Computing and Professional Development for Teachers in the UK & Unplugged Session

Dr Sue Sentance
National Academic Coordinator, Computing At School

6th June 2014
... started in 2008 with 5 members

... in 2014 almost at 12,000 members ...

There is no THEM, only US!
How CAS Started

Something is wrong – but I feel powerless

Something is wrong – but I feel powerless
How CAS started

If we get together, perhaps we can get something done.
CAS Membership

CAS Online Membership

Members

Date

Jan '09 Jul '09 Jan '10 Jul '10 Jan '11 Jul '11 Jan '12 Jul '12 Jan '13 Jul '13 Jan '14 Jul '14
Simply a group of individuals, concerned about the state of computing education in our schools

Including:
- Teachers
- Industry (eg. Google, Microsoft)
- University academics (incl. CPHC, UKCRC)
- Members of exam board (eg. AQA)
- Members of professional societies (eg. BCS)
- Parents
- Local educational advisers
- Teacher trainers

Varied backgrounds, with common concerns
Curriculum change – the story

2008 - 2011

Computing AT School
Educate · Engage · Encourage
In collaboration with BCS, The Chartered Institute for IT

OCR
Oxford Cambridge and RSA
Curriculum change – the story

2012 - 2014

Computer science part of English Baccalaureate
By Sean Coughlan
BBC News education correspondent

4 February 2013 Last updated at 19:33 GMT

Statutory guidance
National curriculum in England: computing programmes of study

Organisation: Department for Education
Page history: Published 11 September 2013
Policy: Reforming qualifications and the curriculum to better prepare pupils for life after school
Applies to: England
Collections: National curriculum

Network of Excellence
Computer Science Teaching

Computing at School
Educate · Engage · Encourage
Across the UK

United Kingdom

Scotland

N.I.

England

Wales

Already had Computer Studies in the curriculum at secondary school

New curriculum in Computing in September 2014

New 14-16 qualifications in Computer Science
Computer science is a proper, rigorous school subject discipline, on a par with mathematics or chemistry, that every child should learn from primary school onwards.

NB. It is not about computers, nor is it about programming
Curriculum change (in England)

Department for Education

- Computer Science at heart of new curriculum
- Schoolchildren from age 5 to be taught computer science
- Still includes digital literacy as major component
- September 2013 Final Programme of Study for Computing published
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
Key stage 1

Pupils should be taught to:

- understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions
- create and debug simple programs
- use logical reasoning to predict the behaviour of simple programs
- use technology purposefully to create, organise, store, manipulate and retrieve digital content
- recognise common uses of information technology beyond school
- use technology safely and respectfully, keeping personal information private; identify where to go for help and support when they have concerns about content or contact on the internet or other online technologies.
Content of curriculum: KS2 (7-11)

Key stage 2

Pupils should be taught to:

- design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts
- use sequence, selection, and repetition in programs; work with variables and various forms of input and output
- use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs
- understand computer networks including the internet; how they can provide multiple services, such as the world wide web; and the opportunities they offer for communication and collaboration
- use search technologies effectively, appreciate how results are selected and ranked,
Content of curriculum: KS3 (11-14)

Key stage 3

- use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions

- understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; understand how numbers can be represented in binary, and be able to carry out simple operations on binary numbers [for example, uses in circuits and programming; understand how numbers can be represented in]

- understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits
4 The following is a function:

```plaintext
FUNCTION IsPrefix (name, value)
    IF name[1] = value
        RETURN true
    ELSE
        RETURN false
    ENDIF
ENDFUNCTION
```

4 (a)(i) Give the name of one parameter used by this function.

..........................................................................................................................(1 mark)

4 (a)(ii) State the data type of name.

..........................................................................................................................(1 mark)

4 (a)(iii) State the data type of the return value.

..........................................................................................................................(1 mark)

4 (b) Give three reasons why programmers use functions.

