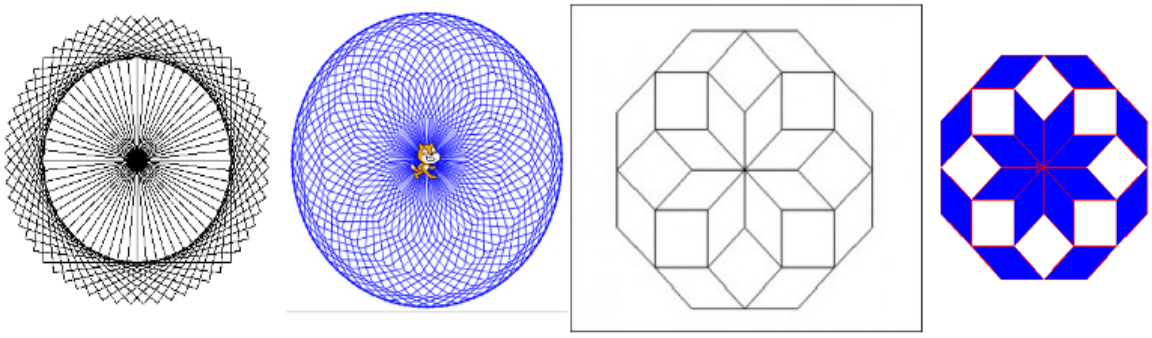
COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE
In collaboration with BCS, The Chartered Institute for IT

UCL ENGINEERING
Change the world

UCL/CAS Training for Master Teachers
Algorithms and Programming
Modules 1 and 2



AN OVERVIEW OF AN EXEMPLAR COURSE IN ALGORITHMS AND PROGRAMMING MODULES 1 AND 2

CONTENTS

ACKNOWLEDGEMENTS

Thanks to

- Professor Steven Hailes, Deputy Head of Department of Computer Science at UCL, and Rae Harbird, Teaching Fellow in the Department of Computer Science at UCL, with whom it has been a pleasure to work on this CAS/UCL venture on Training CAS Master Teachers.
- The first cohort of level 1 Master Teachers who have helped to develop and tune this set of Module 1 Workbooks for Master Teachers/Teachers.
- James Dent, CAS Master Teacher and Head of Computing at the Richard Hale School, Hertford, for trialling with me the Worksheets and Activities with his year 8 and 9 pupils, and the invaluable feedback that ensued.
- Clive Benham Head of Computing, and Caron Fry computing teacher for similar trialling work at Passmores Academy, Harlow.
- Phil Alcock Master Teacher and Head of IT, for the work at Little Parndon Junior School, Harlow
- To Mark Dorling for his enthusiasm and support as co-author of a paper which helped to set the stage for this Course.

BACKGROUND TO THE NATIONAL CURRICULUM – WHAT IS ACHIEVABLE

Scratch: A Way to Logo and Python by Dorling, M., White, D.,
Excerpt from a paper: to be presented at The Proceedings of the 46th SIGCSE technical symposium on Computer science education (2015) Kansas City, Missouri, USA: ACM.

From September 2014, pupils in English state-maintained schools will be expected to follow the programmes of study set out in the national curriculum document. Department for Education. 2013. The national curriculum in England, Framework document. Available: www.education.gov.uk/nationalcurriculum [Accessed 13-08-2013].
Computing at School has responded has responded with targeted resource: Computer science: A Curriculum for Schools 2012. Available from: http://www.computingatschool.org.uk/data/uploads/ComputingCurric.pdf

The programme of study has high-level aims in terms of the introduction of computer science. The following extracts illustrate learner capabilities at different stages of primary and secondary education.

At Key Stage 2 (age 7-11) pupils should be able to (amongst other things): "… solve problems by decomposing them into smaller parts." and "use ***sequence, selection, and repetition*** *in programs; work with variables and various forms of input and output." and also "… detect and correct errors in algorithms and programs." (p. 189).At Key Stage 3 (ages 11-14) pupils should be able to: "… make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular*

programs that use procedures or functions." and also "Understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming…" (p. 190).

For primary educators there is no specification in the programme of study to teach either a graphical or text-based language, instead the emphasis is placed on teaching concepts and principles. In contrast, the programme of study for secondary education, Key Stage 3 makes it explicit: Pupils should be able to: use two or more programming languages, at least one of which is textual, to solve a variety of computational problems (p. 190).

