Computing in the Curriculum

The Word is Out

Eric Schmidt of Google’s address to the MacTaggart lecture in August finally brought into public view many of the points CAS members have been making of late. He was ‘flabbergasted to learn that today computer science isn’t even taught as standard in UK schools’ and made the point that to teach the use of software rather than insight into how it was made was throwing away our great computing heritage.

Behind the big media article, the word is out. Concern is widespread among the IT professional community who see what their children are being taught – just read the letters page of any of the main IT professional magazines. They are ready to help CAS, they just need to be shown how.

“Our IT curriculum focuses on teaching how to use software, but gives no insight into how it is made. That is just throwing away your great computing heritage.”
Eric Schmidt
Google Chairman

Our views are increasingly being embraced by others. The lobbying work Simon Peyton-Jones and others are engaged in with BCS, The Chartered Institute for IT (see p6) is getting us listened to in government with encouraging signs.

This has not been achieved with just big steps in the press and corridors of power, the little steps are important as well. Every teacher introducing a new project has an impact. Every school governor that is helped to understand the issues, helps us. The work behind the scenes by CAS members is directly driving this change.

Let’s keep up the good work – because it is starting to work!

Teacher Conference Keeps on GROWING

The 3rd CAS annual conference on 24 June was a huge success with a record number of delegates attending. Over 250 delegates attended the event, kindly hosted by the University of Birmingham, School of Computing.

Keynote presentations were given by leading figures in the field of Computing including:
• Karen Brennan from MIT Media Lab, the institution that created the SCRATCH programming language for 8 - 14 year olds,
• Professor Steve Furber, Chair of the Royal Society Advisory panel and designer of the BBC micro
• Sir Tony Hoare, inventor of the Quicksort algorithm
• Doug Aberdeen, Google Zurich, gave examples how to demonstrate technology in the classroom

In addition the 21 practical workshops, 4 seminars, 5 Forums and a session for trainee teachers made it a packed day. One teacher commented, “With the plenaries, workshops and lectures, I was totally blown away!” Delegate feedback overall was very positive. “New ideas for teaching, inspirational speakers, so many fantastic ideas!” being typical of the positive comments received. Make sure you don’t miss next year! Claire Davenport

The “Computing At School” working group (CAS) is a membership association in partnership with BCS, The Chartered Institute for IT and supported by Microsoft, Google and others. It aims to support and promote the teaching of computing in UK schools.
TWO COMPETITIONS TO CELEBRATE ALAN TURING

The 4th Annual UK Schools Computer Animation Competition received almost 900 entries from over 150 schools. The 45 lucky winners received their prizes on 1 July, at the Animation11 Film Festival and Inspirational Computer Science Day, at The University of Manchester. Over 350 schoolchildren, teachers and parents attended, with afternoon activities including a keynote talk on Animation in Games from Fred Gill of Electronic Arts, a presentation about stereo vision, and two rooms of hands-on workshop activities. A link to the winning entries and video of the day are in the supplement.

"Animation12 launches in September, with an entry deadline of April 2012. New for the 2012 competition will be a larger set of animation programs, and a special Alan Turing Centenary Year strand to celebrate his life and work. We are also pleased to be hosting CAS’s Code-Breaker competition, a programming challenge based on Greenfoot to celebrate the Turing Centenary. Further details of both competitions on the website. The UK Schools Computer Animation Competition is generously funded by Electronic Arts, Google, NESTA, and The Granada Foundation.

Toby Howard"

PRIMARY SCHOOL ‘LIGHT HARP’ USING ARDUINO AND SCRATCH

Pupils at St Mary’s CE School, Albrighton had used Scratch before to make things happen on-screen. Headteacher, John Rowe wanted to make the connection that they could make things happen beyond the computer.

As John says; “Programming can be as creative an activity as anything else, even at primary level”. Scratch for Arduino (S4A) enables an Arduino microcontroller to act as an interface board to connect Scratch to external sensors and actuators. The native programming environment is based on the Processing language, but learning a subset of C++ is rather beyond the scope of a primary school project. The new-found S4A (see supplement) makes things easy to set up and puts the use of Arduino within the reach of those likely to be using Scratch with children.

The children at St Mary’s quickly learned to program a sprite to move on-screen. After a very short time they had incorporated push switches mounted on the Arduino boards. We wanted the children to make something that worked in the real world, using the microcontroller as an interface for sensing. A Pico board (see below left) would have been ideal for this activity, but John was thinking of future activities and wanted to have a board that offered output as well as reading inputs. Scratch gave them the means to listen to the microcontroller and read the voltage given on the pins connected to a light-dependent resistor (LDR); when the LDR was put into darkness the voltage dropped. Those of us of a certain age may remember some awe and wonder at first seeing Jean Michel Jarre’s Laser Harp in the 1980’s. It has the same effect on children in 2011 - after watching a video of J-MJ they all wanted to make one! So they set out a project plan - find out how to make a note play, wire up the LDRs to the Arduino and make Scratch read the voltage as a trigger for the note. Scratch can only read six pins from the Arduino, but a quick lesson on pentatonic scales and traditional Chinese music set the parameters for the final task - design and make a light harp to play some sheet music that I had brought along, courtesy of Shropshire’s music adviser.

Once pupils knew the Scratch number for a note they worked out their own system for pitch. Chat levels were high as they excitedly passed new ideas between groups. They solved various physical problems, including varying light levels in the room and not enough differential between ambient light and darkness (using a tube over the LDR to make light come in only one direction). Oh yes, also make sure the leads connecting the LDR to the breadboard are long enough! Alongside the discussion of programming, physical problem-solving made them stop, think and negotiate a solution. This was real world learning.