..........................................................................................................................
16-18: Typical A-Level material

Work at this level includes programming, databases, networking, computer architecture, etc. including a substantial software development project.

The unsolvable problem of determining whether any program will eventually stop given particular input.

The abstract model of the Turing Machine and the Universal Machine.
### Uptake of Computer Science in school

<table>
<thead>
<tr>
<th></th>
<th>2012</th>
<th>2013</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1766</td>
<td>4253</td>
</tr>
</tbody>
</table>

- Computing Programme of study in England becomes statutory from 5-14 for state-maintained schools in September.
- GCSE Computer Science now offered by 5 awarding bodies in England (first results were in 2012). Entries rose from 1766 in 2012 to 4253 in 2013. Take up in 2014 – 14,000 estimate (one awarding body only).
- More schools showing interest in A-Level Computing since start of GCSE, with increase at AS-level of 15%.
CAS for Teachers

CAS Community Website
- Resources
- Discussion
- Events

Networking/training opportunities
- CAS Hubs
- CAS Master Teachers
- CAS Conference(s)
Home page – who is near you?

Location

Nearest hub: Cambridge, 8 km (5 miles)

Nearest Master Teacher: Peter Gaynord, 13 km (9 miles)

Nearest Lead School: Woodingdean Primary School, 146 km (91 miles)
## Resources on CAS – shared by teachers

### Filters (AND)
- English Curriculum
- Scottish Curriculum
- Language/Platform
- Resource Type
- Uncategorised

### Resources

<table>
<thead>
<tr>
<th>Resource Title</th>
<th>Description</th>
<th>Created by</th>
<th>Last Modified by</th>
<th>Date Created</th>
<th>Date Last Modified</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary computing keywords posters</td>
<td>Key words from the 2014 computing curriculum for KS1 and KS2 in pupil speak.</td>
<td>Pete Dring</td>
<td>Pete Dring</td>
<td>Feb 08 2014</td>
<td>Feb 08 2014</td>
<td>19</td>
</tr>
<tr>
<td>Progression Pathways Assessment Framework KS1 (Y1) to KS3 (Y9)</td>
<td>The purpose of the Progression Pathways Assessment Framework is to support teachers in assessing their pupils’ progress in computing from Key Stage 1 (Year 1) through to Key Stage 3 (Year 9).</td>
<td>Mark Dorling</td>
<td>Mark Dorling</td>
<td>Jan 21 2014</td>
<td>Jan 21 2014</td>
<td>32</td>
</tr>
<tr>
<td>OCR A451 GCSE Computing Theory and Homework Booklet</td>
<td>OCR Computing Theory and Homework Booklet</td>
<td>Tamsin Laber</td>
<td>Tamsin Laber</td>
<td>Jul 03 2013</td>
<td>Jul 03 2013</td>
<td>54</td>
</tr>
<tr>
<td>OCR GCSE Computing: An Unofficial Teacher’s Guide</td>
<td>An unofficial (but OCR endorsed) collection of notes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Resources on CAS – shared by teachers

Teach Programming with Python (from Year 7)

last edited May 18 2013 by Alan O'Donohoe

Created by Alan O'Donohoe. Other contributors:

This resource is public; the short and full description below are publicly viewable, as are the files (but not the comments)

Short description:

A series of lesson plans for teaching Python to children as young as 11

Full description:

This is a guide to teaching a series of lessons to pupils from Year 7 upwards. The materials have been tried and tested with classes and evaluated by a number of teachers including non-specialists and Python developers.

The lessons start with no previous experience of Python and then build up to creating a text-based game with some basic artificial intelligence. A second unit is currently being developed to follow on, for teaching from Year 8 with progression.