Given good pedagogy, expert support for teachers and a strategy for supporting all pupils with making the transition from graphical to text-based languages, the authors conclude:

- *Pupils can learn a text-based programming language whilst at primary school.*
- *All secondary school pupils are able to master the basics of text-based programming.*
- *A visual programming language like Scratch would be a good introduction to programming for beginners at both primary and secondary levels.*

RATIONALE FOR COURSE DESIGN: COMPUTING -- THE NEW LITERACY

We recognise the need to maintain high standards for Computing and at the same time make the subject accessible on a broad scale in schools. As a result of research and experience with Teachers (see: http://ispython.com/wp/wp-content/uploads/2014/11/Asigsce17_titled.pdf) and classroom experience, we are designing a Course which integrates the following components in an approach to Algorithms and Programming Module 1:

1. An easy to set up and use environment (with an immediate graphical response)
2. A projects-based developmental course exploring relevant cross-curricula subjects
3. Computational Thinking underpinning the course
4. A Constructionist base in our Teaching and Learning Models.

1. EASY (FREE) TO SET UP, USE AND LINK COMPUTING ENVIRONMENTS (WITH AN IMMEDIATE GRAPHICAL RESPONSE)

We have chosen, in the first instance, just 3 instructions which drive sprite/turtle-like objects in

- Programming projects in an 'Unplugged' Programming Language (UPL: Logo/Python) -- Workbooks 3-5
- Action Geometry: the background to a cross-curriculum topic Workbooks 6-7
- **Pathway 1**: Unique Patterns in Scratch 2.0 and Python 3 -- Workbooks 8-12
- **Pathway 2:** Creating quality learning and teaching materials in Computing including event-driven programming approaches. Workbook 13 (to be completed)

All three environments are easy to access and use off-line and locally.

2. PROJECTS-BASED DEVELOPMENTAL COURSES ON CROSS-CURRICULA SUBJECTS

Our choice of cross-curricula subject is Mathematics, starting with regular shapes in geometry starting from drawings of the square, polygons, stars, circles, leading to the creation and exploration of unique symmetrical patterns. We use the 5 control structures of programming to drive the sprite/turtle in the early projects which include

- Designing Capital Letters
- Square Arrays of Squares
- Generalisation to Rectangular Arrays of Rectangles
- Polygons and Polygrams (Stars)
- Rotation and Translation of Shapes
- Creating Unique (coloured) Patterns

BEYOND SPRITES AND TURTLES

- **Pathway 3:** A 'number guessing' game – away from sprites and turtle. Two paradigms of Algorithm.
- **Pathway 4:** Arithmetic in the 21st Century. The Interactive Shell. Algorithms for evaluating linear arithmetic expressions. Recursion.

3. COMPUTATIONAL THINKING

For a detailed discussion, see:
http://ispython.com/computational-thinking/

4. A CONSTRUCTIONIST BASE IN OUR MODEL OF LEARNING AND TEACHING.

See http://ispython.com/workbook-2-learning-and-teaching-in-computing/

OVERVIEW OF ALGORITHMS AND PROGRAMMING

This series of 15 Workbooks covers the Algorithms and Programming component of the UCL/CAS Training Course for Master Teachers Level 1. The material covered is directed broadly at the KS1/KS2/KS3 bands in schools in Module 1, and KS4/GCSE IN Module 2 (in preparation).

We seek to deliver a Course in Algorithms and Programming which illustrates and integrates some of the wide variety of approaches to learning and teaching available in a ***pedagogy of Computing,*** to develop and make explicit the ***Computational Thinking*** that underpins the subject. The Course explores two contrasting paradigms of processing in programming:

- ***sequential process programming,*** in which the order of execution is sequential and largely determined by the programmer.

- ***event-driven process programming,*** in which the order of execution of the program is determined by outside events.
- ***and an introduction a third paradigm: object-oriented programming***

We summarise:

A Five Element Model of Computational Thinking (ADAGE)*

- ***Algorithmic Thinking***
- ***Decomposition***
- ***Abstraction***
- ***Generalisation***
- ***Evaluation***

Five Control Structures of Programming

- ***Sequence***
- ***Repetition***
- ***Functions***
- ***Decision***
- ***Communication***

The Five Data Structures of Programming

- ***Simple Data Types***
- ***Strings***
- ***Lists and related structures***
- ***Files***
- ***Database***

***An adage is a short, usually philosophical, but memorable *saying*.

MODULE 1 KS1 – KS3

This Module is part of a Training Course in Computer Science for Level 1 Master Teachers, sponsored by Computing at School and the Network of Excellence. It is to be held at University College London commencing 16th September 2014.

