# Curriculum Map – Computer Science – Year 11

Immanuel College
Church of England Academy

| | Half term 1 | Half term 2 | Half term 3 | Half term 4 | Half term 5 | Half term 6 |
|---|---|---|---|---|---|---|
| **Key focus** | Programming Fundamentals | Programming Fundamentals | Paper 1 Revision<br>Paper 2 Revision | Exam Practice | Exam | |
| **Intent** | To equip students with a solid understanding of programming fundamentals, enabling them to develop, test, and refine code effectively. This scheme of work aims to build essential programming skills that will empower students to solve real-world problems through structured, logical thinking and coding practices. | To equip students with a solid understanding of programming fundamentals, enabling them to develop, test, and refine code effectively. This scheme of work aims to build essential programming skills that will empower students to solve real-world problems through structured, logical thinking and coding practices. | To revise content covered across the course, making use of exam style questions and retrieval practice | Paper 1 and Paper 2 practice in preparation for the terminal exam | | |
| **Key knowledge and skills** | **Core Programming Concepts**:<br><br>Understanding of fundamental concepts, including variables, data types, and operators.<br><br>Introduction to control structures such as loops and conditional statements, enabling students to manage program flow.<br><br>**Problem Solving and Logical Thinking**:<br><br>Developing problem-solving skills by breaking down complex problems into manageable tasks.<br><br>Encouraging the use of pseudocode and flowcharts to plan code structure and logic.<br><br>**Algorithm Development**: | **Core Programming Concepts**:<br><br>Understanding of fundamental concepts, including variables, data types, and operators.<br><br>Introduction to control structures such as loops and conditional statements, enabling students to manage program flow.<br><br>**Problem Solving and Logical Thinking**:<br><br>Developing problem-solving skills by breaking down complex problems into manageable tasks.<br><br>Encouraging the use of pseudocode and flowcharts to plan code structure and logic.<br><br>**Algorithm Development**: | Everything covered in the course | Everything covered in the course | | |

“**Perseverance** produces **character,** and character, **hope**” (Romans 5:4)

| | | | | | | |
|---|---|---|---|---|---|---|
| | Introduction to algorithms as systematic methods for solving problems.<br><br>Exploration of basic algorithms, including searching and sorting, and their applications in coding.<br><br>**Data Structures**:<br><br>Familiarity with essential data structures, such as arrays and lists, and their appropriate usage in program design.<br><br>Understanding how data structures help manage and organize data efficiently.<br><br>**Error Handling and Debugging**:<br><br>Teaching strategies for identifying, diagnosing, and correcting errors in code.<br><br>Emphasis on debugging techniques to reinforce good programming habits.<br><br>**Testing and Evaluation**:<br><br>Applying testing methodologies, including desk checking, unit testing, and validation.<br><br>Developing skills in evaluating and optimizing code for efficiency and clarity.<br><br>**Practical Application and Creativity**:<br><br>Opportunities for students to apply programming knowledge through | Introduction to algorithms as systematic methods for solving problems.<br><br>Exploration of basic algorithms, including searching and sorting, and their applications in coding.<br><br>**Data Structures**:<br><br>Familiarity with essential data structures, such as arrays and lists, and their appropriate usage in program design.<br><br>Understanding how data structures help manage and organize data efficiently.<br><br>**Error Handling and Debugging**:<br><br>Teaching strategies for identifying, diagnosing, and correcting errors in code.<br><br>Emphasis on debugging techniques to reinforce good programming habits.<br><br>**Testing and Evaluation**:<br><br>Applying testing methodologies, including desk checking, unit testing, and validation.<br><br>Developing skills in evaluating and optimizing code for efficiency and clarity.<br><br>**Practical Application and Creativity**:<br><br>Opportunities for students to apply programming | | | | |

# Curriculum Map – Computer Science – Year 11

Immanuel College
Church of England Academy

| | | | | | | |
|---|---|---|---|---|---|---|
| | practical, hands-on projects.<br><br>Encouragement for students to create small applications that solve specific problems or address user needs. | knowledge through practical, hands-on projects.<br><br>Encouragement for students to create small applications that solve specific problems or address user needs. | | | | |
| **Key words/ vocabulary** | Variable, Data Type, Constant Operator, Expression, Statement, Sequence, Selection, Iteration, Loop, Conditional Statement, Function, Procedure, Parameter, Return Value, Array, List, String, Algorithm, Pseudocode, Flowchart, Syntax Error, Logic Error, Runtime Error, Debugging, Testing, Trace Table | Variable, Data Type, Constant Operator, Expression, Statement, Sequence, Selection, Iteration, Loop, Conditional Statement, Function, Procedure, Parameter, Return Value, Array, List, String, Algorithm, Pseudocode, Flowchart, Syntax Error, Logic Error, Runtime Error, Debugging, Testing, Trace Table | | | | |
| **Assessment method** | Summative assessment | Summative assessment | Final Exam | Final Exam | Final Exam | |
| **Wider links** | Maths, Science | Maths, Science | Maths, Science, Engineering, English, Sociology, Law | Maths, Science, Engineering, English, Sociology, Law | | |
| **Enrichment opportunities** | Cyberfirst | Programming competition | | | | |
| **Careers links** | Programmer, Program designer | Programmer, Program designer | Computer Scientist, Software developer/Engineer, data analyst, data scientist | Computer Scientist, Software developer/Engineer, data analyst, data scientist | | |