For some background, read my blogpost about teaching Python to Year 7 and this
jQuery - zero to hero

last edited Aug 16 2012 by Laura Dixon

Created by Laura Dixon. Other contributors:

Short description:
A booklet taking you through from zero HTML to jQuery ninja (sort of)

Full description:
A booklet covering the basics of HTML, CSS, JavaScript and jQuery. Contains a tutorial for making a sliding box and a Mr T quote generator.
Resources on CAS – shared by teachers
<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Time</th>
<th>Organiser</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb 20</td>
<td>Preparing to teach GCSE Computing 2 day event</td>
<td>9:00AM until 4:00PM</td>
<td>Darren Travi</td>
<td>Buckinghamshire</td>
</tr>
<tr>
<td>Feb 21</td>
<td>Python Training Course</td>
<td>10:00AM to 3:00PM</td>
<td>Graham Bradshaw</td>
<td></td>
</tr>
<tr>
<td>Feb 25</td>
<td>Introducing the World Wide Web for KS2</td>
<td>1:00PM to 4:00PM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feb 25</td>
<td>Fun with Algorithms - KS1 (Years 1 and 2), Lower KS2 (Years 3 and 4),</td>
<td>4:00PM to 6:00PM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Upper KS2 (Years 5 and 6)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feb 25</td>
<td>Demystifying the Primary Computing Curriculum</td>
<td>5:45PM to 7:45PM</td>
<td>Trevor Bragg</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Demystifying the Primary Computing Curriculum for KS 1 &amp; 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Setting up and getting started with the Raspberry Pi</td>
<td>9:00AM to 12:00PM</td>
<td>Christian Turton</td>
<td>Luton and Beds</td>
</tr>
<tr>
<td></td>
<td>KS1 &amp; KS2: An Introduction to Computing</td>
<td>9:00AM to 1:00PM</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Discussions

Subscribed General Discussion Forums

**General**
General discussions about computing at school
105 topics with recent unread posts  (Mark all read)

Last post on *How can you possibly teach programming in schools?*  
by Kevin Wright, about 9 hours ago

**Secondary Education**
Discussions related to secondary-level education
48 topics with recent unread posts  (Mark all read)

Last post on *Over SUMing*  
by Peter Donaldson, about 8 hours ago

**The Network of Excellence**
Discussion forum for all those involved with the new CPD project for CAS 'The Network of Excellence'
1 topics with recent unread posts  (Mark all read)

Last post on *CAS Stamford Lincolnshire Hub First meeting*  
by Radha Rajput, 05 Feb 2014

Subscribed Special Interest Discussion Forums

**Computer science education research in schools**
Computer science education research in schools
1 topics with recent unread posts  (Mark all read)

Last post on *Interesting reading from recent Future Directions in Computing Education summit*  
by Sue Sentance, 09 Feb 2014

**Site Issues**
A place to post issues or questions related to the CAS Community website itself
4 topics with recent unread posts  (Mark all read)

Last post on *Ordering of entries in the Hubs list*  
by Neil Brown, 1 day ago

EDUCATE • ENGAGE • ENCOURAGE
A national programme of professional development for teachers of Computer Science
How does the Network of Excellence work?

- Through the Network of Excellence we will train 400 Master teachers who will offer training (at a low cost) to other teachers in their local area.

- **Master teachers** are experienced teachers who receive funding for some release from school in order to support and train other teachers in their area.

- The Network of Excellence also includes universities who are committed to supporting schools and teachers in developing subject knowledge.

- Schools which wish to be leaders in their area for the teaching of Computing can apply to be **Lead Schools**.
Becoming a master teacher

If you can answer 'YES' to the following then the Master teacher programme is for you!

- Are you a primary or secondary teacher in a state maintained school?
- Would you like funding to develop your skills and knowledge of computing in the new National Curriculum?
- Are you an experienced teacher with good or outstanding teaching seen in your recent lesson observations?
- Can you confidently engage with your peers in a professional environment?
- Do you have the ability to design and deliver practical and interactive workshops for teachers with appropriate course material?
- Do you have a passion about sharing best practice in teaching and learning?
Master teachers

- Run low-cost training events for teachers
- Speak at events
- Visit teachers in schools
- Active members of CAS Community
Local, face-to-face, peer-to-peer support
Program your teacher to make a Jam Sandwich (Sandwich Bot)…

Philip Bagge - 11 videos
Supporting teachers in the NoE

Universities <-> Master Teachers <-> Hubs

- Work together
- Provide:
  - Training courses
  - Support
  - Networking
- Can access:
  - Teachers
... started in 2008 with 5 members

... in 2014 almost at 10,000 members ...

