IGCSE Computer Science 0478 Intensive Lesson Plan (16-Week Framework)

Total Duration: 16 weeks

Target: Papers 1 & 2 (both compulsory) - Grades A*-G

Weekly Structure: 3 theory lessons + 2 practical programming sessions + 1 past paper review

Programming Language: Python recommended (or any high-level language)

Topic Weightings & Time Allocation

Topic Area	Paper	Weeks Allocated
Computer Systems (Topics 1-6)	Paper 1 (50%)	7 weeks
Algorithms & Programming (Topics 7-10)	Paper 2 (50%)	7 weeks
Revision & Mock Exams	Both	2 weeks

WEEK 1-3: DATA REPRESENTATION & TRANSMISSION (Paper 1)

Week 1: Number Systems & Text Representation

- Day 1: Binary, denary, hexadecimal conversions & arithmetic
- Day 2: Two's complement, representing negative integers
- Day 3: Text representation (ASCII, Unicode), character sets
- Day 4: **Practical**: Python program to convert between number bases
- Day 5: Past paper practice: Data representation questions (Paper 1)
- Day 6: Review common conversion errors & programming solutions

Week 2: Images, Sound & Data Compression

- Day 1: Bitmap images (resolution, colour depth, file size calculations)
- Day 2: Sound sampling (sample rate, bit depth, file size calculations)
- Day 3: Data compression (lossy vs lossless, RLE, Huffman coding)
- Day 4: **Practical**: Calculate image/sound file sizes using Python
- Day 5: Past paper practice: File size calculation questions
- Day 6: Review compression algorithms & their applications

Week 3: Data Transmission & Networking

- Day 1: Data transmission methods (serial/parallel, simplex/duplex)
- Day 2: Error detection & correction (parity bits, checksums)
- Day 3: Network types (LAN, WAN), topologies, protocols
- Day 4: **Practical**: Simulate parity bit generation in Python
- Day 5: Past paper practice: Data transmission & networking
- Day 6: Review network security concepts & exam technique

WEEK 4-6: HARDWARE, SOFTWARE & INTERNET (Paper 1)

Week 4: Computer Hardware

- Day 1: CPU architecture (ALU, CU, registers, cache), fetch-execute cycle
- Day 2: Memory types (RAM, ROM, secondary storage, virtual memory)
- Day 3: Input/output devices, storage devices & media
- Day 4: **Practical**: Research & present hardware specifications
- Day 5: Past paper practice: Hardware specification questions
- Day 6: Review fetch-execute cycle in detail

Week 5: Software & Operating Systems

- Day 1: System software vs application software
- Day 2: Operating system functions (file management, memory, I/O)
- Day 3: Types of software (proprietary, open-source, custom-written)
- Day 4: **Practical**: Compare OS features (Windows, Linux, macOS)
- Day 5: Past paper practice: Software comparison questions
- Day 6: Review utility programs & their functions

Week 6: Internet & Automated Technologies

- Day 1: Internet structure (IP addresses, DNS, routers, servers)
- Day 2: Security threats (malware, phishing, hacking) & protection
- Day 3: Automated systems, robotics, AI, emerging technologies
- Day 4: **Practical**: Investigate cybersecurity case studies
- Day 5: Past paper practice: Internet security scenario questions

- Day 6: Review automated technology applications

WEEK 7: BOOLEAN LOGIC & MID-TERM REVIEW (Paper 1)

Week 7: Boolean Logic & Paper 1 Consolidation

- Day 1: Logic gates (NOT, AND, OR, NAND, NOR, XOR), truth tables
- Day 2: Logic circuits, constructing from problem statements
- Day 3: Simplifying logic expressions, applying De Morgan's laws
- Day 4: **Practical**: Build logic gate simulator in Python
- Day 5: **Full Paper 1 mock** (1h45m) + detailed marking review
- Day 6: Review all Paper 1 topics, identify gaps

WEEK 8-11: ALGORITHMS & PROBLEM-SOLVING (Paper 2)

Week 8: Algorithm Design Fundamentals

- Day 1: What is an algorithm? Structure diagrams, top-down design
- Day 2: Flowchart symbols, creating flowcharts for given problems
- Day 3: Pseudocode syntax (declaration, assignment, input/output)
- Day 4: **Practical**: Convert flowcharts to pseudocode
- Day 5: Past paper practice: Algorithm design questions (Paper 2)
- Day 6: Review common pseudocode mistakes

