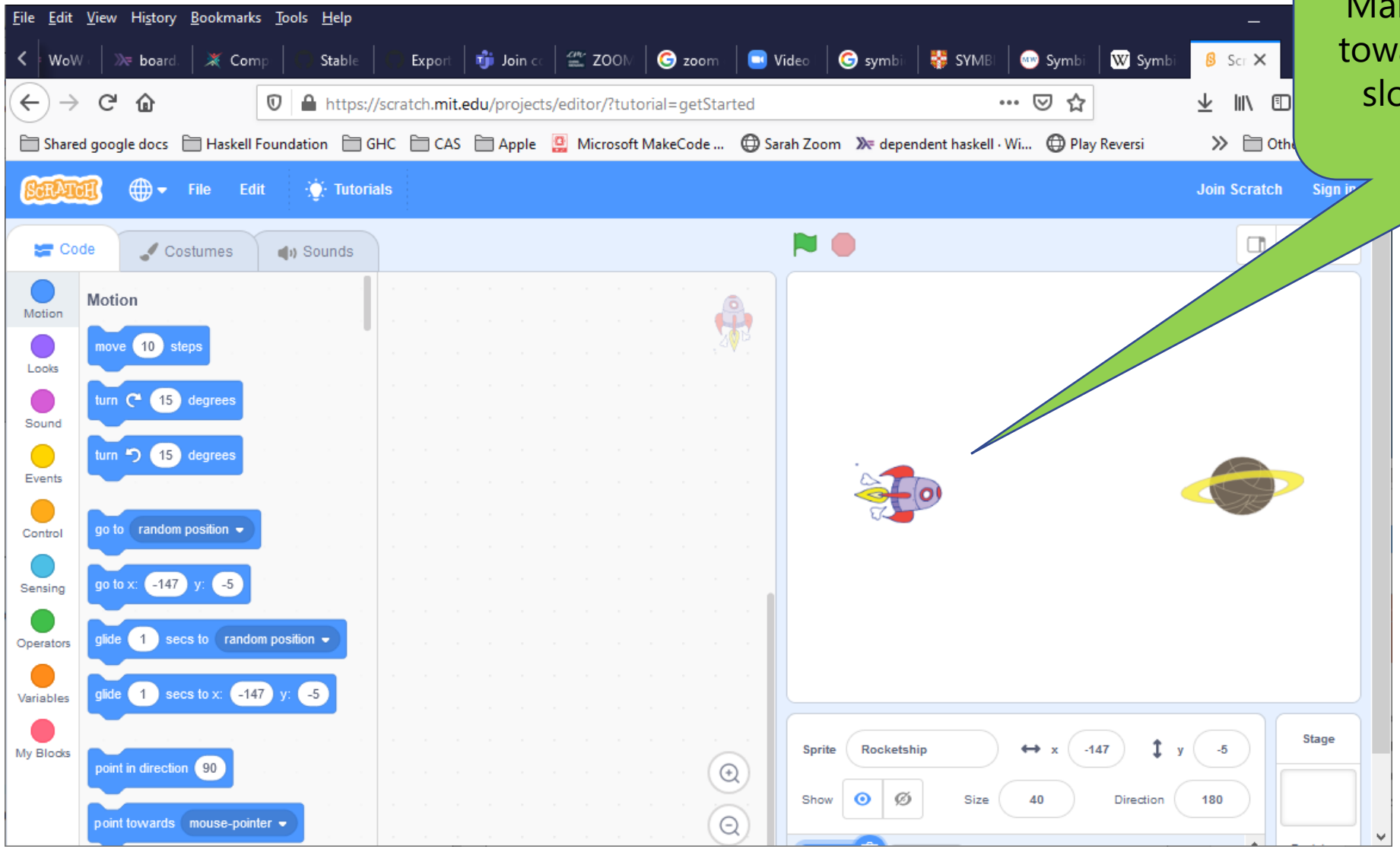
This is a very different learning experience than say a Socratic lesson where the teacher shows, step-by-step, how to edit inequalities, to make them more complex-looking or to simplify them, and to solve them and then students practice what the teacher demonstrates for similar examples. Using the 3 lines of Python code above, students may quickly edit the code, run the code to get immediate feedback, and learn both coding and mathematics concepts, playfully and incidentally.

