GCSE Computer Science for OCR

Overview Scheme of Work

The following assumes a two-year model.

During the course, the final challenges can be used for practice in computational thinking, algorithm design and coding as well as maintaining a practical element during the study of the more theoretical aspects of the course.

They can also be used for introducing the requirements of the **Programming Project (J276/03/04)**: analysis, design, development, testing, evaluations and conclusions, and the use of appropriate programming techniques.

They can also be used to stress the items listed in the specification section **2.3** **Producing robust programs**:

* defensive design considerations:
  + input sanitisation/validation
  + planning for contingencies
  + anticipating misuse
  + authentication
* maintainability:
  + comments
  + indentation
* the purpose of testing
* types of testing:
  + iterative
  + final/terminal
* how to identify syntax and logic errors
* selecting and using suitable test data.

**In the following overview, suitable ‘final challenges’ are suggested to accompany the content.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **1 Algorithms** | 7 | Explain what an algorithm is and create algorithms to solve specific problems. | 2.1 Algorithms | Algorithmic thinking |
| Use sequence, selection and iteration in algorithms. | 2.2 Programming techniques | the use of the three basic programming constructs used to control the flow of a program: sequence, selection and iteration |
| Use input, processing and output in algorithms. | 2.2 Programming techniques | the use of variables, constants, operators, inputs, outputs and assignments |
| Express algorithms using flow charts and pseudocode. | 2.1 Algorithms | how to produce algorithms using pseudocode and flow diagrams |
| Analyse, assess and compare different algorithms. | 2.1 Algorithms | interpret, correct or complete algorithms |
| Create, name and use suitable variables. | 2.2 Programming techniques | the use of variables, constants, operators, inputs, outputs and assignments |

**1 Algorithms (continued)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Use arithmetic, relational and Boolean operators. | 2.2 Programming techniques  2.4 Computational logic | the common arithmetic operators; the common Boolean operators  apply computing related mathematics:  +, -, /, \*, exponentiation, MOD, DIV |
| Use conditional statements. | 2.2 Programming techniques | the use of the three basic programming constructs used to control the flow of a program: selection |
| **Final challenge:** | Create an algorithm to help a taxi company calculate its fares. | | | |
| **Additional Cambridge Elevate resources** | 10 interactive activities  6 worksheets  2 animations | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **2 Iteration** | 6 | Explain what is meant by iteration. | 2.2 Programming techniques | the use of the three basic programming constructs used to control the flow of a program: iteration (count and condition controlled loops) |
| Explain the difference between definite and indefinite iteration. |
| Use for loops. |
| Use while loops. |
| Use do … until loops. |
| Use nested loops. |
| Analyse algorithms using trace tables. | 2.1 Algorithms | interpret, correct or complete algorithms |
| Use iteration when designing algorithms. | 2.2 Programming techniques | the use of the three basic programming constructs used to control the flow of a program: iteration (count and condition controlled loops) |
| **Final challenge:** | Write an algorithm for a computer game. | | | |
| **Additional Cambridge Elevate resources** | 5 interactive activities  5 worksheets | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **3 Boolean logic** | 2 | Create truth tables for Boolean operators | 2.4 Computational logic | simple logic diagrams using the operations AND, OR and NOT  truth tables |
| Draw AND, OR and NOT logic gates | 2.4 Computational logic | simple logic diagrams using the operations AND, OR and NOT |
| Combine logic gates into logic circuits | 2.4 Computational logic | simple logic diagrams using the operations AND, OR and NOT  truth tables |
| Create truth tables for logic circuits. | 2.4 Computational logic | applying logical operators in appropriate truth tables to solve problems  combining Boolean operators using AND, OR and NOT to two levels |
| **Final challenge:** | Design logic circuits to solve a control problem. | | | |
| **Additional Cambridge Elevate resources** | 3 interactive activities  4 worksheets  1 animation | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **4 Data types and structures** | 3 | Explain what is meant by ‘data type’ and list some common types | 2.2 Programming techniques | the use of data types:  integer, real, Boolean, character and string  casting |
| Use the correct data types in algorithms |
| Carry out various manipulations such as finding the length of and slicing and concatenating ‘string’ data types | 2.2 Programming techniques | the use of basic string manipulation |
| Create and work with simple array data structures | 2.2 Programming techniques | the use of arrays (or equivalent) when solving problems, including both one and two dimensional arrays |
| Create and work with two dimensional arrays. |
| Describe other data structures | 2.2 Programming techniques | the use of records to store data  the use of SQL to search for data |
| **Final challenge:** | Encode and decode messages with an encryption key. | | | |
| **Additional Cambridge Elevate resources** | 5 interactive activities  4 worksheets  1 animation | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **5 Searching and sorting algorithms** | 4 | Explain why sorted lists are of more value than unsorted lists | 2.1 Algorithms | standard sorting algorithms:  bubble sort, merge sort, insertion sort |
| Describe the bubble sort, selection sort and merge sort algorithms |
| Use these algorithms to sort lists into ascending and descending order |
| Describe the linear and binary search algorithms | 2.1 Algorithms | standard searching algorithms:  binary search and linear search |
| Use these algorithms to search sorted and unsorted lists |
| Write code for the implementation of these algorithms. | 2.2 Programming techniques | All techniques mentioned above including one and two dimensional arrays |
| **Final challenge:** | Write an algorithm to find the top ten. | | | |
| **Additional Cambridge Elevate resources** | 4 interactive activities  5 worksheets  4 animations | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **6 Input and output** | 3 | Explain why user input is needed. | 2.3 Producing robust programs  2.6 Data representation | defensive design considerations:  input sanitisation/validation  planning for contingencies  anticipating misuse  authentication  The purpose of testing  The types of testing  Check digits |
| Describe ways in which data input can be validated. |
| Format output. |  | |
| Work with text files. | 2.2 Programming techniques | the use of basic file handling operations:  open, read, write, close |
| **Final challenge:** | Write a program to create and manage logins. | | | |
| **Additional Cambridge Elevate resources** | 4 interactive activities  4 worksheets  2 animations | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **7 Problem solving** | 4 | Explain what is meant by computational thinking | 2.1 Algorithms | computational thinking:  abstraction  decomposition |
| Explain what is meant by *decomposition* and *abstraction* and use them to solve problems |
| Create algorithms to solve problems that you have analysed | 2.3 Producing robust programs | All sections previously mentioned |
| Explain what is meant by top-down and bottom-up problem solving | 2.1 Algorithms | computational thinking:  abstraction  decomposition |
| Create structured programs using procedures | 2.2 Programming techniques | how to use sub programs (functions and procedures) to produce structured code |

