




# YEAR 13 A LEVEL COMPUTER SCIENCE SUMMER TERM 3 – PAPER 2

'An ambitious curriculum that meets the needs of all'

## Medium Term Planning – Computational Thinking

Curriculum Intent	Pupils will be taught the following National Curriculum guidelines this term:
<b>Skills/Assessment Objective Links</b>	<p><b>At the end of this Unit all students should be able to:</b></p> <ul style="list-style-type: none"> <li>▪ explain the differences between an abstraction and reality</li> <li>▪ describe the need for reusable program components</li> <li>▪ identify the inputs and outputs for a given situation</li> <li>▪ interpret simple algorithms to describe their purpose</li> <li>▪ give an example of how caching is used in a computer system</li> <li>▪ Identify the components of a problem</li> <li>▪ identify the points in a solution where a decision has to be taken</li> <li>▪ give an example of a Divide and Conquer algorithm</li> <li>▪ give examples of backtracking, heuristics, performance modelling and visualisation</li> </ul> <p><b>Most students will be able to:</b></p> <ul style="list-style-type: none"> <li>▪ determine the preconditions for devising a solution to a problem</li> <li>▪ describe the nature, benefits and drawbacks of caching</li> <li>▪ identify the components of a problem and its solution</li> <li>▪ determine the order of steps needed to solve a problem</li> <li>▪ determine the logical conditions that affect the outcome of a decision</li> <li>▪ determine how decisions affect flow through a program</li> <li>▪ identify sub-procedures needed to solve a problem</li> <li>▪ explain how a Divide and Conquer algorithm works</li> <li>▪ explain what is meant by backtracking, heuristics, performance modelling and visualisation</li> </ul> <p><b>Some students will be able to:</b></p> <ul style="list-style-type: none"> <li>▪ describe the nature of and need for abstraction</li> <li>▪ devise an abstract model for a variety of situations</li> <li>▪ design algorithms to solve complex problems</li> <li>▪ hand trace a complex algorithm to say what it does</li> <li>▪ determine the parts of a problem that can be executed concurrently</li> <li>▪ outline the benefits and trade-offs that might result from concurrent processing in a particular situation</li> <li>▪ apply techniques of backtracking, data mining, heuristics, performance modelling and visualisation to the solution of problems</li> </ul>
<b>Numeracy</b>	Abstraction and decomposition
<b>Literacy</b>	<p><b>Vocabulary Tier 3:</b> computational thinking, abstraction, algorithm, preconditions, greatest common divisor, modelling, simulation, preconditions, hierarchy charts, modularisation, trace table, concurrent processing, parallel processing, graph data structure, shortest path, exhaustive search, backtracking, data mining, Big Data, heuristics, rule of thumb, intractable problem, Travelling Salesman problem, performance modelling, pipelining, visualisation</p> <p><b>Vocabulary Tier 2:</b> shortest, data, graph, table, charts, model</p> <p><b>Reading:</b> Worksheets, presentations, answer sheets, exam questions, mark scheme, further reading for homework, conduct research for NEA</p> <p><b>Writing:</b> Answer on the worksheet via word, complete NEA</p> <p><b>Oracy:</b> listening and using tier 3 words</p>
<b>Becoming future ready</b>	<p><b>Careers/Employability:</b></p> <p>Understand the grade requirements at universities and the topics that can be applied for. Explore apprenticeship opportunities with a range of industries.</p> <ul style="list-style-type: none"> <li>▪ Software Architect.</li> <li>▪ Data Scientist.</li> <li>▪ Machine Learning Engineer.</li> <li>▪ Blockchain Developer</li> </ul>

	<ul style="list-style-type: none"> <li>▪ Cybersecurity Engineer.</li> <li>▪ Cloud Solutions Architect.</li> <li>▪ AI Research Scientist.</li> <li>▪ Full-Stack Developer.</li> </ul>
<b>Adaptation</b>	Throughout this topic, quality first teaching will provide differentiation:
<b>QFT/SEND Provision</b>	<p><b>By product:</b> Learners are asked to present outcomes writing different code, not all code will be equal in style and sophistication, all code will work with teachers input, top end programmers will be set challenges on how to extend code and be expected to conduct a level of independent research. Learners are asked to present outcomes in a different way via pieces of writing, targeted questioning, models and drawings and speaking.</p> <p><b>By resource:</b> Worksheets are well presented and accessible. Instructions are clearly outlined and separate from the information so that pupils know where to begin and end. Handouts are differentiated by outcome. Resources used will appeal to the range of preferred learning styles of pupils e.g. visual, auditory or kinesthetic learners. Scaffolding of tasks – word frames.</p> <p><b>By Intervention:</b> By providing different levels of supervision and support</p> <p><b>By Progressive Questioning:</b> Exploring pupils’ understanding through interactive dialogue using Blooms Taxonomy.</p> <p><b>By Grouping:</b> According to prior coding attainment, gender, social preference, preferred learning style.</p> <p><b>By Task:</b> Pupils identify targets which are meaningful via level of coding ability and feedback sheets</p> <p><b>By Offering Optional Activities:</b> In class or as homework, to extend learning.</p> <p>This QFT/SEND provision will be explicit within the lesson by lesson schemes of work.</p>
<b>Implementation Curriculum Delivery</b>	To be able to:
<b>Learning Outcomes (Knowledge)</b>	<p><b>Topic 1 Abstraction</b>  Understand the nature of and need for abstraction  Describe the differences between an abstraction and reality  Devise an abstract model for a variety of situations</p> <p><b>Topic 2 Reusable Programs</b>  Identify the inputs and outputs for a given situation  Determine the preconditions for devising a solution to a problem  Understand the need for reusable program components  Understand the nature, benefits and drawbacks of caching</p> <p><b>Topic 3 Thinking Procedurally</b>  Identify the components of a problem  Identify the components of a solution to a problem  Determine the order of the steps needed to solve a problem  Identify sub-procedures necessary to solve a problem</p> <p><b>Topic 4 Logical and Concurrent Thinking</b>  Identify the points where a decision has to be taken  Determine the logical conditions that affect the outcome of a decision  Determine how decisions affect program flow  Determine which parts of a program can be tackled at the same time  Determine the benefits and trade-offs that might result from concurrent processing in a particular situation</p> <p><b>Topic 5 Problem Recognition</b>  Explore different strategies for problem-solving  Understand the concept and application of the “divide and conquer” approach  Describe features that make a problem solvable by computational methods</p> <p><b>Topic 6 Problem solving</b>  Learn to solve problems by applying  o backtracking</p>

	<ul style="list-style-type: none"> <li>o heuristics</li> <li>o performance modelling</li> <li>o visualisation</li> </ul> <p>End of unit assessment</p> 
<b>Current learning to be developed in the future within:</b>	Links into understanding programming techniques, data structures and NEA
<b>Assessment</b>	See assessment maps for formative and summative assessment opportunities.
<b>Impact</b>	<p>Review assessment results and target pupils that require further support via:-</p> <ul style="list-style-type: none"> <li>• Learning conversation</li> <li>• Changing seating plan</li> <li>• Plan lessons to address areas of concern in assessment</li> <li>• Targeted homework based on low performance areas identified in the assessment and marked pieces</li> <li>• Stretch and challenge high ability pupils by identifying ambitious next steps to expand knowledge</li> </ul> <p>Create a feedback sheet for each student</p> <p>Each student identifies areas of Green, Amber and Red using Mark Assessment on their feedback sheet</p> <p>Complete NOW task on areas identified as Amber and Red</p>