Dale Jones

Scratch SENSOR BOARDS

Karen Brennan, from the MIT Media Lab, demonstrated some simple ideas using external sensors in Scratch projects at the CAS Teacher Conference. Initially, Scratch sold their own Scratchboard via their website but now suggest using the similar Pico Board which is available from various educational suppliers. These have built in light and sound sensors, a slider and button plus 4 additional connections to which various sensors can be attached. They connect to the pc via usb. A getting started guide gives several projects possible in primary schools and there is a wealth of ideas available via the Scratch forums. Links to all are in the supplement.
STUDENTS ENJOY AWE INSPIRING TIME AT YOUNG REWIRED STATE

The idea of YRS is simple; firstly get together a bunch of young programmers, an enthusiastic mentor, a welcoming centre and some government open data. Then, let the ingredients do their magic for 4 days. The result, 43 open government data apps.

I had no idea what to expect when I signed up, I had previously studied computing at school but this promised something very different. Where the focus in school had often been on organised and well maintained code, Young Rewired State certainly emphasised the need sometimes for the quick and dirty. Living in Kendal, my nearest centre was Madlab, Manchester (a hackspace for the digital community).

Quickly from day one it became apparent that there were a wide range of people taking part in these events, some veterans of past YRS events, some people with an experience programming but not in this sense and some people just eager to learn. It should also be noted that there were people from as young as 10 all the way up to the upper age limit of 18. This may seem daunting but the best thing about the event was how well everyone worked together so that everyone was involved. Some of the projects created throughout the week were awe-inspiring from the more trivial to the creation of a social network based around the books in Madlab ("Who says you can't build a social network overnight!").

At the end of the week each centre travelled down to the Microsoft’s headquarters in London to present their ideas. From absolute novices to web development experts Young Rewired State is a great opportunity if you want to get started coding and to make some great friends. Oh yes, there is also always a copious amount of pizza. Matthew Pickering

FOLK DANCING AND FAIRY TALES: NOVEL WAYS TO TEACH CONCEPTS

Algorithms and data structures can be difficult concepts for students to grasp. This is particularly true when they struggle with the complexities of implementing them in an alien programming language. As Paul Curzon demonstrates in Computing Without Computers (see column right), gaining a conceptual understanding is key. Children learn in many different ways so providing a variety of approaches can help get the penny to drop. At Elon University, North Carolina, Shannon Duvall has written a set of simple fairy tales to illustrate concepts such as loops, variable scope and algorithmic complexity. She uses them in introductory classes ‘kindergarten style’. In Romania, Algorhythmics is an initiative from Sapientia University that has set common sorting algorithms to traditional folk dances. Six sorting algorithms, including the Quicksort, are currently available on their YouTube channel. They are a marvellous tool in class to help visualise and reinforce the complex sequences.

INTROCUDING COMPUTING WITHOUT COMPUTERS

Getting to grips with programming can be a daunting task. One of the great strengths of resources like Scratch, is that they allow students to visually construct logical sequences without worrying about syntax. However, even the best visual language is of little use if the student hasn’t grasped the underlying constructs and related concepts. Paul Curzon, a key inspiration behind CS4FN tackles these in ‘Computing Without Computers’.

Available for free download it is a teachers dream. All the main concepts we tend to cover when introducing programming are covered. The approach is to use many varied analogies with which your pupils will be familiar. By using everyday examples, he makes computing concepts accessible to all. If you have ever struggled to explain an idea, this book will provide a treasure trove of exemplars. Moreover, Paul recognises that pupils learn best by doing. Interspersed with the explanations are some wonderful puzzles, often supported by material on the CS4FN website, that help reinforce the ideas he explains. This excellent resource can be applied to teaching programming at any level. Roger Davies

GOOGLE CODE-IN 2011

Google Code-In is aimed at getting 13-18 year old students involved in the development of open source software. The contest lets students compete for tasks like translations, documentation, research and coding. They work on projects for organizations who mentor them and help them through the tasks. All students receive a t-shirt, and all can compete for the grand prize: an all-expenses-paid trip to Google’s headquarters in California. Code-In 2011 begins the week of November 21.
BUILDING STAFF CONFIDENCE THROUGH A COMPUTING CLUB

I am not afraid to say I was nervous at the prospect of setting up a Computing Club, worried that the students would find out that my computing knowledge was worse than theirs. This wasn’t helped by a very low turn out of students the first week, but I started with the basics and we learnt how to use GameMaker. I was surprised by how easy it was to use, and even more surprised when the students coped with it so well. The in-built tutorial is a fool-proof way to start and the speed at which students can see working results really got them interested. Numbers grew week by week and our competency grew together. The students enjoyed teaching me things they found and loved that we were learning together on a level playing field. My fear of all things ‘computing’ left and I realised that I was supporting them in the same way I would support any piece of software I was teaching. Even better, the students were becoming independent learners, learning how to support their own learning by accessing all the available information necessary from the internet.

We moved onto Flash as our next topic, which the students loved. After learning a few basics they had to create adverts for the club. Winners were selected and I put them onto the school website. They were really proud of their efforts. I have also been able to experiment with ideas in the club so I can introduce them into lessons. It has been great to see their confidence grow and watch as they realise that computers can do more than just browse the net and creating PowerPoint presentations. They have created mini e-portfolios to house everything they do in the Computing Club and record anything they want to share with the others.