Computer is pitiless, but also non-judgemental

Some examples

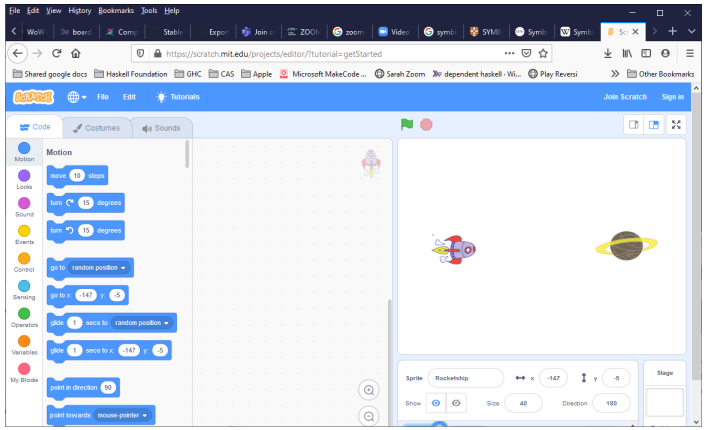
Caveat: I am not a teacher!

Example: computing movement



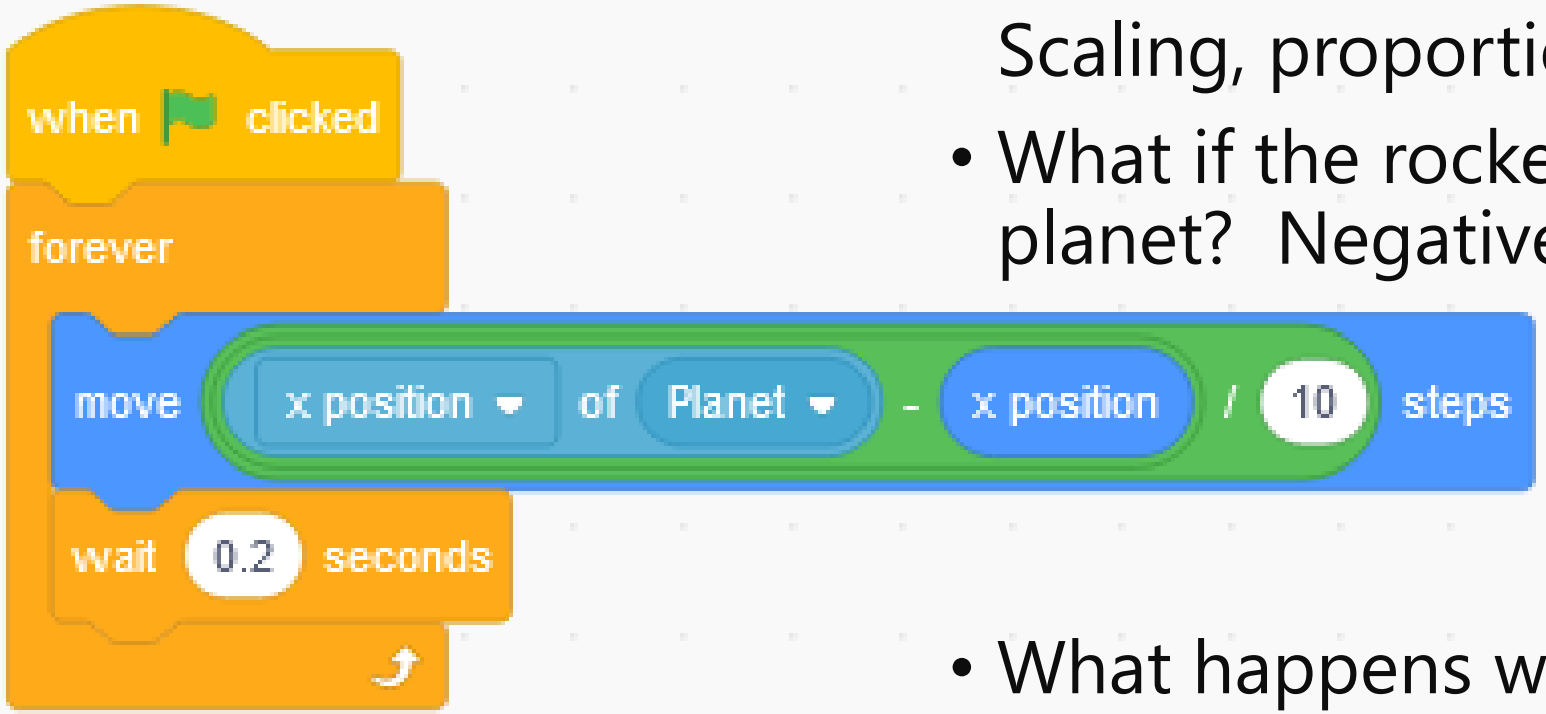
Task

Make the rocket move towards the planet, but slow down as it gets closer



