Year 12 Computer Science – System Architecture and Processor Types

West Kirby Grammar School

Learning about this topic is important because: **Develops our understanding of the processor further, introduces the idea of different types of processors and the architecture of modern CPUs**	
This builds on: **System Architecture (Y10) / Memory and Storage (Y10)**	
This leads to: **N/A**	
We will learn: • **List the basic internal components of the processor: ALU, Control Unit, registers and buses** • **Name the different registers used in the Fetch-Decode-Execute cycle** • **List factors which affect the performance of the CPU: clock speed, number of cores, cache** • **Explain what is meant by a multicore system and parallel processing** • **Describe the uses of RAM, ROM and virtual storage** • **Describe typical uses of magnetic, flash and optical storage devices** • **Describe the Fetch-Decode-Execute cycle and its effect on registers** • **Describe the role of the data, address and control buses** • **Describe the use of pipelining in a processor to improve efficiency** • **Describe von Neumann and Harvard architectures and the advantages and use of each** • **Describe the characteristics and use of a Graphics Processing Unit (GPU)** • **Describe the benefits and potential problems of virtual storage** • **Describe the uses of magnetic, flash and optical storage devices** • **Describe how different I/O and storage devices can be applied to the solution of different problems** • **Describe features of contemporary processor architecture** • **Describe the differences between, and uses of, CISC and RISC architecture**	We will develop/practise skills including: **Coding skills – Assembly language (using LMC to simulate RISC systems)** **Research / analysis** **Comparison / evaluation**

Year 12 Computer Science – System Architecture and Processor Types

West Kirby Grammar School

• **Explain how system architecture and bus width relates to assembly language programs**	
Some of the vocabulary that we will use includes: **Arithmetic and Logic Unit (ALU), Control Unit, bus, registers, program counter (PC), accumulator (ACC), memory address register (MAR), memory data register (MDR), current instruction register (CIR), Fetch-Decode-Execute cycle, clock speed, core, cache, pipelining, von Neumann architecture, Harvard architecture, CISC, RISC, multicore, parallel systems, graphics processing unit (GPU), magnetic, flash and optical storage devices, RAM, ROM, virtual storage**	You could learn more about this topic by: **Craig n Dave – Structure and function of processor videos – Click here** **Crag n Dave – Types of processor videos – Click here** **Craig n Date – Input / Output / Storage videos – Click here** **Characteristics of contemporary processors – Click here**
Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are: SLR1/SLR2/SLR3 Key Homework Tasks End of Unit Assessment	

Year 12 Computer Science – Operating Systems and Application Generation

West Kirby Grammar School

Learning about this topic is important because:
Operating systems and system utility software are crucial for any computer system to function. This builds on the GCSE system software topic but also explores in depth what happens when written code is compiled or interpreted.

This builds on:
Year 11 – System Software

This leads to:
Year 13 – Exchanging Data (Autumn 2)

We will learn:

- **The need for, function and purposes of operating systems**
- **Memory management (paging, segmentation and virtual memory)**
- **The role of interrupts and Interrupt Service Routines (ISR) within the fetch-decode-execute cycle**
- **Scheduling: round robin, first come first served, multi-level feedback queues, shortest job first and shortest remaining time**
- **Distributed, embedded, multi-tasking, multi-user and real-time operating systems**
- **BIOS**
- **Device drivers**
- **Virtual Machines**
- **The nature of applications and justifying suitable applications for a specific purpose**
- **Utility Software**
- **Open vs Closed source software**
- **Translators: interpreters, compilers and assemblers**
- **Stages of compilation (lexical analysis, syntax analysis, code generation and optimisation)**
- **Linkers, loaders and libraries**

We will develop/practise skills including:

- **Comparison of the efficiency of algorithms (operating system scheduling) for different scenarios.**

Some of the vocabulary that we will use includes:

You could learn more about this topic by:

Year 12 Computer Science –
Operating Systems and Application Generation

Operating System, Memory Management, Paging, Segmentation **Virtual Memory, Interrupt, ISR, Scheduling, RRFCFS, MLFQ, SJF, SRT** **Distributed OS, Embedded OS, Multi-tasking OS, Multi-user OS** **Real Time OS, BIOS, Device Drivers, Virtual Machine**	**Craig n Dave – Application Generation – Click Here** **Craig n Dave – Operating Systems – Click Here** **Maths and Physics Tutor – Click Here**
Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are: SLR4/SLR5/SLR6 Key Homework Tasks End of Unit Assessment	

Year 12 Computer Science – Programming Techniques

West Kirby Grammar School

Learning about this topic is important because:
You will learn how to code in a programming language which is widely used in industry but also about programming concepts/techniques which can be applied to other languages. You will learn about more advanced techniques which are used in modern coding.