Our next step will hopefully be looking at Greenfoot and introducing programming. I would be lying if I said I am not nervous as I really did hate programming at University. But I am excited to re-visit it and, to be honest, if I can’t get a handle on it and get my students involved in computing, then maybe I’m in the wrong job or at least the wrong department?

Michael Constantino

GETTING TO GRIPS WITH GAMES IS CHILD’S PLAY WITH KODU

Kodu is a free visual programming environment developed by Microsoft and accessible to children of all ages. Nicki Maddams, from Hartdown Technology College recounts her experiences and urges others to give it a go.

Kodu is controlled in two ways, my preferred method is using a wired Xbox controller, but you can also use a mouse and keyboard. I began by spending a considerable amount of time “playing” with Kodu myself to learn how to use it. It is fairly intuitive so once I had mastered the software I wrote a seven-lesson scheme of work taking children from the basics right through to designing their own game.

Games running slowly can be a problem if you do not have hi spec machines.
One handy tip - changing the Graphics to Standard and turning off Smoothing in the Visual Effects has a big impact as students can become disillusioned if their games are unresponsive.

Children are keen to have a go, though some with more success than others. The most successful learners are happy to try different things, explore the menus and learn independently. Unfortunately not all children are happy with this and require constant reassurance. This proved quite a barrier. It is tempting to give them all the answers, although as a teacher it is also important to encourage them to problem-solve on their own. Independent thinking is something I am trying to encourage more and more throughout the curriculum as it is a huge requirement when learning to program in any way, whether it is through Kodu, Scratch or more traditional methods. Kodu supports children in thinking (and learning) for themselves with context sensitive assistance.

Some may prefer to teach Kodu within a less guided club environment. I run a programming club and although I often set them challenges they much prefer to create their own thing and often produce games that are much more imaginative than anything I could have come up with! It is great to see children running free with the software in this way, although I doubt this method would work in a lesson context.

One obstacle I have found more difficult is staff training. Not all staff who teach ICT are from a programming background and find the use of these kind of tools quite daunting. I have found one of the best methods for learning Kodu is by challenging myself to recreate old arcade games. Situations often arise that you have to solve and this extends your knowledge. Some examples that I have worked on are Pong, Pacman and Space Invaders. The latest version of Kodu includes some excellent step-by-step tutorials that you can work on at your own pace. It is great software for beginners, so if you haven’t tried it yet, see the link in the supplement, download it and have a go!

Nicki Maddams

SWITCHEDON web supplement: www.computingatschool.org.uk
DEVELOPING A CURRICULUM FOR SCHOOLS: CAS NEEDS YOUR HELP

Last issue we reported on the publication of ‘Computing: A Curriculum For Schools’. At the CAS Teacher Conference initial plans were sketched out to develop this into a fully fledged resource for teachers. Cynthia Selby outlines the ambitious plan.

Following conference, the CAS Resources Working Group has now been formed. A group of volunteers, mostly teachers at secondary level from across the country, have joined together to tackle the challenge of moving the Computing Curriculum from paper into the classroom.

The objective is to provide both new and existing teachers with resources, aligned to the Computing Curriculum, which can be used in Key Stage 3 classrooms. We recognise that ICT teachers will have different opportunities to introduce computing in their existing curriculum and want to provide a range of self standing resources that can be used straight out of the box. A set or box of resources may be topic focused, suitable for 1 or 2 hour lessons, and cover specific topics from the curriculum. On the other hand, a box of resources may be themed, suitable for a half-term project, and cover a broad range of concepts and skills from the curriculum. This flexibility should provide teachers with an opportunity to introduce aspects of the Computing Curriculum suited to the needs of their learners. Teachers who are currently delivering aspects of the curriculum in their schools have already contributed some resources to start the project off. These are now being prepared for presentation. Watch the CAS website for further information about their availability.

To provide a complete curriculum is a huge task. We are therefore appealing to all CAS teachers to help. All contributions will be appreciated. It doesn’t have to be perfect. Teachers rarely have the time to polish resources to their liking, but any contribution will be developed into a consistent format. A contribution could be a lesson idea, links to useful web sites or tools, a worksheet, a fully resourced 60-minute lesson, or even a series of lessons. All authors will be credited with their work. Many hands make light work. Can you help? If so, please get in touch with Stephen Hunt or myself. Contact details in the supplement. Cynthia Selby

INTRODUCING AND DEVELOPING A SCHEME OF WORK

At Ysgol Gyfun Gwyr, Swansea we have just completed a successful RoboMind module with Year 7 and are trialling a Kodu module with our Year 9 (pre-GCSE group) ready for inclusion into the Year 8 curriculum next year. The Year 7 work was good but as with all Schemes of Work it will be refined and developed further next year. The pupils really enjoyed writing commands but some got stuck on the higher elements involved in programming. The Year 9 (Kodu) module was extremely successful and has generated a lot of interest in my pre-GCSE group as an enrichment task. As we work on the task both the pupils and I are evaluating the work. This evaluation will shape the new Year 8 module next year. We will then build on that. Using a year 9 group to trial and evaluate a module intended for a different year has been very useful, with the pupils valuing the responsibility involved in checking the tasks and suggesting improvements. Next year we hope to put together a small Year 9 group together to put forward a competition entry for the BAFTA Young Game Designers 2011 award.