Example: computing movement

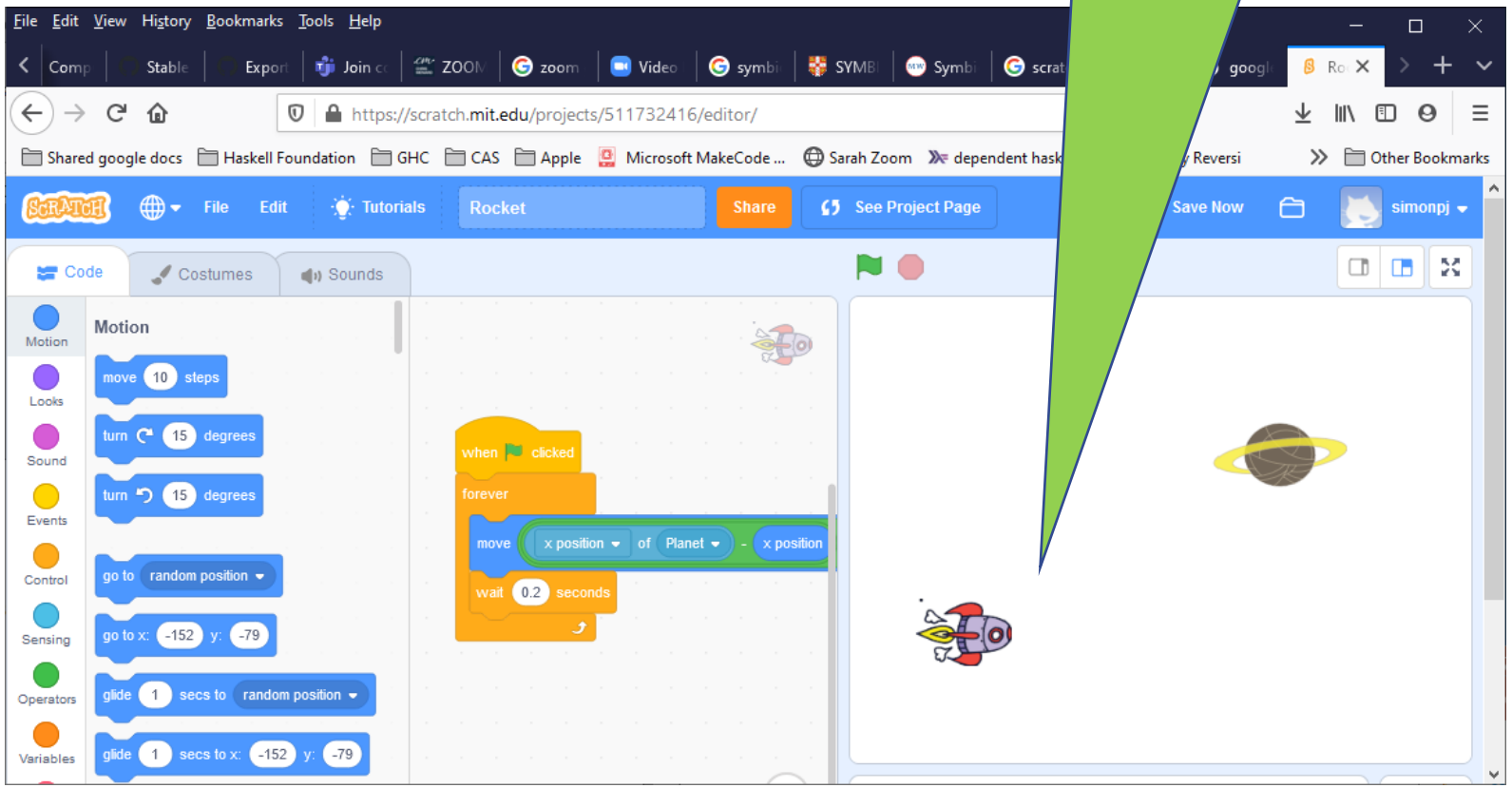
- Difference in x-coordinates
- Visual representation of parentheses
- How to make it move faster/slower. Scaling, proportion.
- What if the rocket is the other side of the planet? Negative numbers.