MODULE 1. The pathways in Module 1, through cross-curricula topics constitute a complete introduction to the ***5 elements of computational thinking***, ***algorithms and elementary programming (5 control structures)***, via both the ***sequential processing*** and the event-driven ***paradigms of programming*** in Scratch and/or Python, and lead to a transition to Python 3. We use these pathways to explore some curriculum elements of mathematics from a computing-oriented perspective.

THE COURSE MODULE 1 ENGAGES WITH:

- Models of Learning and Teaching in Computing
- Computational Thinking (unplugged and at the screen)

- Transition from Unplugged UPL to Programming in Scratch and/or Python
- Algorithms and Constructs in Programming (5 Control Structures)
- Transition from Visual-based Languages to Text-based Languages using Scratch as a pseudocode
- Programming Languages: Scratch 2.0, Logo, Python 3 and UPL (unplugged programming language).

MODULE 1 OUTLINE

1. **Introducing Algorithms and Programming:** Driving Sprites/Turtles: Unplugged Programming: UPL and transition to Scratch 2.0/Python 3.
   Projects: arrays of squares and iconic lettering.
2. **Pathway 1:** Geometric Shapes: regular polygons and stars, rotation, and translation of shapes. Projects: generating unique patterns by rotation and translation
3. **Pathway 2**: Event-driven Programming: Project: to produce teaching and learning materials. Scratch and Python programming (to be completed).
4. **Pathway 3**: Algorithms and Strategy in a 'Guessing the Number' Game.
   Project: Algorithms for the computer to both host and 'play' simple games.
5. **Pathway 4**: Python's Interactive Shell: Arithmetic in the 21st Century – strings of numbers, numbers of strings, evaluation of linear arithmetic expressions, recursion
   Project: Algorithms to evaluate simple arithmetic expressions (to be completed).

MODULE 2: KS4 AND GCSE;

MODULE 2. We develop Algorithms and Programming to include: driving more cross-curricula topics, creating learning-software and learning materials and Apps, by programming involving the ***5 data structures (range of data types, strings, lists, files, databases***) and the event-driven processing paradigm of programming for Apps to GCSE level. (To be developed)

Further planned Courses include gui-based programming, concurrency and object-oriented programming to A level --- always in the context of Computer Science Pedagogy, Computational Thinking and Programming Expertise.

AIMS

To equip Master Teachers with the knowledge, skills and pedagogy to deliver CPD in Computing to Teachers. The material covered is directed broadly at the KS1/KS2/KS3 bands in schools in Module 1 and KS4/GCSE in Module 2.

THE COURSE MODULE 2 ENGAGES WITH:

1. Models of Learning and Teaching in Computing

2. Computational Thinking (unplugged and at the screen)
3. Programming Language: Python 3
4. 5 Data Structures
5. A look at the Object-Oriented Paradigm of Programming

In short, it's an attempt to examine in theory and practice: why we learn and teach computing, what we learn and teach, and how we learn and teach it.

STARTING POINT

This is a component of a Course for CAS Master Teachers Level 1. The knowledge and experience of Computing in this cohort of 10 level 1 Master Teachers (5 Primary and 5 Secondary) varies appreciably. Consequently, no knowledge is assumed of teaching programming or of the programming languages involved in the Course. The main thrust of this module is to develop quality "Teaching and Learning of Computational Thinking and the Fundamental Principles of Programming" to achieve a basic level of practical competence in developing algorithms and programming.

OBJECTIVES FOR MODULE 1

During this part of the Course Master Teachers will have

1. addressed and discussed their model of learning and teaching with regard to computing
2. delivered CPD sessions in computing on the Course, with self, peer and tutor appraisal
3. undertaken and undergone a self and peer appraisal of a CPD session outside the course.
4. developed an understanding of what learning and teaching 'programming' means in CPD and the classroom
5. gained an informed overview of Scratch 2.0 and Python 3/Logo and the transition process in the context of KS2/KS3 bands
6. a working knowledge of computational thinking
7. an understanding of the 5 control structures in programming
8. experienced the relevance of unplugged sessions in learning programming and background material.
9. undertaken at least practical elementary programming in Scratch 2.0, Python 3 and UPL(Logo)
10. understand the difference between paradigms of programming e.g. procedural, event-driven and object-oriented
11. undertaken a programming project to help with their teaching in Computing.