Gareth Edmondson

COMPUTER SCIENTISTS ASSISTING IN SCHOOLS

A group of students from Loreto College Coleraine, Specialist School, were recently invited to visit the school of Computing and Information Engineering at the University of Ulster, Coleraine. This was the climax of a 12-week Tutoring in Schools project. During the project, UU students Ciara Murphy and Rebecca Lynam shared their expertise with students from Loreto in an extra-curricular programming club. While in first year at UU, Ciara and Rebecca were highly placed in the Microsoft Imagine Cup for games development – beating many final year and post-graduate students. As part of the project, Loreto students learned the principles of software design and development. This led to group development of games, including ‘Super Maria’ – a clone of Super Mario Brothers.

During the visit to University of Ulster, Loreto students were met by current post-graduate students including Debbie Rankin, a past pupil of Loreto and now a leading research student at UU. Pupils got some hands-on experience of their research, including a 3D networked ’Pac Man’ clone and mobile phone ‘app’ development.

Throughout, the pupils benefited from Ciara and Rebecca’s talent which built on their own enthusiasm. UU’s Dr Michaela Black and Martin McKinney recognised the mutual benefit and thanked Loreto for facilitating the Tutoring in Schools project. Clarke Rice
NO CAS LOCAL HUB NEAR YOU? SET ONE UP!

Several new CAS Teachers’ hubs held their first meetings in the last Summer Term, offering an opportunity for networking with fellow ICT or Computing teachers and attracting new members to the CAS group. The first Sussex Hub meeting took place on 17 May at Dorothy Stringer School in Brighton. This was followed on 7 June by the inaugural meeting of the Wiltshire Hub at New College, Swindon. On 30 June Bristol and Preston held their first meetings, at University of West of England (UWE) and Our Lady’s School respectively. Finally Milton Keynes met at The National Museum of Computing (TNMOC) Bletchley Park on 7 July.

Many hubs have made plans for a meeting in the coming term.
12 Sept: Sussex, Dorothy Stringer School, Brighton
22 Sept: South West, UWE
26 Sept: Hertfordshire, venue tbd
3 Oct: Essex, venue tbd
6 Oct: High Wycombe, Royal Grammar School, High Wycombe
12 Oct: Surrey, University of Surrey, Guildford
12 Oct: Norfolk, University of East Anglia (UEA), Norwich
20 Oct: Preston, Our Lady’s School, Preston
Oct (date tbc): South and East London – BCS London office
3 Nov: Dorset and South Coast, Poole Grammar School

If your area isn’t represented, and you’re interested in getting a hub up and running, please get in touch and we’ll help.
CAS have produced a guide to get you started. Further details in the supplement.
Claire Davenport

LOBBING FOR CURRICULUM CHANGE: CAS GOES TO WHITEHALL

As many of you will know, as well as tons of fantastic grassroots work, BCS and CAS have also been busy making the case for computing at school at national level. Just at the moment there is a lot happening. Here are some of the bigger things:

- The Livingstone/Hope Report came out. Links to this, and other reports mentioned are in the supplement. Recommendation 1: “Bring computer science into the national curriculum as an essential discipline”. Fantastic. The same group is now spinning up “NextGen skills” to promote further progress.
- The DfE is running a Review of the National Curriculum. We wrote a submission and have had subsequent meetings with officials at the DfE, up to and including the minister, Nick Gibb. They haven’t exactly said “oh yes, we’ll do that”, but clearly get the Computing message, and we are firmly on their radar.
- The Royal Society Computing in Schools panel has been working for the last year under the chairmanship of Steve Furber (who was deeply involved in the genesis of the BBC micro). CAS is well represented on the panel. The RS made an interim submission to the DfE for the National Curriculum review (link below) that supported the main CAS messages. The report will come out late in 2011; the Royal Society is a heavy hitter, so I expect it to be taken notice of.
- The Commons Select Committee on Education ran a short investigation into the English Baccalaureate. We made a submission arguing that Computing should be a subject that contributes to the EBac, something that would really help head teachers feel able to offer Computing at GCSE. The Committee’s report (yes, link below) was pretty critical about the EBac as it is now, but didn’t make any explicit mention of Computing. I’m writing to the committee to suggest a broader dialogue about Computing.
- Tom Crick has been enormously effective in reaching out to politicians in Wales. He has addressed the Welsh Government and organised the CAS in Wales Teachers Conference in conjunction with the Technocamps project, which had the Deputy Minister for Skills as a keynote speaker. He has been invited to consult on the new Welsh strategic science framework by the Office of the Chief Scientific Advisor, as well as being invited to sit on the Department of Education and Skills’ Digital Skills advisory group. CAS in Wales are also working closely with the Campaign for Science & Engineering (CaSE) to support the Computing education agenda in Wales, especially with respect its position with the STEM portfolio, the availability of Computing qualifications and initial teacher training.
- In Scotland, the Royal Society of Edinburgh is leading a project to provide exemplification for Computing Science elements of the new Curriculum for Excellence, with input from Education Scotland, the universities and industry. The Scottish Qualifications Agency are developing new Computing qualifications that centre on core aspects of computation and that will provide a sound basis for study beyond the secondary school level.

While the outcomes of all this activity remain to be seen, CAS is now getting a sympathetic hearing in the corridors of power. Over the last two years we have gone from being an unknown guerrilla group to being a serious player. So far, however, we have not engaged much with school leaders (head teachers, governors, and the like). It is time to do that, especially as schools are encouraged to become more autonomous. With that in mind we’re getting in touch with head teachers’ associations and governors organisations - but maybe you can help too. Are there any head teachers or governors in your orbit with whom the CAS message might resonate, and who might help us get into dialogue with school leadership?

Simon Peyton-Jones
Can Programming be Taught Without Introducing Variables?