- What happens when it is really close?

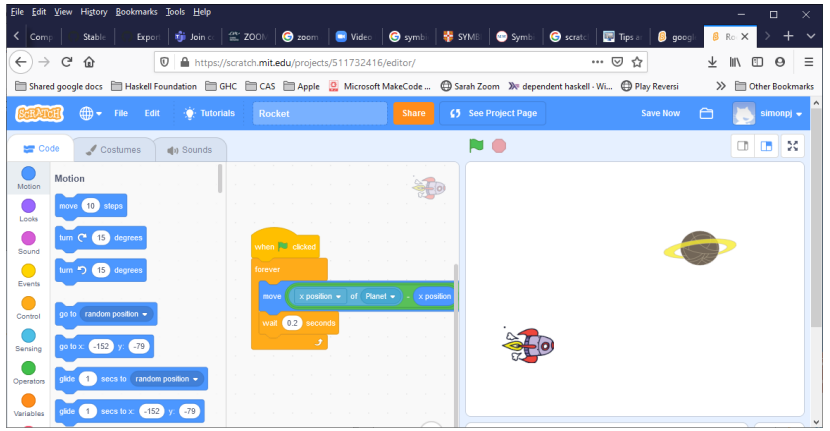
Two dimensions

Task
Make the rocket move towards the planet



- Compute x and y changes separately
- Vectors

Example: computing movement



Can go pretty far…

- Include mass and velocity
- Inverse square law
- 3-both problems
- Back to spreadsheets: table of position/velocity vs time
- Graphs to visualise

