

Revision List for Year 12 Computer Science

This a comprehensive list of everything you have studied on the course so far.

Component 1 Computer Systems

1.1.1 Structure and function of the processor

- (a) The Arithmetic and Logic Unit; ALU, Control Unit and Registers (Program Counter; PC, Accumulator; ACC, Memory Address Register; MAR, Memory Data Register; MDR, Current Instruction Register; CIR). Buses: data, address and control: how this relates to assembly language programs.
- (b) The Fetch-Decode-Execute Cycle; including its effects on registers.
- (c) The factors affecting the performance of the CPU: clock speed, number of cores, cache.
- (d) The use of pipelining in a processor to improve efficiency.
- (e) Von Neumann, Harvard and contemporary processor architecture.

1.1.2 Types of processor

- (a) The differences between and uses of CISC and RISC processors.
- (b) GPUs and their uses (including those not related to graphics).
- (c) Multicore and Parallel systems.

1.1.3 Input, Output and Storage

- (a) How different input, output and storage devices can be applied to the solution of different problems.
- (b) The uses of magnetic, flash and optical storage devices.
- (c) RAM and ROM.
- (d) Virtual storage.

1.2 Software and software development

1.2.1 Systems software

- (a) The need for, function and purpose of operating systems.
- (b) Memory Management (paging, segmentation and virtual memory).
- (c) Interrupts, the role of interrupts and Interrupt Service Routines (ISR), role within the Fetch-Decode-Execute Cycle.
- (d) Scheduling: round robin, first come first served, multi-level feedback queues, shortest job first and shortest remaining time.
- (e) Distributed, embedded, multi-tasking, multi-user and Real Time operating systems.
- (f) BIOS.
- (g) Device drivers.
- (h) Virtual machines, any instance where software is used to take on the function of a machine, including executing intermediate code or running an operating system within another.

1.2.2 Applications Generation

- (a) The nature of applications, justifying suitable applications for a specific purpose.
- (b) Utilities.
- (c) Open source vs closed source.
- (d) Translators: Interpreters, compilers and assemblers.
- (e) Stages of compilation (lexical analysis, syntax analysis, code generation and optimisation).
- (f) Linkers and loaders and use of libraries.

1.2.3 Software development

- (a) Understand the waterfall lifecycle, agile methodologies, extreme programming, the spiral model and rapid application development.
- (b) The relative merits and drawbacks of different methodologies and when they might be used. (c) Writing and following algorithms.

1.2.4 Types of programming language

- (a) Need for and characteristics of a variety of programming paradigms.
- (b) Procedural languages.
- (c) Assembly language (including following and writing simple programs with the Little Man Computer instruction set). See appendix 5d.
- (d) Modes of addressing memory (immediate, direct, indirect and indexed).
- (e) Object-oriented languages (see appendix 5d for pseudocode style) with an understanding of classes, objects, methods, attributes, inheritance, encapsulation and polymorphism.

1.3 Exchanging data

1.3.1 Compression, encryption and hashing

- (a) Lossy vs Lossless compression.
- (b) Run length encoding and dictionary coding for lossless compression.
- (c) Symmetric and asymmetric encryption.
- (d) Different uses of hashing.

1.3.2 Databases

- (a) Relational database, flat file, primary key, foreign key, secondary key, entity relationship modelling, normalisation and indexing. See appendix 5f.
- (b) Methods of capturing, selecting, managing and exchanging data.
- (c) Normalisation to 3NF.

(d) SQL – Interpret and modify. See appendix 5d.

(e) Referential integrity.

(f) Transaction processing, ACID (Atomicity, Consistency, Isolation, Durability), record locking and redundancy.

1.3.3 Networks

(a) Characteristics of networks and the importance of protocols and standards.

(b) The internet structure: • The TCP/IP Stack. • DNS • Protocol layering. • LANs and WANs. • Packet and circuit switching.

(c) Network security and threats, use of firewalls, proxies and encryption.

(d) Network hardware.

(e) Client-server and peer to peer.

1.3.4 Web technologies

(a) HTML, CSS and JavaScript. See appendix 5d.

(b) Search engine indexing.

(c) PageRank algorithm.

(d) Server and client-side processing.

Paper 2 – Computational Thinking, Algorithms and Programming

2.1.1 Computational Thinking

- ☐ Principles of computational thinking:
 - abstraction
 - decomposition
 - algorithmic thinking

2.1.2 Designing, Creating and Refining Algorithms

- ☐ Identify the inputs, processes and outputs for a problem
- ☐ Structure diagrams
- ☐ Create, interpret, correct, complete and refine algorithms using
 - pseudocode
 - flowcharts
 - reference language/high-level programming language
- ☐ Identify common errors
- ☐ Trace tables

2.1.3 Searching and sorting algorithms

- ☐ Standard searching algorithms:
 - binary search
 - linear search
- ☐ Standard sorting algorithms:
 - bubble sort
 - merge sort
 - insertion sort

2.2.1 Programming fundamentals

- ☐ The use of variables, constants, operators, inputs, outputs and assignments
- ☐ The use of the three basic programming constructs used to control the flow of a program:
 - sequence
 - selection
 - iteration (count and condition-controlled loops)
- ☐ The common arithmetic operators

- ☐ The common Boolean operators AND, OR, NOT.

2.2.2 Data types

- ☐ The use of data types:
 - integer
 - real
 - Boolean
 - character and string
 - casting

2.3.2 Testing

- ☐ The purpose of testing
- ☐ Types of testing:
 - iterative
 - final/terminal
- ☐ Identify syntax and logic errors
- ☐ Selecting and using suitable test data
 - normal
 - boundary
 - invalid/erroneous
- ☐ Refining algorithms

2.4.1 Boolean logic

- ☐ Simple logic diagrams using the operators AND, OR and NOT
- ☐ Truth tables
- ☐ Combining Boolean operators using AND, OR and NOT
- ☐ Applying logical operators in truth tables to solve problems

2.5.1 Languages

- ☐ Characteristics and purpose of different levels of programming language:
 - high-level languages
 - low-level languages
- ☐ The purpose of translators
- ☐ The characteristics of a compiler and an interpreter

2.5.2 The Integrated Development Environment (IDE)

- ❑ Common tools and facilities available in an IDE:
 - editors
 - error diagnostics
 - run-time environment
 - translators