If you wanted to teach programming, what would you show first? A simple command, then a variable and assignment statement to do a simple calculation; and maybe next a program that contains a loop. Those are the basics, aren't they?

Well, not necessarily, for there's a style of programming, and languages to support it, where none of these things are used. In functional programming, there are no commands, no variables (not ones that change their value, anyway), no assignments, and no loops. By losing these features of conventional languages, a functional language gains other things that sometimes have greater value: a way to treat data without worrying how storage for it is allocated, and the ability to build programs from kits of parts that interact in productive but manageable ways.

Let's consider a very simple problem: summing the squares of the first \( n \) natural numbers. A functional program to solve this problem is written,

\[
\text{sum } (\text{map square } [1..n])
\]

where \( \text{square x} = x * x \)

The program begins with a list \([1..n]\) containing the first \( n \) numbers; applying the function \( \text{map square} \) to this list makes a new list in which each number has been replaced by its square; then the function \( \text{sum} \) takes this list of squares and adds them all up. A conventional program for the same task would have to introduce two assignable variables, one to range from 0 up to \( n \), and another to keep a running total of the squares:

\[
k := 0; \quad \text{sum} := 0;
\]\n
while \( k < n \) do
\[
k := k+1; \quad \text{sum} := \text{sum}+k*k
\]
end

This example is very simple, but it still shows that whilst the loop in the conventional program is all of a piece and cannot be split into smaller components, the functional program contains several reusable parts. Only the function \( \text{square} \) is specific to this program, and the rest is put together from general-purpose components such as \( \text{sum} \), which adds up any list of numbers, and map, which may be used to apply any function uniformly to all members of a list.

At first sight, it seems wasteful to create a list containing the \( n \) numbers, then make another list containing their squares, and finally add them up. But the answer from a functional program is the same whatever order the calculations are carried out, and a compiler is free to interleave the processes of generating the numbers, squaring them, and summing the squares, so that the program actually carries out the same actions in the same sequence as its conventional equivalent. It's this opportunity to view programs as more than an explicit sequence of actions that makes functional programming such a valuable tool for teaching students to think about programming, and also provides the inspiration behind Google's MapReduce system for carrying out immense calculations on clouds of computers working in parallel.

Graphics & Functional Programming: GeomLab

At Oxford University, we think functional programming is so important that we make it the basis for the very first programming related course taken by undergraduates. To give students a taste of functional programming before they come to us, we've put together an online activity called GeomLab that combines functional programming with graphics. The basic functions of the Geom-Lab language assemble pictures from pre-defined tiles by allowing them to be placed side-by-side or one above another. The first picture shows the result of the expression

\[
\text{man } S \ (\text{woman } \& \ \text{tree})
\]

where a stick man is placed beside (\( S \)) a picture in which a woman appears above (\( \& \)) a tree.

Participants are guided through a sequence of exercises where they begin to describe more complex pictures using recursion. They create the second picture from a varying series of rows, each made up of several copies of the \( \text{man} \) picture, describing it with a pair of nested recursive functions.

Before long, they are investigating how more striking pictures, like the Escher image shown below, can be assembled from a handful of basic tiles.

We have used GeomLab successfully to run whole-class extension activities for a wide range of ages from Year 10 up. An hour is enough to get an idea of what is possible, but the website provides sufficient material for an exploration that lasts one or two days, and others have reported that they have used the materials as the basis of a computer club meeting weekly over the course of a term. See the web supplement for the link.

Michael Spivey
WHEN SIZE DOES MATTER (PART ONE): GETTING TO GRIPS WITH THE NOTION OF ALGORITHMIC COMPLEXITY

The AQA Computing specification for A2 requires candidates to have some appreciation of the complexity of a problem defined as the growth rate of the algorithm which solves the problem, i.e. its big O complexity. In a two-part article, AQA Chief Examiner, Kevin Bond suggests some ways to introduce this tricky area to your classes.

When does size matter? When an algorithm executes on its input to produce an output. For example, one might have to wait a significant amount of time for a list of 100 000 strings to be rearranged into alphabetical order. The time an algorithm takes compared with another algorithm performing the same task is of considerable interest when the data set is of significant size. Computer Scientists are interested in estimating how a particular algorithm's execution time depends on the size of input notwithstanding that different computers run at different speeds. To eliminate the difference in speed, Computer Scientists try to answer the question "what happens to the execution time when the size of the input is doubled?" For example, if the number of strings to be placed in alphabetical order is doubled does it take twice as long, four times as long, etc?

Many students have difficulty solving problems where the solution involves making estimates. The Physicist Enrico Fermi used to challenge classes with problems that seemed impossible. One such problem was estimating the number of piano tuners in Chicago given only the population of the city. The answer is about 150 (see supplement for the solution). The approach to solving this type of problem became known as the Fermi Approach. It relies on knowing some facts, ignoring unnecessary details and making reasonable assumptions. The same approach can be used to estimate the speed of execution of an algorithm.