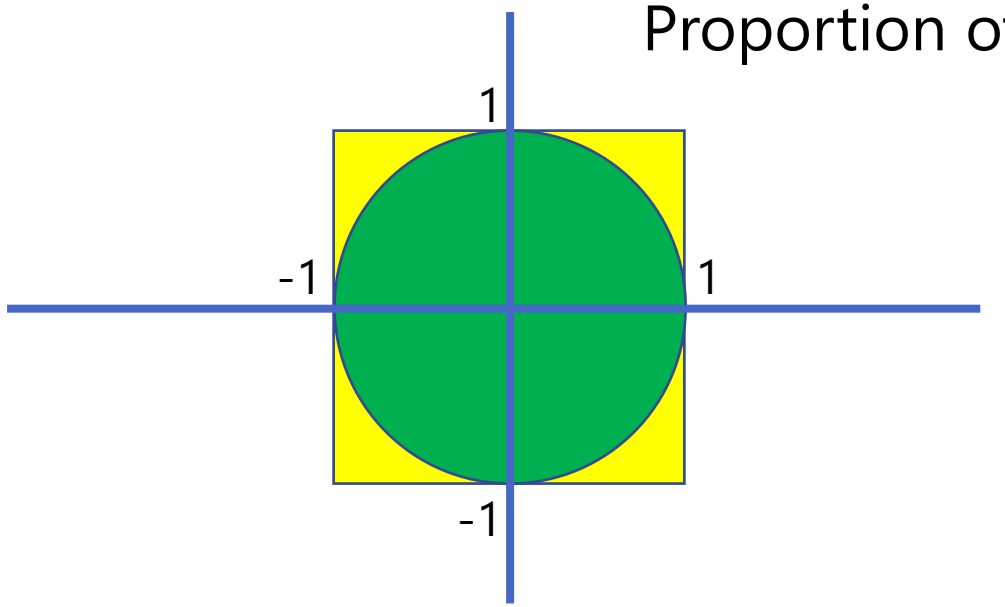
Exploratory, experimental, playful

Example: estimate π

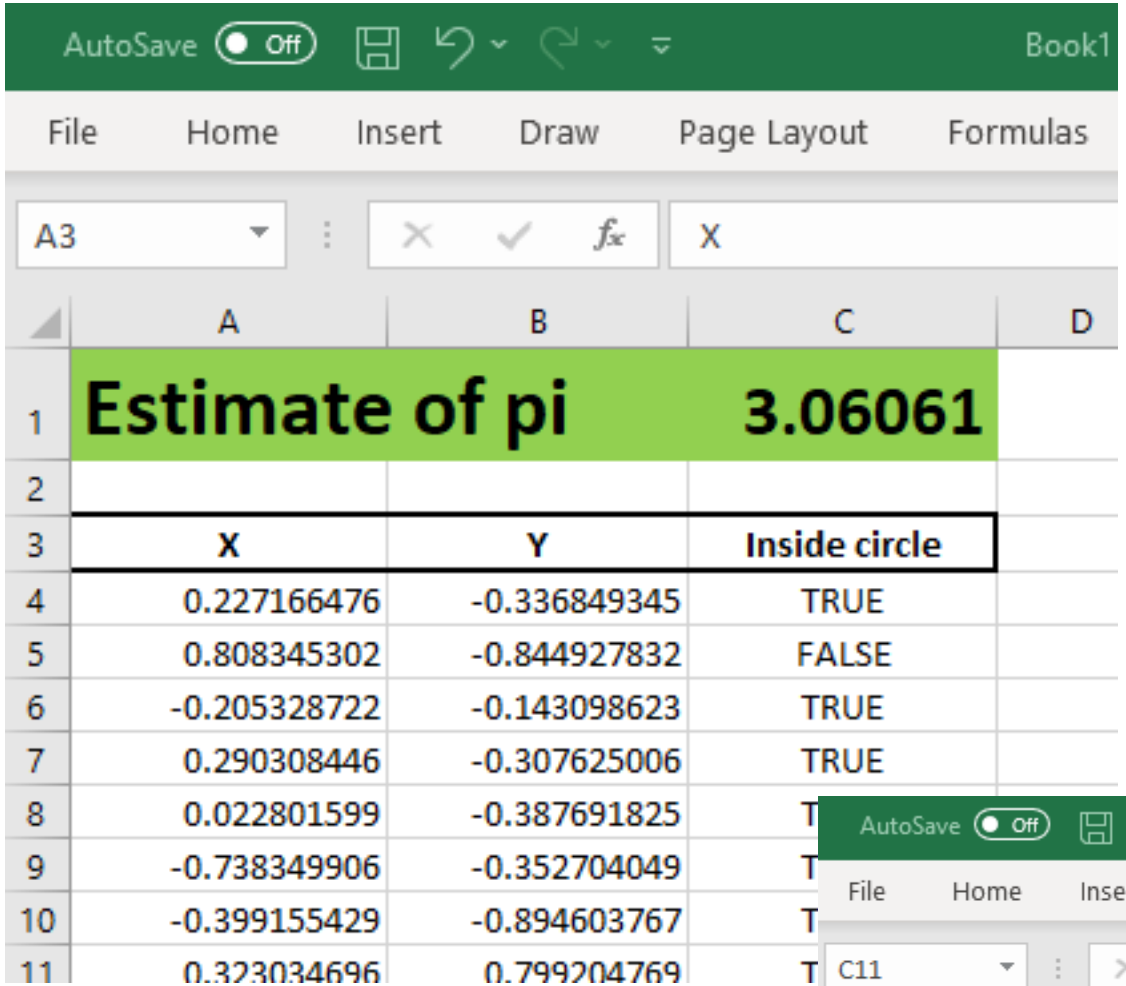
Area of yellow square $= 4r^2$

Area of green circle $= (a\ bit\ less\ than\ 4)r^2$

Proportion of darts inside $= \frac{\pi r^2}{4r^2} = \frac{\pi}{4}$



- Fire a random dart into the yellow rectangle.
- Count how many fall into the green circle (Pythagoras)
- Ratio is area-of-circle/area-of-square (areas)

