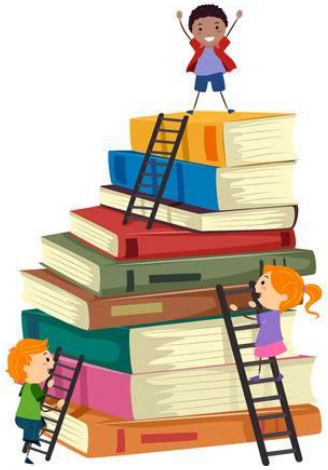
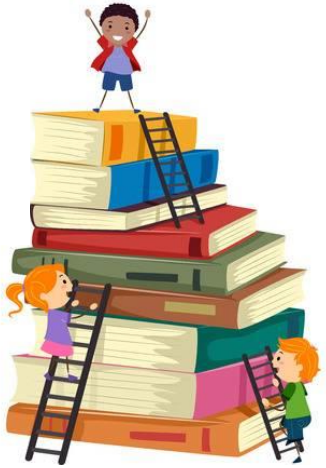
HALEWOOD ACADEMY

Computer Science



KEY STAGE 3
Y7 - Y9
ASSESSMENT STEPS

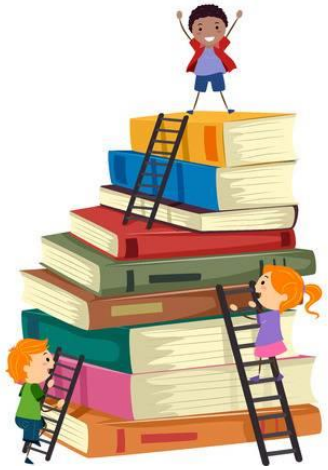


‘We Seek the Best’

AO	Emerging	Developing	Securing	Mastering
AO1Recall & Understanding	- Identifies basic computing terms (e.g. algorithm, input, output)- Recalls simple facts	- Describes key concepts with some accuracy- Understands basic functions of hardware/software	- Explains concepts in own words (e.g. binary, abstraction)- Uses correct vocabulary consistently	- Shows detailed understanding of advanced concepts (e.g. compression, logic gates)- Connects multiple topics
AO2Application & Skills	- Follows step-by-step instructions (e.g. copying code)- Uses software with guidance	- Modifies given code and can complete guided tasks (e.g. Scratch, Python basics)	- Designs solutions independently- Writes programs with selection and iteration	- Creates well-structured programs- Debugs and optimises code efficiently
AO3Evaluation & Analysis	- Identifies when something doesn’t work- Seeks help when prompted	- Can find and fix basic errors- Begins to explain why a solution works	- Tests code systematically- Reflects on how and why solutions are effective	- Compares different approaches- Suggests meaningful improvements

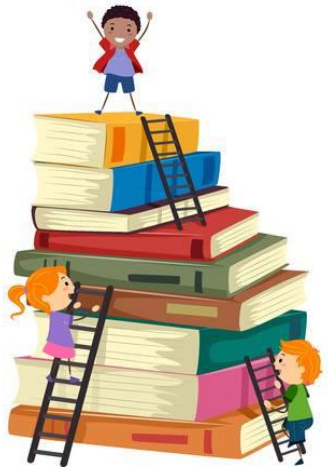
HALEWOOD ACADEMY

Computer Science



KEY STAGE 4

Y10 - Y11



ASSESSMENT STEPS

Topic	Assessment Objective	Key Skills / Student Outcomes	Example Tasks
1.1 Systems Architecture	AO1	Describe the purpose of the CPU; explain functions of ALU, CU, and registers	Label a CPU diagram; define registers like MAR/MDR
	AO2	Apply knowledge of the fetch-decode-execute cycle	Match steps to stages; sequence stages of FDE
	AO3	Analyse how processor performance is affected by cache, cores, clock speed	Evaluate upgrade options for a device
1.2 Memory and Storage	AO1	Identify types of memory (RAM, ROM, flash); define primary and secondary storage	Definitions quiz or short answers
	AO2	Apply knowledge to compare storage devices	Choose storage for a given scenario (e.g. SSD vs HDD)
	AO3	Justify storage decisions based on scenario	Explain the best storage solution for a music producer
1.3 Computer Networks	AO1	Explain types of networks (LAN, WAN); describe network topologies	Short written answers; identify LAN/WAN in context
	AO2	Apply understanding of protocols and hardware	Identify needed hardware for a small office setup
	AO3	Analyse network designs and evaluate improvements	Compare bus and star topology for efficiency
1.4 Network Security	AO1	Recall security threats (phishing, malware, etc.)	Define and identify different threats
	AO2	Apply security measures to scenarios	Recommend protection strategies for a business
	AO3	Evaluate effectiveness of security methods	Compare use of antivirus vs. penetration testing
1.5 Systems Software	AO1	Describe purpose and functions of OS and utility software	Define multitasking, drivers, file management
	AO2	Match utility tools to their functions	Choose utilities for specific tasks (e.g. backup)
	AO3	Evaluate use of utilities in different contexts	Analyse when defragmentation is useful or not
1.6 Ethical, Legal, Cultural, Environmental	AO1	Explain legislation (e.g. Data Protection Act, Copyright)	Multiple-choice or definition questions
	AO2	Apply ethical/legal issues to scenarios	Respond to a scenario involving data misuse
	AO3	Debate or evaluate social/environmental impact	Essay on the environmental impact of discarded devices

2.1 Algorithms	AO1	Define terms: input, process, output, decomposition, abstraction	Recall activity; define terminology
	AO2	Trace algorithms (flowcharts, pseudocode)	Determine output of search/sort algorithms
	AO3	Design algorithms for given problems	Create flowcharts or pseudocode for a program
2.2 Programming Fundamentals	AO1	Understand syntax and structure of programming concepts	Define variables, constants, data types
	AO2	Apply constructs (IF, loops, functions) in code	Complete partially written Python programs
	AO3	Write, test, and refine complete solutions	Develop a Python program from scratch
2.3 Producing Robust Programs	AO1	Understand validation, verification, testing types	Match terms to definitions
	AO2	Apply testing techniques (e.g. dry run)	Create a test plan for a program
	AO3	Debug and improve code	Identify and fix errors in a provided script
2.4 Boolean Logic	AO1	Recall logic gate symbols and truth tables	Draw or complete truth tables
	AO2	Apply logic to evaluate expressions	Solve logic circuit problems
	AO3	Design logic circuits from scenarios	Draw a circuit to meet set logical requirements
2.5 Programming Languages & IDEs	AO1	Understand high-level vs. low-level languages	Define examples and characteristics
	AO2	Apply knowledge to scenarios (e.g. compiler vs interpreter)	Choose appropriate tool for a use case
	AO3	Evaluate IDE features for development	Compare IDLE vs. other IDEs for student use