Activities utilising the Fermi approach (see column left) can be made kinaesthetic. One activity is tracing a Bubble Sort on weights, ordered by increasing size of weight. The diagram shows an outline of a first pass with a set of objects initially placed in descending order of weight, D, C, B, A. Object D is first compared with C (and swapped), then with B, and so on. The process moves D creating a new ordering C, B, A, D. The weighing cycle then begins again and finishes with the ordering B, A, C, D. The final cycle produces the desired outcome A, B, C, D. If the letter n is used to represent the number of weights then the number of weighing cycles is \( n - 1 \). This is exactly the same as the number of fetch-weigh-return operations in each cycle. Let's represent the time in seconds for a fetch-weigh-return operation by the letter \( t \), then one cycle will take \( (n - 1) \) times \( t \) seconds. Therefore, \( n - 1 \) cycles will take \( (n - 1)t \) times \( (n - 1) \) times \( t \) seconds. The Enrico Fermi Approach says that we can approximate \( (n - 1) \) to \( n \) when \( n \) is large compared to \( 1 \), say \( n = 100 \) weights. Therefore, \( (n - 1)(n - 1)t \) is to a good approximation \( 2n^2t \) times \( n \) times \( t \) when \( n \) is large. We write \( n \) times \( n \) times \( t \) as \( n^2t \). We now have a formula for how long the weighing algorithm will take to execute on large inputs. Call this total time \( T \). Therefore \( T = n^2t \).

We can now use this formula to estimate by what factor the execution time changes when the input size is doubled, e.g. from 100 weights to 200 weights.

Time for 100 weights: \( T_{100} = 100 \times 100 \times t \)
Time for 200 weights: \( T_{200} = 200 \times 200 \times t \)
This is 40 000 compared to 10 000, a ratio of 4 to 1. This is bad news.
Doubling the number of weights takes 4 times as long. If the time for the fetch-weigh-return is 5 seconds, then \( T_{100} = 50 000 \) seconds and \( T_{200} = 200 000 \) seconds. Approximately 56 hours compared with 14 hours!

This exercise could be carried out with weights constructed from 35mm film canisters containing coins to make the different weights. The canisters should be colour coded so that they can be distinguished visually. A balance is not necessary because the weights can be compared by hand. The activity can be extended with a "no exchange of weights" indicator. Using a cup placed up or down, the cycle of weighing continues until the no exchange indicator shows no exchange of weights, thus indicating the list is fully sorted. Before each cycle the cup is reset. This is an abbreviated article, taken from an Educational Computing Services course for A2 computing. Details in the supplement.

Kevin Bond

![Diagram of algorithm execution](image-url)
FROM BASICS TO A LEVEL: FIVE DAY PROGRAMMING SUMMER SCHOOL

A Python Summer School, funded by the TDA, at Anglia Ruskin University introduced trainee ICT teachers to programming. The resources will be invaluable for many teachers, covering the basics to the level needed to teach A2 Computing.

Many trainees come to ICT teaching without a Computing-related degree or programming experience. However, once in post they may be asked to deliver GCSE or A-Level Computing. A secondary PGCE course is very full and does not allow much time for subject knowledge development. The course was an intensive week for trainees to experience programming. Adam McNicoll and I, covered topics that included lists, functions and files, building a database, Object-Oriented Programming and GUI’s.

Most trainees had no prior experience of programming. The first day was hard as they covered many of the basic concepts. An AS Computing student might cover this ground in six weeks! However, these basic skills were practised over and over again in all the material covered in the subsequent days. Another big jump came on the fourth day, starting Object-Oriented Programming using Adam’s inventive Farming scenario. By the end of the day they had written a complete class and on the next were able to create an event-driven GUI to support it. So many skills in one week! The trainees made impressive progress and feel much better prepared to teach computing in their schools. They helped each other and worked well as a group. Paul Curzon talked about CS4FN, putting programming into the context of stimulating pupils’ interest in Computing. The magic and audience participation made it a good end to the course.

Why Python? Recent research has highlighted its value as a programming language for beginners. Its syntax is simple and close to pseudocode; it is freely available and works on all platforms. It is also a perfect language to teach object-oriented programming, lends itself well to server-side applications and comes complete with sqlite to build database applications. PyQT was our tool of choice for GUI’s, enabling professional looking interfaces to be built.

This type of training should be incorporated into the PGCE but are keen to repeat the course for qualified as well as trainee teachers. If you would be interested in attending a future summer school please get in touch. All the materials are available online for others to use, trainee or teacher alike. See the supplement for links and contact details.  

Sue Sentance

USING LINUX TO EXPLORE ARTIFICIAL INTELLIGENCE

In Issue 4, Aaron Sloman made the case for teaching introductory courses in Artificial Intelligence / Cognitive Science using ‘thinky’ programming techniques. He has made available a wealth of introductory programs implemented in the Linux based AI development environment PopLog. If you would like to access these but run Windows at school, CAS members have provided two ways to run Linux. Firstly Lee Gillam has set up a cloud based Linux server with AI tools preinstalled. All you need is an account to access it. Alternatively, Tim Bateson has provided instructions for setting up a Linux Virtual Box on school pc’s and an installation package for the AI tools. Both can be used to give pupils experience of a Linux environment. A link to Aaron’s more detailed instructions and support in the supplement.

INCREASING USE OF BLUE-J TO TEACH JAVA

BlueJ is a free, open source educational programming environment for the Java language. First released in 1999, it supports development of generic applications – which puts it in competition to professional development tools, such as NetBeans and Eclipse – but it differs from those systems in important aspects. It is designed specifically for beginning programming at sixth form or early university level. It features a simpler interface and additional tools designed specifically for learners are included. Most prominent is a graphical UML-style class structure display which helps students understand the fundamental concepts of object orientation.

BlueJ is a sibling of Greenfoot, both developed by the Programming Education Tools Group at the University of Kent. While both tools have common ancestry, they have distinct uses. BlueJ and Greenfoot are often used in sequence – students start with Greenfoot and transfer to BlueJ once they are ready for deeper, more generic study of programming concepts. Common design elements make the transition easy and avoid conceptual discontinuities.