**7 Problem solving (continued)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Follow the systems development cycle to analyse problems, design and implement solutions and test the outcomes. | 2.3 Producing robust programs  2.5 Translators and facilities of languages | maintainability: comments and indentation  types of testing:  iterative and final/terminal  how to identify syntax and logic errors  selecting and using suitable test data.  common tools and facilities available in an integrated development environment (IDE):  editors, error diagnostics,  run-time environment, translators |
| **Final challenge** | Write a program for ordering a pizza online. | | | |
| **Additional Cambridge Elevate resources** | 3 interactive activities  5 worksheets  3 animations | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **8 Binary and hexadecimal** | 5 | Explain how data is represented by computer systems. | 2.4 Computational logic  2.6 Data representation | why data is represented in computer systems in binary form  **Units**  bit, nibble, byte, kilobyte, megabyte, gigabyte, terabyte, petabyte  how data needs to be converted into a binary format to be processed by a computer. |
| Explain why the binary system is essential for computer processing. |
| Convert binary numbers into denary and vice versa. | 2.6 Data representation | how to convert positive denary whole numbers (0-255) into 8 bit binary numbers and vice versa |
| Carry out addition, subtraction, multiplication and division on binary numbers. | 2.6 Data representation | how to add two 8 bit binary integers and explain overflow errors which may occur |
| Use left and right shifts when multiplying or dividing binary numbers by powers of 2. | 2.6 Data representation | binary shifts |