This builds on:
Programming Techniques (Y10) / Producing Robust Programs (Y11)

This leads to:
Data Structures / Algorithms (Y12) / Programming Project (Y12/Y13)

We will learn:

- **use an IDE to develop and debug a program**
- **describe the use of an IDE to check for syntax errors**
- **explain the difference between a variable and a constant**
- **write a pseudocode solution for a simple problem involving**
- **use nested selection and iteration statements**
- **use arithmetic operations and Boolean operations NOT, AND and OR**
- **use functions and library subroutines including random number generation**
- **know how to define and call a subroutine (procedure or function) with parameters**
- **construct algorithms using one and two-dimensional arrays**
- **define the OOP terms class, object, method, attribute, inheritance, encapsulation and polymorphism**
- **use local and global variables in subroutines**
- **trace through a recursive algorithm**
- **compare iterative and recursive algorithms for solving a problem**
- **complete given pseudocode for an object-oriented program**
- **write complex algorithms involving data structures, subroutines and file-handling**
- **distinguish between passing parameters by value and by reference**
- **write a recursive algorithm to solve a problem**
- **use object oriented programming techniques to solve problems**

We will develop/practise skills including:

- **Coding skills in Python**
- **Testing and debugging code**
- **Refining / optimising code**
- **Planning and coding algorithms for real world problem solving**

Year 12 Computer Science – Programming Techniques

Some of the vocabulary that we will use includes: **Integrated Development Environment (IDE), syntax errors, logic errors, debug, watch, breakpoint, trace algorithm, structured programming, data type, variables, constants, assignment, arithmetic operations, Boolean operators, sequence, selection, branching, definite and indefinite iteration, top down design, modular programming, subroutine, procedure, function, parameter, argument, pass by value, pass by reference, global and local variables, recursion, object oriented programming, class, object, method, attribute, inheritance, encapsulation, polymorphism**	You could learn more about this topic by: **Code Academy (online programming tutorials) – Click here** **Programming and Problem Solving – Click here** **Programming Techniques (Craig n Dave) – Click here** **Programming Paradigms (Craig n Dave) – Click here**

Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are:

SLR7/SLR14
Key Homework Tasks
End of Unit Assessment

Year 12 Computer Science – Data Structures and Algorithms

West Kirby Grammar School

Learning about this topic is important because:
So far in programming we have just used variables to store individual items of data and Python lists to store sets of data. Here we introduce more complex structures used to store sets of data.

This builds on:
Year 12 – Programming Techniques

This leads to:
Year 12 – Algorithms (Summer 1)

We will learn:

The following data structures:

- Arrays of up to three dimensions
- Records
- Lists
- Tuples
- The properties of stacks and queues
- Linked lists
- Graph (directed and undirected)
- Trees
- Binary Search Tree
- Hash table
- How to create, traverse, add data to and remove data from data structures,

We will develop/practise skills including:

- **Following, writing and refining algorithms.**
- **Writing and testing Python code.**
- **Comparing different data structures for specific purposes.**

Some of the vocabulary that we will use includes:

Array, Records, Lists, Tuple, Linked list, Directed Graph, Undirected Graph, Stack, Queue, Tree, Binary Search Tree, Hash Table

You could learn more about this topic by:

Craig n Dave – Data Structures – Click Here
Physics and Maths Tutor – Data Structures – Click Here
Code Academy (online programming tutorials) – Click here

Year 12 Computer Science –
Data Structures and Algorithms

Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are: **SLR14, Key Homework Tasks, End of Unit Assessment**

Year 12 Computer Science –
Computer Networks and Web Technologies

Learning about this topic is important because:
We will be learning about how the Internet works and also some of the main coding languages used to create web content. We will also learn about the background workings of some of the things we do online on a daily basis such as using search engines (and how the order pages appear on the results is determined).

This builds on:
Year 11 – Network Security

This leads to:
Year 13 – NEA Programming Project

We will learn:

- **Characteristics of networks and the importance of protocols and standards.**
- **Internet structure: The TCP/IP stack DNS: Protocol layering • LANs and WANs. Packet and Circuit switching.**
- **Network security and threats; use of firewalls, proxies, and encryption**
- **Network hardware.**
- **Client-server and peer-to-peer**
- **HTML, CSS and JavaScript**
- **Search engine indexing**
- **PageRank™ algorithm**
- **Server- and client-side processing**

We will develop/practise skills including:

- **Coding using HTML, CSS and JavaScript**
- **Calculations to determine the ranking of a web page using the PageRank Algorithm.**

Some of the vocabulary that we will use includes:

Protocol, TCP/IP Stack, DNS, Protocol Layering, LAN, WAN, Packet Switching, Circuit Switching, Firewall, Proxies, Encryption, Client-Server, Peer to Peer, HTML, CSS, JavaScript, Search Engine Indexing, PageRank Algorithm, Sever Side Processing, Client-Side Processing

You could learn more about this topic by:

Craig n Dave Networks Videos – Click Here
Craig n Dave Web Technologies Videos – Click Here
Physics and Maths tutor – Networks and Web Technologies – Click Here
W3Schools HTML/CSS/JavaScript online tutorials – Click Here

