Curriculum Information Year 11

Throughout year 11, alongside the theory topics outlined below students will develop their practical programming skills using a combination of teacher taught lessons Time2code and various online platforms. This is assessed through observations, discussions and screenshot evidence of work submitted into Google classroom.

Autumn Term

Unit title	Key Questions	Knowledge	Assessing Understanding
Algorithms and programming fundamentals	What are the principles of computational thinking? What is the purpose of decomposition and abstraction? What are structure diagrams? Why is computational thinking important? What is a linear and binary search and what are the benefits of each? What is a merge, insertion, bubble sort?	Know what is meant by computational thinking, decomposition and abstraction and provide examples How to produce a structure diagram in order to assist in decomposing a problem Know how to complete each of the search and sorting algorithms and be able to compare them. Know how to produce algorithms using flowcharts and code Application of Python programming knowledge to produce solutions to problems	***How understanding is assessed*** Completion of online workbooks, homework notes, online quizzes and smartrevise tasks - Peer, self and teacher assessment throughout the term of work produced. ***Skills*** Designing, creating, and refining algorithms using flowcharts and pseudocode; understanding abstraction and decomposition Implementing binary/linear search and bubble/merge/insertion sort algorithms Application of Python programming knowledge to produce solutions to problems

	What are the benefits of each of these sorts? What are the different types of errors that can occur when programming? How and why do programmers use a trace table?		Be able to produce/complete trace tables ***Assessment Point Information*** Exam questions on topic Past exam papers given for paper 1 revision prior to December mock
Producing robust programs	What issues should a programmer consider to ensure a program caters for all likely input values? What does code maintainability mean? What are the different types of errors that can occur in a program? What are the features of good testing strategy? What makes a robust program?	Know what is meant by the term “defensive design considerations” when writing programs. Understand why input validation is necessary. Know a range of validation techniques that can be used to write a robust program. Know and consider defensive design Understand some authentication techniques a programmer may choose to use to protect their program from misuse. Understand what programmers can do to make their code more readable. Understand how to refine algorithms in order to make them more robust.	***How understanding is assessed*** Completion of online workbooks, homework notes, online quizzes and smartrevise tasks - Peer, self and teacher assessment throughout the term of work produced. ***Skills*** Using variables, constants, operators (Boolean, arithmetic), and data structures (arrays). Testing techniques, error handling (syntax/logic errors), and producing maintainable code ***Assessment Point Information*** Exam questions on topic Past exam papers given for paper 1 revision

		Know reasons why a program should be tested. Know what iterative testing is. Know what final/terminal testing is. Know what a syntax error is. Know what a logic error is Understand that because a program works, it doesn't mean it works for all inputs. Understand that suitable test data for a program needs to include: Normal data Boundary data Invalid data Erroneous data	prior to December mock

Spring Term

Unit title	Key Questions	Knowledge	Assessing Understanding
Boolean Logic	What are the symbols used in logic diagrams?	Know how to make simple logic diagrams from Boolean expressions using AND, OR, NOT.	***How understanding is assessed*** Completion of online workbooks, homework notes, online quizzes and

	How do you complete a truth table? How do you create logic diagrams from truth tables?	Understand how to complete truth tables from one and two level logic diagrams. Understand how to create, complete or edit logic diagrams and truth tables for given scenarios.	smartrevise tasks - Peer, self and teacher assessment throughout the term of work produced. ***Skills*** Using logic gates (AND, OR, NOT) and truth tables. ***Assessment Point Information*** Exam questions on topic
Programming languages and IDES	What are the differences between high and low level languages? How do you write a program in assembly language? How does code a programmer writes become binary a computer can execute? Why do programmers use IDEs?	Know the characteristics of high level and low level programming languages. Understand the terms: Source code Assembly code Machine code Know what a translator does. Understand the differences between compilers and interpreters Know a range of facilities provided by an integrated development environment (IDE) to assist the programmer in writing code.	***How understanding is assessed*** Completion of online workbooks, homework notes, online quizzes and smartrevise tasks - Peer, self and teacher assessment throughout the term of work produced. ***Skills*** Understanding translators (compilers/interpreters) and Integrated Development Environments (IDEs) ***Assessment Point Information***

			Exam questions on topic

Summer Term

Unit title	Key Questions	Knowledge	Assessing Understanding
Revision	What topics do I as an individual and us as a class need to work on prior to the exams?	Dependent on the results of the December mocks, a schedule of revision topics to be covered will be drawn up and worked through with the class	***How understanding is assessed*** In class discussions Past exam questions and full paper practice Smart revise completion accuracy ***Skills*** Through dedicated revision and exam practice students develop their skills further in exam technique and gain experience in answering exam questions ready for their final exams in both papers ***Assessment Point Information*** Exam papers given out for homework and also completed under exam conditions in class