**8 Binary and hexadecimal (continued)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Explain why hexadecimal numbers are used. | 2.6 Data representation | how to convert positive denary whole numbers (0-255) into 2 digit hexadecimal numbers and vice versa  how to convert from binary to hexadecimal equivalents and vice versa |
| Convert between binary, denary and hexadecimal. |
| **Final challenge** | Write a program that will convert between different number formats. | | | |
| **Additional Cambridge Elevate resources** | 4 interactive activities  5 worksheets  1 animation | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **9 Binary representations** | 4 | Explain how characters are represented in binary. | 2.6 Data representation | **Characters**  the use of binary codes to represent characters  the term ‘character-set’  the relationship between the number of bits per character in a character set and the number of characters which can be represented (for example ASCII, extended ASCII and Unicode) |
| Calculate the ASCII code for any character. |
| Calculate the size of a text file. |
| Explain how images are represented in binary. | 2.6 Data representation | **Images**  how an image is represented as a series of pixels represented in binary metadata included in the file  the effect of colour depth and resolution on the size of an image file |
| Calculate the size of an image file. |

**9 Binary representations (continued)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Explain how sound is represented in binary. | 2.6 Data representation | **Sound**  how sound can be sampled and stored in digital form  how sampling intervals and other factors affect the size of a sound file and the quality of its playback:  sample size  bit rate  sampling frequency |
| Calculate the size of an audio file. |
| Explain the disadvantages of large image and audio files. | 2.6 Data representation | **Compression**  need for compression  types of compression:  lossy and lossless |
| Explain how file compression reduces the size of files. |
| Explain the differences between lossless and lossy file compression. |
| **Final challenge** | Create a program to compress and decompress image files for a social media site. | | | |
| **Additional Cambridge Elevate resources** | 4 interactive activities  4 worksheets | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **10 Programming languages** | 2 | Describe the difference between low and high level languages. | 2.5 Translators and facilities of languages | Characteristics and purpose of different levels of programming language, including low-level languages |
| Explain the advantages of using high level languages. |
| Explain how program instructions are encoded in low level languages |
| Explain why high level languages need to be translated | 2.5 Translators and facilities of languages | the purpose of translators |
| Explain the characteristics and use of  • an assembler  • a compiler  • an interpreter. | 2.5 Translators and facilities of languages | the characteristics of an assembler, a compiler and an interpreter |
| **Final challenge** | Write programs using a low level language. | | | |
| **Additional Cambridge Elevate resources** | 1 interactive activity  3 worksheets  1 animation | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **11 Computer systems: hardware** | 4 | Explain what is meant by a computer system. | 1.2 Memory | the difference between RAM and ROM  the purpose of ROM in a computer system  the purpose of RAM in a computer system  the need for virtual memory  flash memory. |
| Explain what is meant by an embedded system. | 1.1 Systems architecture | embedded systems:  purpose of embedded systems  examples of embedded systems |
| Describe the structure of the central processing unit and the functions of its components. | 1.1 Systems architecture | the purpose of the CPU  Von Neumann architecture:  MAR (Memory Address Register)  MDR (Memory Data Register)  Program Counter  Accumulator  common CPU components and their function:  ALU (Arithmetic Logic Unit)  CU (Control Unit)  Cache |

**11 Computer systems: hardware (continued)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Describe the fetch-decode-execute cycle. | | 1.1 Systems architecture | the function of the CPU as fetch and execute instructions stored in memory | |
| Explain the need for and role of multiple cores and cache and virtual memory. | | 1.1 Systems architecture | how common characteristics of CPUs affect their performance:  clock speed  cache size  number of cores | |
| Describe secondary storage media and the advantages and disadvantages of each. | | 1.3 Storage | the need for secondary storage  data capacity and calculation of data capacity requirements  common types of storage:  optical, magnetic, solid state  suitable storage devices and storage media for a given application, and the advantages and disadvantages of these, using characteristics:  capacity, speed, portability, durability, reliability, cost | |
| **Final challenge** | Create a learning resource. | | | | | |
| **Additional Cambridge Elevate resources** | 4 interactive activities  5 worksheets  2 animations | | | | | |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | | | |
| **12 Computer systems: systems software** | 1 | Explain what is meant by systems software. | 1.7 Systems software | | | the purpose and functionality of systems software  operating systems:  user interface  memory management/ multitasking  peripheral management and drivers  user management  file management |
| Explain what is meant by an operating system. |
| Describe the functions of the operating system. |
| Explain what is meant by utility systems software. | 1.7 Systems software | | | utility system software:  encryption software  defragmentation  data compression  the role and methods of backup:  full  incremental |
| List some examples of utility systems software and their functions. |
| **Final challenge** | Create a program to clean up a hard disk drive. | | | | | |
| **Additional Cambridge Elevate resources** | 2 interactive activities  3 worksheets | | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Number of weeks** | **Learning outcomes** | **OCR Specification references** | |
| **13 Networks** | 5 | Explain what is meant by a computer network and list the different types  of networks. | 1.4 Wired and wireless networks | types of networks:  LAN (Local Area Network)  WAN (Wide Area Network) |
| Describe the differences between client server and peer-to-peer networks. | 1.4 Wired and wireless networks | the different roles of computers in a client-server and a peer-to-peer network |
| Explain the functions of the hardware needed to connect computers. | 1.4 Wired and wireless networks | the hardware needed to connect stand-alone computers into a Local Area Network:  wireless access points  routers/switches  NIC (Network Interface Controller/Card) |
| Explain how computers communicate using cable and microwave. | 1.4 Wired and wireless networks  1.5 Network topologies, protocols and layers | transmission media  factors that affect the performance of networks  Wifi:  frequency and channels  ethernet |
| Describe network topologies. | 1.5 Network topologies, protocols and layers | star and mesh network topologies |