Year 12 Computer Science –
Computer Networks and Web Technologies

Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are: **SLR11/SLR12, Key Homework Tasks, End of Unit Assessment**

Year 12 Computer Science –
Algorithms – Searching, Sorting and Optimisation

West Kirby Grammar School

Learning about this topic is important because:

This builds on:
Year 12 – Data Structures (Spring 1)

This leads to:
Year 13 – NEA Programming Project

We will learn:

- **Analysis and design of algorithms for a given situation.**
- **The suitability of different algorithms for a given task and data set in terms of execution time and space.**
- **Measures and methods to determine the efficiency of different algorithms and Big O notation (constant, linear, polynomial, exponential and logarithmic complexity)**
- **Comparison of the complexity of algorithms**
- **Algorithms for the main data structures: stacks, queues, trees, linked lists, depth-first (post-order) and breadth-first traversal of trees**
- **Standard algorithms (bubble sort, insertion sort, merge sort, quick sort, Dijkstra's shortest path algorithm, A* algorithms, binary search and linear search)**

We will develop/practise skills including:

Some of the vocabulary that we will use includes:

Algorithm, Big O Notation, Bubble Sort, Insertion Sort, Merge Sort, Quick Sort, Dijkstra's Shortest Path, A* Algorithm, Binary Search, Linear Search

You could learn more about this topic by:

Craig n Dave – Algorithms – Click Here
Physics and Maths Tutor – Click Here
Optimisation Algorithm Visualiser – Click Here

Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are:
SLR14, Key Homework Tasks, End of Unit Assessment

Year 12 Computer Science –
Algorithms – Searching, Sorting and Optimisation

Year 12 Computer Science – Software Development Lifecycle

West Kirby Grammar School

Learning about this topic is important because:
We will learn about both traditional and more modern approaches to software development and compare/contrast a number of approaches. Understanding of the approaches, and knowledge of various testing strategies will also benefit you for the development and testing process of the NEA Programming Project.

This builds on:
Year 12 Programming Techniques

This leads to:
Year 13 NEA Programming Project

We will learn:

- **Understand the waterfall lifecycle, agile methodologies, extreme programming, the spiral model and rapid application development**
- **The relative merits and drawbacks of different methodologies and when they might be used**
- **Different test strategies including black- and white-box testing and alpha and beta testing**
- **Test programs that solve problems using suitable test data and end-user feedback; justify a test strategy for a given situation.**

We will develop/practise skills including:

- **Evaluating different approaches to the development lifecycle depending upon project requirements and choosing the most appropriate model.**

Some of the vocabulary that we will use includes:

SDLC, Waterfall Model, Agile Methodologies, Extreme Programming, Spiral Model, RAD

You could learn more about this topic by:

Craig n Dave – Software Development Methodologies – Click Here
Physics and Maths Tutor – Software Development – Click Here

Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are: **Homework Tasks, End of Topic Assessment**

Year 12 Computer Science – Computational Thinking and Methods

West Kirby Grammar School

Learning about this topic is important because:
In this topic, we learn about the theory of computational methods and also their applications for solving computational problems. Understanding of these methods will benefit you when you come to plan and design algorithms for your NEA Programming Project.

This builds on:
Year 11 - Algorithms

This leads to:
Year 13 – NEA Programming Project

We will learn:

- **The nature of abstraction and its use in computational models**
- **Identifying inputs and outputs for a given scenario**
- **Determining pre-conditions for devising a solution to a problem**
- **The benefits and drawbacks of caching**
- **The need for reusable components in a solution.**
- **Identify sub-procedures necessary to solve a problem**
- **Determine the parts of a problem that can be tackled at the same time**
- **Outline the benefits and trade-offs that might result from concurrent processing in a particular situation**
- **Problem recognition and decomposition**
- **Divide and conquer approaches**
- **The following computational methods: backtracking, data mining Heuristics, performance modelling, pipelining, visualisation.**

We will develop/practise skills including:

- **Problem solving skills such as abstraction, decomposition, backtracking, visualisation and data mining.**
- **Application of methods to particular problems to determine the best approach.**

Some of the vocabulary that we will use includes:

Abstract Model, System Inputs, System Outputs, Solution Preconditions, Computational Methods, Decomposition, Divide and Conquer, Concurrent Processing, Computational Methods, Backtracking, Data Mining, Heuristics, Performance Modelling, Visualisation

You could learn more about this topic by:

Craig n Dave Thinking Abstractly – Click Here
Craig n Dave Thinking Ahead – Click Here
Craig n Dave Thinking Procedurally – Click Here
Craig n Dave Thinking Logically – Click Here
Craig n Dave Thinking Concurrently – Click Here
Craig n Dave Computational Methods – Click Here

Year 12 Computer Science – Computational Thinking and Methods

Your teacher will assess your knowledge & understanding throughout the topic by looking at your work, questioning, discussion and giving you feedback in lots of different ways. The key pieces of work in this topic are: **SLR18-22/24 (combined?), Key Homework Tasks, End of Unit Assessment**	