There is no THEM, only US!
Summary

Computing At School started as a vehicle for advocacy to promote CS in school.

CAS is a true “community of practice”

Curriculum change has happened (in England) and growth is very rapid!

CAS : “There is no THEM, only US”

Our professional development model (NoE) is holistic – a local, peer-to-peer, face-to-face model... empowering enthusiastic teachers
Questions?
Section 2
Teaching Computing with unplugged-style activities
Why unplugged?

- We don’t always have access to computers
- We want to encourage cooperation and social skills
- Principles do not require computer hardware
- Unplugged is fun!
5 Activities for you to try

1. Cup Robotics
2. The Imp Computer
3. How the Computer Works
4. Painting by numbers
5. Emotional Robot

☐ For each:
   ▶ What are the informatics learning points?
   ▶ Could I use this in my class?
Groupings

1. Cup Robotics
   - We will work in groups of 5 with one robot in a group

2. The Imp Computer
   - I will need 7 keen volunteers!

3. How the Computer Works
   - We will do this activity in groups of 3

4. Painting by numbers
   - Divide the room into two halves

5. Emotional Robot
   - I will need 6 keen volunteers!
Activity 1: Cup Robotics

Divide into groups of 5 (ish) - decide who will be the robot.
Cup Robotics

- You will instruct a cup building robot to build a stack of cups. You may only use these predefined instructions.
- One person will step aside and be the robot, the others will write a simple program to make a stack.
An example

☐ **Rules**

☐ When a cup is picked up it is automatically risen above the highest cup.

☐ One step (forwards or backwards) = ½ cup width.
An Example
Activity 2: The IMP Computer

7 volunteers please!
Aims

• Give you deeper understanding of core topics
  – Getting started programming
  – If statements and flow of control
  – Compilation, compile-time and run-time
• Give you practical ways to teach computing in a fun, thought provoking way
  – away from computers, focus on concepts
• Linked activity sheets and booklets can be downloaded from our website:
  www.teachinglondoncomputing.org
Running programs physically

- A really good way to build a deep understanding of programming constructs is to act out the program
- Compile the program on to people!
- They follow the instructions
- Makes abstract ideas visible and tangible
An Insulting Program

answer = input ("Can I insult you?")

if answer == "Y":
    print ("You smell!")
else:
    print ("You smell of roses!")

print ("Thank you!")

We can compile this on to a toilet roll computer
Compiling it on to Imps

• Each node is a student
  – connected by rope.

• You pass in a tube at one end
  – You are the operating system

• When it is handed back something should be printed on the screen
  – written on the board
What it compiles to …

```python
answer = input("Can I insult you?"
If answer == "Y"
    Print "You smell!"
ElseIf answer == "N"
    Print "Thank you"
Else
    Print "You smell of roses!"
```
Explaining programming concepts

• We’ve made an invisible program tangible
• Can now explain all sorts of concepts
  – If statements
  – Control structures
  – The program counter
  – Run-time versus compile time
• Similar approach for loops
More advanced concepts

• Can also explore more advanced ideas
  – How program changes change structure
  – Optimizing compilers
  – Bugs
A Snap Program

card1 = input ("Next card")
card2 = input ("Next card")

if card1 == "RED" and card2 == "RED":
    print ("SNAP!")
else:
    if card1 == "BLACK" and card2 == "BLACK":
        print ("SNAP!")

Print ("Bye")

We can compile this on to a toilet roll computer
What it compiles to ...