**13 Networks (continued)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Explain how users connect to and use the internet. | 1.4 Wired and wireless networks | the internet as a worldwide collection of computer networks:  DNS (Domain Name Server); hosting; the cloud |
| Explain how data is transmitted across networks. | 1.5 Network topologies, protocols and layers | packet switching |
| Explain the use of protocols. | 1.5 Network topologies, protocols and layers | the uses of IP addressing, MAC addressing, and protocols including:  TCP/IP (Transmission Control Protocol/Internet Protocol); HTTP (Hyper Text Transfer Protocol); HTTPS (Hyper Text Transfer Protocol Secure); FTP (File Transfer Protocol); POP (Post Office Protocol); IMAP (Internet Message Access Protocol); SMTP (Simple Mail Transfer Protocol)  the concept of layers |
| Explain how virtual networks can be set up. | 1.4 Wired and wireless networks | the concept of virtual networks |
| **Final challenge** | Act as a consultant. | | | |
| **Additional Cambridge Elevate resources** | 5 interactive activities  5 worksheets | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Chapter** | | **Number of weeks** | | **Learning outcomes** | | **OCR Specification references** | | |
| **14 System security** | | 3 | | Describe the different strategies that criminals use to attack computer networks | | 1.6 System security | | threats posed to networks:  malware, phishing |
| Explain how people are the greatest security risks to networks | | 1.6 System security | | people as the ‘weak point’ in secure systems (social engineering) |
| Describe the threats posed to networks | | 1.6 System security | | brute force attacks  denial of service attacks  data interception and theft  the concept of SQL injection  poor network policy |
| Explain how these threats can be identified, prevented and combatted | | 1.6 System security | | Identifying and preventing vulnerabilities:  penetration testing; network forensics; anti-malware software; firewalls; user access levels; passwords; encryption |
| Explain the role of network policies. | | 1.6 System security | | network policies |
| **Final challenge** | | Design and code an information point. | | | | | | |
| **Additional Cambridge Elevate resources** | | 3 interactive activities  4 worksheets | | | | | | |
| **Chapter** | **Number of weeks** | | **Learning outcomes** | | **OCR Specification references** | | | |
| **15 Ethical, legal, cultural and environmental concerns** | 4 | | Investigate and discuss the following issues in relation to the development and impact of computer science technologies:  • environmental  • ethical  • legal  • cultural | | 1.8 Ethical, legal, cultural and environmental concerns | | how to investigate and discuss Computer Science technologies while considering:  ethical issues; legal issues; cultural issues; environmental issues  how key stakeholders are affected by technologies environmental impact of Computer Science cultural implications of Computer Science | |
| Discuss issues of data collection and privacy | | 1.8 Ethical, legal, cultural and environmental concerns | | privacy issues | |
| Describe the legislation relevant to computer science. | | 1.8 Ethical, legal, cultural and environmental concerns | | legislation relevant to Computer Science:  The Data Protection Act 1998; Computer Misuse Act 1990; Copyright Designs and Patents Act 1988; Creative Commons Licensing; Freedom of Information Act 2000; open source vs proprietary software | |
| **Final challenge** | Design and code an online test. | | | | | | | |
| **Additional Cambridge Elevate resources** | 2 interactive activities  4 worksheets | | | | | | | |