Week 9: Control Structures

- Day 1: Selection (IF-THEN-ELSE, CASE statements)
- Day 2: Iteration (FOR loops, WHILE loops, REPEAT-UNTIL)
- Day 3: Nested loops, conditional logic
- Day 4: **Practical**: Write Python programs using all control structures
- Day 5: Past paper practice: Trace tables & dry runs
- Day 6: Review loop efficiency & common errors

Week 10: Data Structures & File Handling

- Day 1: Variables, constants, data types (integer, real, string, boolean)

- Day 2: Arrays (1D & 2D), records, file organization
- Day 3: Serial, sequential, random file access
- Day 4: **Practical**: Python programs with arrays & file I/O
- Day 5: Past paper practice: Data structure questions
- Day 6: Review file handling operations

Week 11: Validation, Verification & Error Handling

- Day 1: Validation techniques (range check, length check, format check)
- Day 2: Verification methods, error types (syntax, logic, runtime)
- Day 3: Robust programming, exception handling
- Day 4: **Practical**: Implement validation checks in Python
- Day 5: Past paper practice: Validation scenario questions
- Day 6: Review debugging techniques

WEEK 12-14: PROGRAMMING & DATABASES (Paper 2)

Week 12: Programming Concepts & Techniques

- Day 1: Subroutines (procedures & functions), parameter passing
- Day 2: Parameter passing by value vs by reference
- Day 3: Local & global variables, scope, modular programming
- Day 4: **Practical**: Create modular Python program with functions
- Day 5: Past paper practice: Subroutine questions
- Day 6: Review parameter passing with trace tables

Week 13: Searching & Sorting Algorithms

- Day 1: Linear search, binary search (how they work, comparisons)
- Day 2: Bubble sort, selection sort (algorithm trace)
- Day 3: Efficiency of algorithms (big O notation concept)
- Day 4: **Practical**: Implement search & sort algorithms in Python
- Day 5: Past paper practice: Algorithm comparison questions
- Day 6: Review efficiency comparisons

Week 14: Databases & SQL

- Day 1: Database structure (tables, records, fields, primary keys)
- Day 2: Relational databases, foreign keys, entity relationships
- Day 3: SQL queries (SELECT, FROM, WHERE, ORDER BY, aggregate functions)
- Day 4: **Practical**: Create database and write SQL queries
- Day 5: Past paper practice: Database design & SQL questions
- Day 6: **Full Paper 2 mock** (1h45m) + detailed marking review

---

WEEK 15: SCENARIO QUESTION MASTERCLASS

Week 15: Advanced Programming & Scenario Preparation

- Day 1: Scenario question structure analysis (15 marks)
- Day 2: Planning algorithms for unseen problems
- Day 3: Writing efficient pseudocode (30-40 lines)
- Day 4: **Practical**: 3 past scenario questions with timed practice
- Day 5: Peer review & improvement of pseudocode solutions
- Day 6: **Past paper marathon**: 5 scenario questions with model answers

---

WEEK 16: FINAL REVISION & EXAM PREPARATION

Week 16: Consolidation & Mock Exams

- Day 1: **Full Paper 1 mock** (full exam conditions)
- Day 2: **Full Paper 2 mock** (full exam conditions)
- Day 3: Detailed review of both papers with mark schemes
- Day 4: Topic-based revision: Focus on weakest 3 topics identified
- Day 5: Command words workshop, exam technique, time management
- Day 6: Light review, confidence building, final tips

Practical Programming Integration Strategy

1. **Daily coding**: 30 minutes of Python programming practice

2. **Project-based learning**: Mini-projects every 2 weeks (calculator, quiz game, database manager)
3. **Pseudocode first**: Always plan with pseudocode before coding
4. **Debugging practice**: Introduce intentional errors for students to fix
5. **Peer code review**: Students review each other's algorithms weekly

Key Resources:

- **Past papers**: Cambridge official (2019-2024)
- **Programming**: Python IDLE or online IDE (Replit)
- **Practice**: Smart Exam Resources topic questions
- **Theory**: Cambridge endorsed textbook (Hodder/Walsh)

Assessment Schedule: Bi-weekly topic tests, full mocks at weeks 7, 14, and 16.