Last year saw a significant increase in use with around 1.8 million active users. Support for teachers using BlueJ was recently improved with the opening of the ‘Blueroom’, a community where teachers can share material, discuss teaching strategies, ask questions and talk to the development team. It is still in an early phase and contributions are actively encouraged. If you are a BlueJ user please share your ideas—colleagues will be grateful!

Michael Kölling
SUCCESSFUL FIRST CONFERENCE FOR COMPUTING IN WALES

The first Computing in Wales conference, hosted by Computing At School (CAS), supported by BCS Academy, and the Technocamps project, drew over 100 attendees on 1 July at Swansea University. The conference provided teachers, examination boards, academics and policymakers with a forum to discuss issues surrounding computing education in Wales and was organised by Dr Tom Crick, leader of CAS in Wales.

The Deputy Minister for Skills, the Welsh Government, Jeff Cuthbert, gave a keynote speech highlighting the economic importance of developing computer skills amongst young people, explaining: “I want to make sure that all learners in Wales have access to digital technologies and that they have the skills to make the best use of them. We want learners to fulfil their potential and Wales to prosper in a digital society.” Other invited speakers were Clare Riley (Group Manager, Education Relations, Microsoft), David Davies (Director of NHS Engagement, NHS Wales), Gary Stone (Head of Strategy & Operations for Digital Wales, Welsh Government) and Professor Faron Moller, Director of Technocamps project.

LEADER WINS ‘I’M A SCIENTIST’ COMPETITION

In June Tom Crick triumphed in the Chromium Zone of “I’m a Scientist, Get me out of Here”. Over 100 scientists and 5,000 students from 150 schools took part, with 7,500 questions asked and 68,000 visits to the website. Tom, the only Computer Scientist in the competition, took part in 14 chat sessions and answered over 500 questions from school students. In response to questions, Tom highlighted the wide impact of technology, encouraging students to look into computing as a career, as well as advising on useful A-Levels. Tom said: “It has been a very enjoyable and rewarding two weeks. It is a hugely effective way of engaging with a large audience about your role as a scientist.” Tom is now continuing his science communication work as a 2011 British Science Association Media Fellow with BBC Wales. Claire Davenport

CHALLENGING MYTHS ABOUT CAREERS IN COMPUTING / IT

The provision of computing careers advice is sadly lacking. To the outsider the skills needs of the IT industry appear confusing. A recent event at City University sought to address this. Organiser Andrew Tuson explains.

The day focused specifically on the opportunities for students to study for challenging careers in Computing and IT and made a start on addressing some widespread misconceptions. The opening talk covered how the IT industry works, the current economics and skills needs and what students need to succeed by examining the evidence base. The IT graduate employment market is demanding much higher base competence in entry level roles than even ten years ago. In fact IT is unique among STEM professions in requiring excellence in both technical and soft skills in graduates. It touched on who leads the IT profession and ended with the question ‘What’s a CIO?’: Few hands went up. Students are attracted to aspirational professional roles such as judges, barristers, surgeons, etc. The IT profession has these, but no one knows of them. Addressing this, Marcus East, Chief Information Officer of Comic Relief gave an inspirational talk on running IT and the fun he has doing it. A motivational topic for students is computer games. Chris Child (Childish Things Ltd) gave an overview of how the games industry works, and what employers look for. The distinction between creative and technical roles and the need for excellence in computer science in the latter were two of the take-home messages.

What IT employers look for had to that point been quite abstract. Stuart Elliston (Linkfield Technologies Ltd) presented his experiences of hiring graduates. Thomas Ng talked about the exciting work CAS are doing to raise career awareness and involve teachers. The final talk was on University Courses in Computing & IT. As an undergraduate admissions tutor of 10 years experience I hoped to help make sense of the range of courses and skills needed to study the subjects. The choice is wider than just ‘computer science’ so students may wish to investigate what aspects of computing they are interested in. The value to many admissions tutors of A-level computing both as a science and indicator of motivation was highlighted. A supporting set of careers slides with links to further resources is available via the Google Group.

Andrew Tuson

SPREADING THE MESSAGE TO SCHOOL GOVERNORS

The latest issue of Matters Arising, published by the National Governors Association contains an article written by CAS member Neil Collins, Communications Officer for Norfolk Governors’ Network. There is a lot CAS can do to help support IT Link Governors spread the message about computing in schools, and support CAS teacher members in their relationship with Governor curriculum committees. Having friendly eyes and ears on the governing body can be very helpful with senior management teams. We would like to put together a support pack for such governors. If you are an IT Link governor, or know your governor is supportive of computing in school please get in touch. We would like opinions on the best ways that we can support the governors.

Claire Davenport

SWITCHEDON web supplement: www.computingatschool.org.uk
PEDAGOGY, GREENFOOT AND GAMES AT CAS SUMMER SCHOOL

The first CAS Summer School, based at the magnificent venue of the National STEM Centre in York was very well received. Clive Beale spent a few days of the holiday getting to grips with Greenfoot and making new friends.

I'm no good at break-the-ice activities. My ‘bingo’ card was blank and I needed to find someone who had played Dungeons and Dragons. ‘Ever played D & D?’ I asked the chap coming down the stairs. It turns out that yes, Ian Livingstone OBE, co-founder of Games Workshop and Life President of Eidos, had indeed played once or twice. Ian was there to kick off the first CAS Summer School with a thought-provoking talk about the state of the UK video games and visual effects industries. This included a brief history of video games which brought a wistful smile to many faces. The venue was the magnificent National STEM Centre in York. Everything is on-site, including smart accommodation and the marvellously named Quarks Restaurant, where your dinner doesn't actually exist until you open the serving dish to look at it.