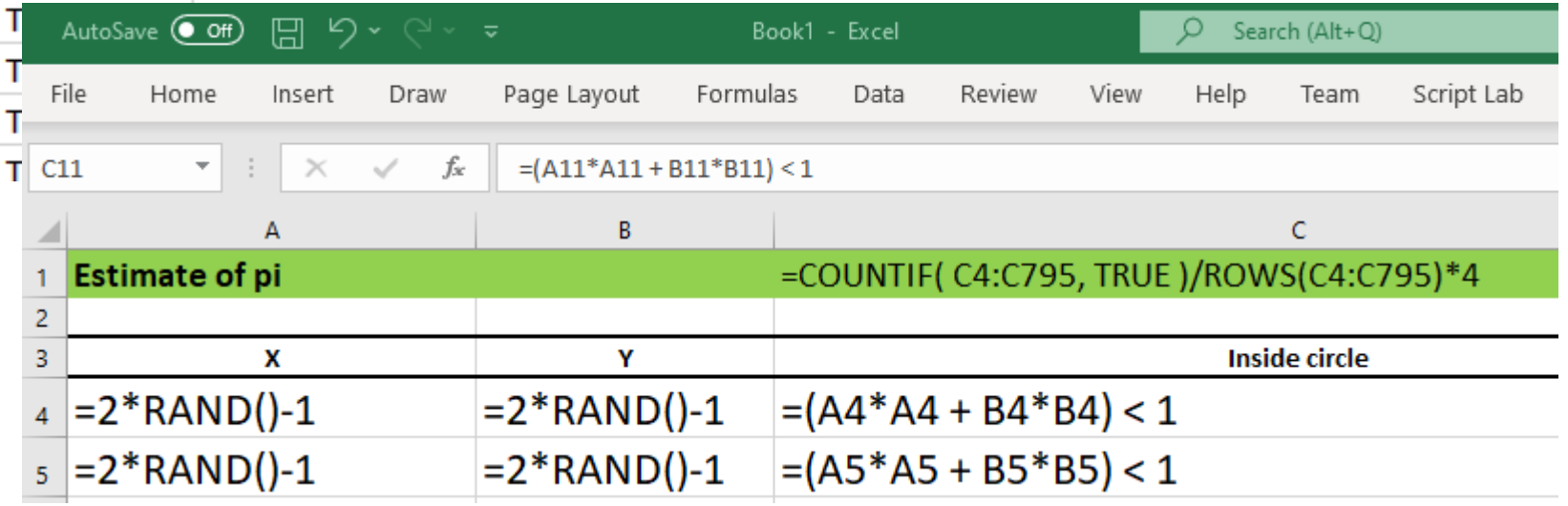


Ideas

- Transforming a random number in [0,1] into one in [-1,1]
- Deciding if (x,y) is inside the unit circle (NB: I omitted the SQRT)
- Do we need a full circle? Or just a quadrant? Discuss. Experiment?
- Booleans as a first class value
- How many experiments do you need to attain a given precision?

Skills

- Operator precedence
- Displaying formulas in cells
- Repeated recalc to re-sample
- COUNTIF

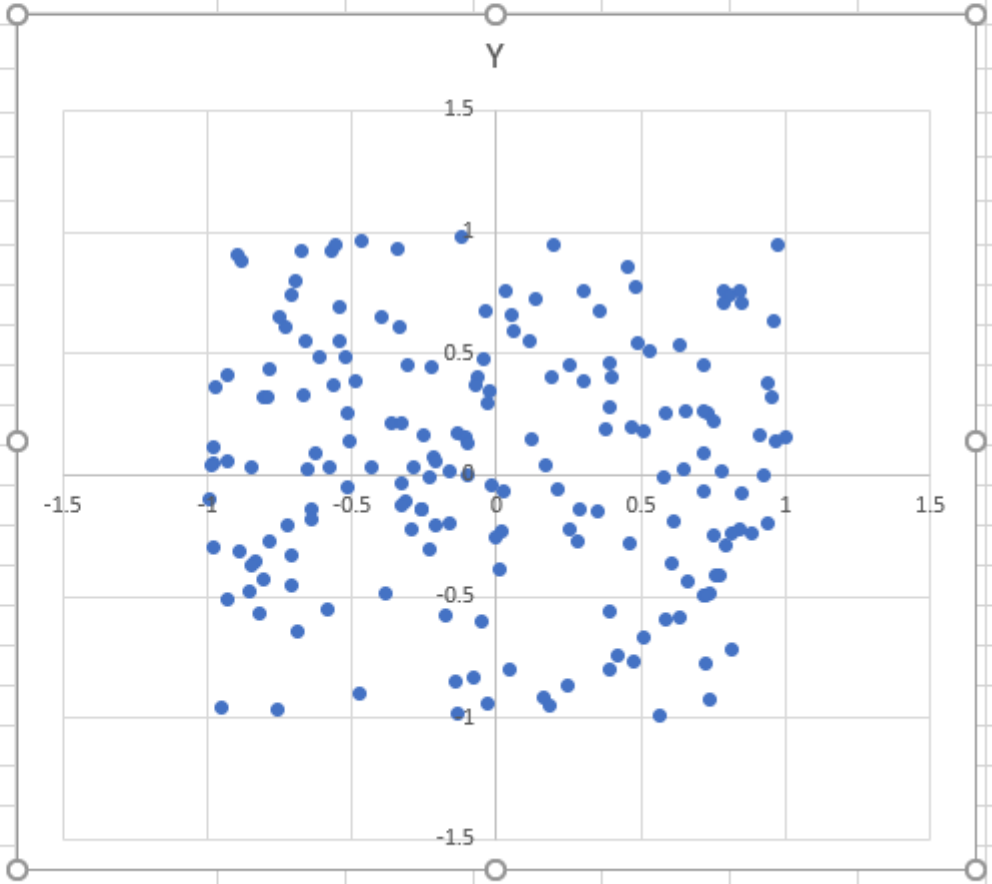


AutoSave Off | Book1 - Excel | Search (Alt+Q)

