OCR GCSE Computer Science Curriculum Progression Map – Year 11

Paper Two: Computational thinking, algorithms and programming

Computer Science Education Vision at ELA:	At East Leake Academy we believe that our students should have the opportunity to follow a Computing curriculum that prepares them for life in modern Britain and take advantage of opportunity this can offer them in both Britain and the wider world. Good quality IT skills enable student to engage positively within the modern work place, while Computer Science skills enables students to take an active part in the design, development and creation of new technologies to be used in the world in which they live. The core to the subject is the understanding of how technology works, can be developed and utilised, and we draw and extend understanding from a range of other subjects outside of IT and Computing including DT, Graphics, Maths, Science and PHSE and embed clear and high quality literacy and numeracy skills through software development, problem solving and evaluation skills. We provide a broad range of skills and experiences at KS3 which are then further developed as students enter KS4 and then extended to KS5.
Paper 2: **Comp thinking, algorithms and programming**	Students apply knowledge and understanding gained in component 01. They develop skills and understanding in computational thinking: algorithms, programming techniques, producing robust programs, computational logic and translators.

Time period	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1
Key Domains of Knowledge	**2.1 Algorithms**	**2.2 Programming fundamentals**	**2.3 Producing robust programs** **2.4 Boolean Logic**	**2.5 Programming languages and IDEs** **Revision**	**Revision**
Topics	1. **Computational thinking** 2. **Designing, creating and refining algorithms** 3. **Searching and sorting algorithms**	1. **Programming fundamentals** 2. **Data types** 3. **Additional programming techniques**	1. **Defensive design** 2. **Testing** 3. **Boolean logic**	1. **Languages** 2. **The Integrated Development Environment (IDE)**	

| **Crucial learning content** | a) Principles of computational thinking:
- Abstraction
- Decomposition
- Algorithmic thinking

b) Identify the inputs, processes and outputs for a problem

c) Structure diagrams

d) Create, interpret, correct, complete and refine algorithms using:
- Pseudocode
- Flowcharts
- High level programming language

e) Identify common errors

f) Trace tables | a) The use of variables, constants, operators, inputs, outputs and assignments

b) The use of three basic programming constructs:
- Sequence
- Selection
- iteration

c) Common arithmetic and Boolean operators

d) The use of data types

e) String manipulation

f) Basic file handling ops

g) Use of SQL to search data

h) Arrays

i) Functions & procedures | a) Defensive design considerations

b) Input validation

c) Maintainability

d) Purpose of testing

e) Types of testing

f) Syntax and logic errors

g) Normal, boundary, invalid test data

h) Simple logic diagrams

i) Truth tables

j) Combining logic operators | a) Characteristics and purpose of different levels of programming languages:
- High level
- Low level

b) Translators

c) Compilers and interpreters

d) Common tools and facilities found in an IDE:
- Editors
- Error diagnostics
- Run time environment
- Translators | |
| **Programming** | Students are to be given the opportunity to undertake a programming task(s) during their course of study which allows them to develop their skills to design, write, test and refine programs using a high-level programming language. Students will be assessed on these skills during the written examinations, in particular component 02 (section B).

Students will spend one lesson per week programming in Python. Over the year they will work through the following concepts:

- Sequence, selection and iteration
- Data types
- Functions and procedures
- Validation | | | | |

Examination skills	AO1 Demonstrate knowledge and understanding of the key concepts and principles of Computer Science AO2 Apply knowledge and understanding of key concepts and principles of Computer Science AO3 Analyse problems in computational terms to make reasoned judgements and to design, program, evaluate and refine solutions				
End Point	By the end of Autumn Term 1 students should be able to apply their knowledge of coding from Year 10 to interpret written algorithms. They should also be able to apply to specified searching or sorting algorithm to a problem. They should be able to understand the terms used to describe their stages of thinking when trying to write a program for a given problem.	By the end of Autumn Term 2 students should be able to apply their knowledge of coding from Year 10 to answer questions on the tree programming constructs. Students will understand the concept and role of databases. They will be able to conduct basic searches using SQL.	By the end of Spring Term 1 students should understand how they can design programs that provide relevant validation and maintainability. They will also practice testing their programs to demonstrate they are fit for purpose. Boolean logic will be revisited and students should be able to complete truth tables, write Boolean expressions and draw logic gates to two levels.	By the end of Spring Term 2 students should be identify high and low level languages, the requirements of each and understand their role. They should also appreciate how IDEs can help them code. Students should need to be able to identify gaps in their knowledge so these can be addressed.	