Day two found us in the classroom where Neil Brown introduced Greenfoot and over the course of the day taught us the basics. The lessons were practical, well-paced and good fun. I'd used Greenfoot before but still learned a lot. An evening trip to the bowling alley helped purge classes and objects from our heads, whilst back at the campus a bus driver gave us a free ride to the local pub where a round came in at under a fiver. It was at this point that I wondered whether the STEM centre was recruiting.

On the last day we covered collision detection and how to handle strings, counters and scoreboards. We also had a long chat about pedagogy and in particular the reasons for the bi-modal distribution of achievement seen in computing courses. Neil finished by giving us some solid tips for teaching Greenfoot. We were a true mixed ability class, from those who had spent whole nights typing out machine code listings in Your Sinclair to those who had little programming experience. It is a testament to Neil's teaching— and to Greenfoot's design— that by the end of the two days we were all writing classes and methods to create working programs.

The CAS Summer School was one of the best CPD courses that I've ever been on, in teaching or in industry. We learned a lot, exercised our brains, had a laugh and met a bunch of interesting people. What better way to spend a few days of your summer holiday?

Clive Beale

GOOGLE COMPUTER SCIENCE GRANT AID

Google funds a grant program called Computer Science for High Schools (CS4HS), designed to help encourage students to study Computer Science and introduce the latest CS techniques and tools. The CS4HS program provides funding to European, Middle Eastern and African universities to work in tandem with local high schools. To be considered for funding the projects must be scalable, impact a wide cross-section of students from all backgrounds, conform to a “train the trainer” model and, most importantly, inspire the next generation of computer scientists.

In 2011, three UK universities were awarded funding. Queen Mary University of London received continued support for CS4FN, the University of Kent for Greenfoot and the University of Manchester for the UK School Computer Animation Competition. Read more about the program and the 2011 recipients on the CS4HS website. Link in the supplement. Application deadlines for 2012 will be announced around December.

GOOGLE RISE SCHOOLS FUNDING

Google RISE Awards are designed to promote STEM and Computer Science initiatives. Google provide awards to organisations working with primary and secondary schools to provide enrichment programs with an emphasis on engaging under-represented groups. Ideal programs and projects should inspire excitement about STEM and CS, enrich students learning and include hands-on activities using CS. The application portal for EMEA is open in September and October. Relevant links and further details in the web supplement.

Alison Daniel-Cutler
MORE CLASSROOM
CS4FN RESOURCES

A Computer Science For Fun project (cs4fn) recently featured in the Royal Society Summer Science Exhibition, in an exhibit about how the human brain processes faces, and whether robots can do the same. A mini-documentary on the subject is now available on YouTube.

The video is 14 minutes long and covers issues in robotics like interpreting faces, giving robots social skills and what robots could be capable of in the future. It also deals with how our own brains recognise faces, and what life is like for someone without that ability. How do you cope if you don’t recognise your loved ones’ faces? We also hear from a world-class portrait painter about how he sees faces.

We also produced cut-out experiments that you might be interested in using with your students. They can program a robot face with some basic emotional expressions, or see an illusion in which our brain is tricked into seeing a hollow face point outwards.

You can explore more of the science of faces in the current issue of cs4fn. Copies of the magazine can be ordered free for schools or you can view the material online by going to the faces portal on the cs4fn website. Links to all the resources can be found in the web supplement.

Jonathan Black

BCS ACADEMY PRODUCE FACTSHEET FOR EMPLOYERS

Bill Mitchell of the BCS Academy has put together a very useful factsheet to discuss with companies the need for computing in the school curriculum. The four page factsheet makes a compelling case for school leavers pursuing a variety of disciplines requiring a familiarity with rigorous computing principles. It contains a succinct list of evidence and a case study in computational biology. As well as providing companies with a compelling argument, the factsheet will be of use to teachers too, helping them argue their case for curriculum change at school.

Computing at School was born out of our excitement with the discipline, combined with a serious concern that students are being turned off computing by a combination of factors. Our goal is to put the fun back into computing at school. Will you help us? Simply mail membership@computingatschool.org.uk

Many thanks to the following for help with this issue of SWITCHEDON: Clive Beale, Jonathan Black, Kevin Bond, Michael Constantino, Tom Crick, Alison Daniel-Cutter, Claire Davenport, Roger Davies, Gareth Edmondson, Lyndsay Hope, Toby Howard, Simon Humphreys, Dale Jones, Michael Kölling, Nicki Maddams, Bill Mitchell, Simon Peyton-Jones, Matthew Pickering, Clarke Rice, Cynthia Selty, Sue Sentance, Aaron Sloman, Genevieve Smith-Nunes, Michael Spivey, Andrew Tuson.

www.computingatschool.org.uk

A PAUSE FOR THOUGHT

This problem involves circuits which feed back into themselves. The output of a gate can be traced through the circuit back into the input of the same gate. This can lead to all sorts of interesting behaviour. What will happen when you switch on the circuits below? What will happen if you change the gates to different types?

Random Access Memory is built using logic gate circuits which feed back into themselves. The most simple of these is the ‘flip-flop’: a device which has two different stable states, and can thus ‘remember’ if a switch has previously been flipped. This problem is from a poster produced by the NRICH Project. Alongside thousands of mathematical problems they also produced free monthly posters, many of which are relevant to teaching computing. Links to their website, the circuit poster, solutions and extra resources in the web supplement.