- `card1 = input "Next card"
- `card2 = input "Next card"
- `card1 == "RED" & card2 == "RED"?
- `card1 == "BLK" & card2 == "BLK"?
- Print "SNAP!"
- Print "Bye"
- Print "SNAP!"
Exploring Different Programs

```python
if card1 == "RED":
    if card2 == "RED":
        print ("SNAP!")
else:
    if card2 == BLACK:
        print ("SNAP!")
print ("Bye")
```

This will compile this into a different toilet roll computer
What it compiles to …

1.

- card1 == "RED"?
  - True
    - Print "SNAP!"
    - Print "Bye"
  - False
    - card2 == "BLK"?
      - True
        - Print "SNAP!"
      - False
        - card2 == "RED"?
          - True
            - Print "SNAP!"
          - False
            - Print "Bye"
• The compiler could optimize the first program
  – to more compact code, fewer imps needed!

```python
print "SNAP!"
print "Bye"
```
Summary

Programming can be introduced in fun ways away from a computer

• By compiling the program onto students
  – You make it visible and tangible
  – Open it to questions
  – And exploration
More support

On our website to support this session:
• Activity sheets
• Story sheets
• Slides

Details of more workshops/courses
• free unplugged sessions
• subsidised courses (e.g. GCSE programming)

www.teachinglondoncomputing.org
Twitter: @TeachingLDNComp
Activity 3:
How the Computer Works
(very simplified!)

(Groups of 3)
How Computers Work

Move into groups of 3

- One person is the display
- One person is the memory (and ALU)
- One person is the central processing unit (CPU)

The CPU gives instructions to the other components
Activity 4: Painting by Numbers

Divide the room into two halves. Each half must not be able to see the other half’s work.
How does a computer store an image?

- Any ideas...?
  
- A computer represents an image using numbers
  - How could it do this?

- Computer screens are divided into pixels
  - Pixels are small dots that make up an image
  - Stored images are shown to us using these pixels
  - Sometimes if you enlarge an image you can see the different pixels
Digitising the image

- Look at the image you have been given
  - it is a smooth line drawing - hand-drawn? real-life scene?

- We will now *digitise* it...
  - turn it into 1’s and 0’s (1 is black and 0 is white)

- Cover image with the 8x8 grid of LARGE boxes
- If a part of the image shows under a box, write 1 in the box
  - otherwise write a 0 in the box

- Pass completed grid to someone on the other side
  - HIDE your original drawing!!
Rendering the digitised image...

- Place your remaining blank grid of large boxes exactly over the numbered grid your partner gave you

- Fill in each square if there is a 1 underneath
  - Fill it completely – use thick pen if you have it

- Can you identify what the image is?
  - Show your picture to others in your group
  - Don't tell the other side what your picture is, even if they guess correctly
Now go through the process again

- But this time, use the 16x16 grid

- Can you see what the picture is more clearly this time?

- Why?
Now you know about black + white...

• How do you think colour is represented?

• What will this do to the size of the image?

• How large is the image file for a picture of size 100x100 which uses 256 colours?

• How do you think a movie is represented?
Image Representation

- **Digitisation**
  - Light/colour levels are turned into numbers
  - Information can be lost at this stage

- **Storage/Transmission**
  - Digital images are stored/transmitted in files of these numbers

- **Rendering**
  - Numbers are converted into coloured pixels
  - Applies to screens, printers etc.
Activity 5: The Emotional Robot

I need 6 volunteers who will be:
2 eyes
2 eyebrows
1 left mouth
1 right mouth
## Evaluation

<table>
<thead>
<tr>
<th>Activity</th>
<th>Learning Points</th>
<th>Useful for?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cup Robotics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The IMP Computer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>How a computer works</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Binary Baubles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The Emotional Robot</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Activities are taken from:

- [www.teachinglondoncomputing.org](http://www.teachinglondoncomputing.org)
- [www.code.org](http://www.code.org)
- [www.csek12.org](http://www.csek12.org)