File Home Insert Draw Page Layout Formulas Data Review View Help Team Script Lab Chart Design

Chart 1

	A	B	C
1	**Estimate of pi**		3.166666667
2			
3	**X**	**Y**	**Inside circle**
4	0.64468672	0.0237929	TRUE
5	-0.832599556	-0.3591722	TRUE
6	-0.261560725	-0.1448313	TRUE
7	-0.016071834	-0.041579	TRUE
8	-0.260029915	-0.1421381	TRUE
9	0.38898371	-0.8043437	TRUE
10	-0.545162219	0.55384031	TRUE
11	0.839779688	-0.2270694	TRUE
12	-0.136525697	-0.984282	TRUE
13	-0.38549271	-0.4855768	TRUE
14	-0.474097402	-0.9012402	FALSE
15	0.465368115	0.19310976	TRUE
16	-0.284222401	0.02959265	TRUE
17	0.879545615	-0.2402651	TRUE
18	0.786568743	0.75513404	FALSE
19	0.846461629	-0.071022	TRUE
20	-0.029814631	0.29689125	TRUE
21	0.727557711	0.25152923	TRUE
22	-0.926634424	0.41148356	FALSE
23	0.803734491	0.7435791	FALSE
24	0.962096572	0.63346398	FALSE
25	0.752777779	-0.2469231	TRUE
26	0.027346749	-0.0683221	TRUE

