A Level Computer Science - Summer Project

Data types, data structures and algorithms

This is one of the first topics we cover as part of the A level Computer Science course. Below you can see the transition content for both KS4 and KS5.

Key Stage 4 Content

Key Stage 4 GCSE Content

2.1.2 Binary logic

Candidates should be able to:

d) explain why data is represented in computer systems in binary form
e) understand and produce simple logic diagrams using the operations NOT, AND and OR
f) produce a truth table from a given logic diagram.

2.1.4 Units

Candidates should be able to:

a) define the terms bit, nibble, byte, kilobyte, megabyte, gigabyte, terabyte
b) understand that data needs to be converted into a binary format to be processed by a computer.

2.1.4 Number

Candidates should be able to:

c) convert positive denary whole numbers (0-255) into 8-bit binary numbers and vice versa
d) add two 8-bit binary integers and explain overflow errors which may occur
e) convert positive denary whole numbers (0-255) into 2-digit hexadecimal numbers and vice versa
f) convert between binary and hexadecimal equivalents of the same number
g) explain the use of hexadecimal numbers to represent binary numbers.

2.1.4 Character

Candidates should be able to:

h) explain the use of binary codes to represent characters
i) explain the term character set
j) describe with examples (e.g. ASCII and Unicode) the relationship between the number of bits per character in a character set and the number of characters which can be represented.

Key Stage 5 Content

Key Stage 5 A Level Content

1.4 Data types, data structures and algorithms
How data is represented and stored within different structures. Different algorithms that can be applied to these structures

1.4.1 Data Types	a) Primitive data types, integer, real/floating point, character, string and Boolean. b) Represent positive integers in binary. c) Use of sign and magnitude and two's complement to represent negative numbers in binary. d) Addition and subtraction of binary integers. e) Represent positive integers in hexadecimal. f) Convert positive integers between binary hexadecimal and denary. g) Representation and normalisation of floating point numbers in binary. h) Floating point arithmetic, positive and negative numbers, addition and subtraction. i) Bitwise manipulation and masks: shifts, combining with AND, OR, and XOR. j) How character sets (ASCII and UNICODE) are used to represent text.
1.4.2 Data Structures	a) Arrays (of up to 3 dimensions), records, lists, tuples. b) The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table. c) How to create, traverse, add data to and remove data from the data structures mentioned above. *(NB: this can be **either** using arrays and procedural programming **or** an object-oriented approach.)*

Activity 1

Converting between denary, binary and hex

No.	Denary	Binary	Hex	Binary value plus 00011110
1	1			
2	5			
3	10			
4	22			
5	40			
6	77			
7	91			
8	121			
9	144			
10	168			
11	170			
12	200			
13	211			

Activity 2

Create a program that analyses a passage of text from a file and then counts:

- How many words
- The average length of a word
- How many times each word occurs
- How many words start with each letter of the alphabet?

The aim of this exercise is to test your ability to develop algorithms.

Activity 3

Complete one of the projects using Python from codeboom.

https://codeboom.wordpress.com/2012/07/30/10-mini-programming-projects/