HOW TO USE WORKBOOKS AND WORKSHEETS

ON THIS COURSE

- We intend to strike up an ongoing conversation with you in your roles as a Master Teacher and Teacher. Comments addressed directly to you as a teacher in Workbooks are marked by the diamond bullet point shown at the head of this paragraph.
  Workbooks and worksheets conform to a logical content design and ordered as a developmental Course in Algorithms and Programming. They are not divided up into lessons or to time slots. Worksheets are designed to provide explanatory material specifically for the graded Activities that follow.
  The Workbooks are not intended as the only learning resource. On the course, they are supported/reinforced with teaching, tutorials, coaching, supervision and the use of tailored teaching software e.g. codeinthebrowser.org, various versions of power.py and up1.py.
  We are attempting to use these Workbooks for master teachers, teachers and through to pupils. We strive to be constructionist rather than instructionist, ask questions to challenge and support, set experiments for trial and error, and give hints and analogies rather than solutions to copy.

IN YOUR ROLE AS MASTER TEACHER/TEACHER

- **Workbooks** are not everything. Teaching sessions, before during and after help to reinforce the learning. Unplugged material gives some space to think and plan. Use up1.py as a halfway house to programming on screen, and power.py and other software on the overhead projector to explain concepts and details; and demonstrate how to use the Scratch and/or Python programming environments and the tenets of good programming with snippets in Scratch and Python.
  Worksheets Some worksheets can be distributed directly to a class. The grading 0-5* gives some idea of degree of difficulty for Activities. It is for you to decide what is appropriate for the class you are teaching (teachers or pupils of differing experience) how you would change the material, what you retain, adapt, leave out and add to the ACTIVITIES to suit your purposes as a master teacher and teacher. For example, some instructions/questions addressed to you as a MasterTeacher/Teacher under a diamond bullet point, may also be useful directly for your class. You decide.
  Feedback on the effectiveness of the Workbooks and Worksheets is welcome. Further HINTS and SOLUTIONS to ACTIVITIES will be made available on http://www.ispython.com --- perhaps as a consequence of the work you undertake.
  Whenever possible get your class (whether it be teachers or pupils) operating in pairs/groups, right from the start to promote group cohesion, collaboration and networking.
  We value your responses as a reflective practitioner. For us, this is a pilot course in training Master Teachers, so we will be undertaking a post course review and undertaking an appraisal of the course. But your interaction with us as we go will be much appreciated.

ACTIVITIES ARE GRADED. (0-5*)

- In order to have tracked the Course and achieved the objectives set out, you should complete all Activities without a star and those marked with one *. There are a number of activities with a higher ** -- ***** rating, some of which we suggest you undertake in order to develop your expertise. Some parts of the Course can be omitted at a first reading

to learn programming. They are there for further practice and to show how programming can be used to explore cross-curricula mathematics (topics) in a uniquely illustrative way. Course Contents are written under a non-commercial creative commons license. The materials therefore will be offered to you in Word format online to use in that spirit to adapt for your learning and teaching purposes.
We ask you to cast a critical eye over the materials, as to whether they achieve what we have proposed, and how they could be improved and developed for Master Teachers and Teachers to follow (let us know).

SUMMARY OF WORKBOOKS FOR MODULE 1

UCL/CAS Training Course for Master Teachers. V5.0 3/02/2015

Workbook 1 v5.3 Algorithms and Programming 20/02/2015

Workbook 2 v5.0 Learning and Teaching Models in Computing 2/02/2015

Workbook 3 v5.7 Programming Unplugged – Cracking the Code 23/02/2015

Workbook 4 v4.4 Programming Unplugged -- Capital Letters 2/02/2015

Workbook 5 v4.4 Programming Unplugged – Arrays of Squares 2/02/2015

Workbook 6 v5.0 Action Geometry -- Joining the Dots 2/02/2015

Workbook 7 v5.0 Action Geometry -- Square and Pentagon 2/02/2015

Workbook 8 v5.0 Scratch: A Way to Logo and Python 2/02/2015

Workbook 9 v5.0 Stars in the Galaxy 2/02/2015

Workbook 10 v5.0 Rotation and Translation Patterns in Scratch and Python 2/02/2015

Workbook 11 v5.0 Unique Patterns in Scratch and Python 2/02/2015

Workbook 12 Event-Driven Programming in Scratch and Python (to be completed)

Workbook 13 v5.0 Appendix 1 Starting Up with Scratch 2.0 2/02/2015

Workbook 14 v5.0 A Number Guessing Game 2/02/2015

Workbook 15 Arithmetic in 21st Century. The Interactive Shell. Introduction to recursion (to be completed).