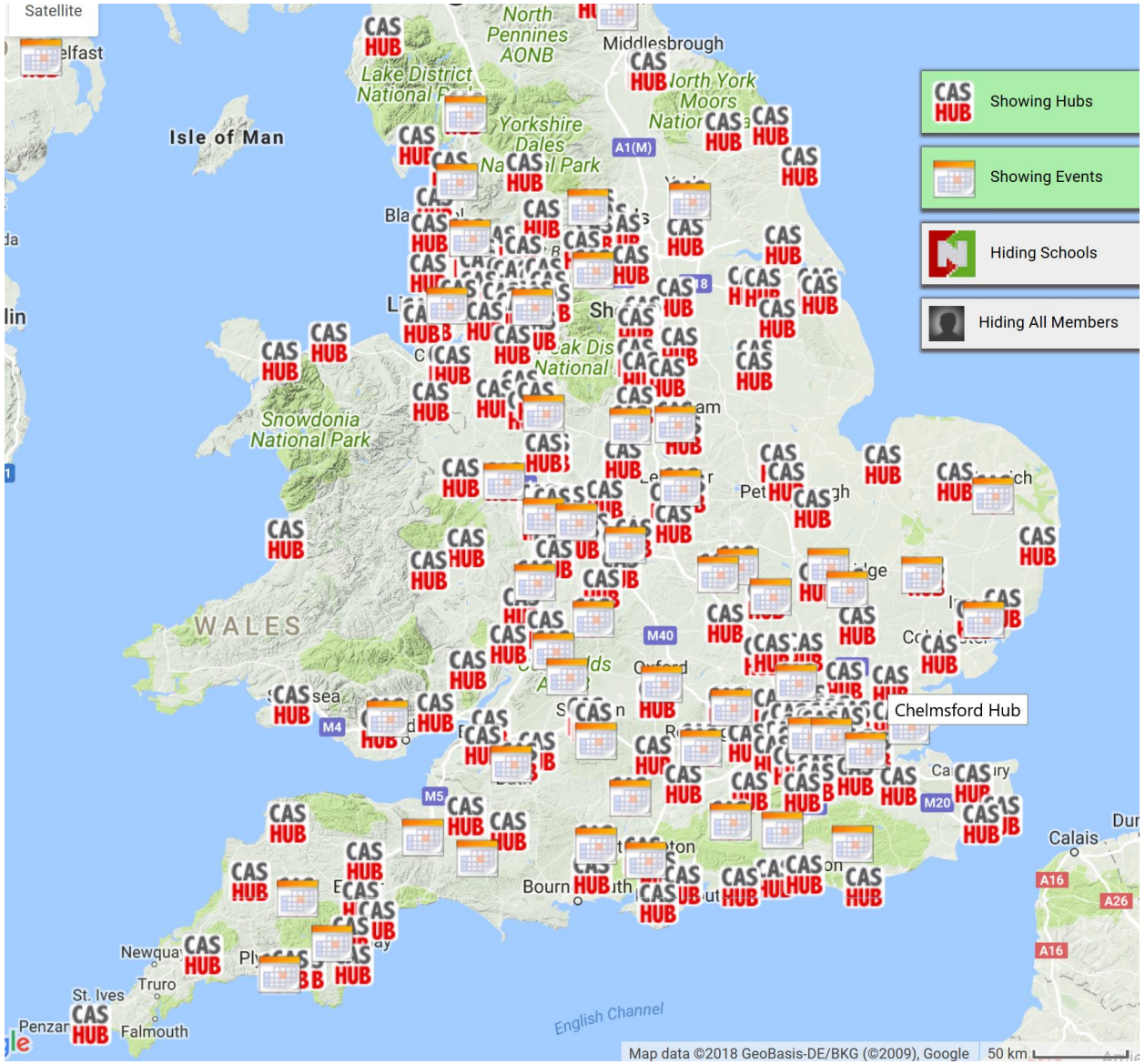


Lessons from the English experience

Lesson 1: Act together

- A **singular focus**: computer science as a foundational subject
- An **educational** message, not just an **vocational** one
- A **single voice**, not competing special interests
- An **independent, grass-roots group**, not an employers group, not a teachers group, not a higher-ed group
- Support from **professional bodies** (eg Royal Soc): influences civil servants
- Support from **industry leaders** (eg Eric Schmidt speech): influences politicians
- Don't wait for policy change: just get on with it

Lesson 2: build a community of practice



COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE
Part of BCS, The Chartered Institute for IT

- 30,000 CAS members (worldwide)
- 350 CAS Local Communities
- 5000+ online resources

Lesson 3: get high-stakes qualifications right

- High-stakes national examinations
  - Drive school behaviour (league tables)
  - Drive teacher behaviour (prepare students for exams)
  - Drive pupil behaviour (pass exams)
  - Drive parent behaviour
- Well designed exams embody the aspirations of the curriculum – badly designed ones kill those aspirations at birth
- Getting this “right” is hard, because the stakes are so high
- Need pathways for different cohorts
  - Computer science, for the geeks of the future
  - Applied computing (websites, robotics, data science, IT…), for those who will use computing to do something else
  - Digital competencies, for everyone

Lesson 4: Loving our teachers

- Teachers are our heroes and heroines. We should love and support them, not make them feel ignored or overwhelmed.
- One engaged, well-equipped, passionate teacher can inspire thousands of students
- **Being a teacher is hard. Being a computing teacher is harder**
  - Everything is in flux
  - Making CS a mainstream subject, including from early years, has consequences right across the educational system
  - Pedagogy is in the early stages
  - Qualifications are in constant change
  - Not enough training and hard to get to it
  - “Its me, me, or ……. me”

Lesson 5: spend a tiny fraction on research

- Other subjects have *centuries* of experience; we have 5 years.
- This needs research, and evidence.
- Current practice is "gut feel". This is stupid. We should study what works and then do that.
- It is bonkers to spend £1bn on teacher salaries and £0 on research that would make them 20% more productive

What's the Best Way to Teach Computer Science to Beginners?

Mark Guzdial questions the practice of teaching programming to new CS students by having them practice programming largely on their own.

Mark Guzdial
"How We Teach Introductory Computer Science is Wrong"
http://bit.ly/1qnv6gy
October 8, 2009

studies with similar set-ups. There ar
two groups of students, each of whic
is shown two worked-out algebra prol
lems. Our experimental group then ge
eight more algebra problems, complet
ly worked out. Our control group solv

Summary

Our opportunity and challenge: 1+1 = 3

In Portugal, can we introduce computing into the curriculum…

- **Establish computing (including computer science) as a foundational subject** that every child learns, as they do maths and natural science.
- **Treat computing not as a competitive upstart, but rather as a distinctive new "lens" on maths education**
  - that lets us see it in a new light
  - and perhaps leads to different approaches, content, emphasis;

  Not up-ending maths education, but helping it find its true identity

Engaged, curious

Empowered, informed

Creative, playful

Employed