Computing Key Stage 4

Long curriculum plan

OAK NATIONAL ACADEMY

1. Philosophy

Six underlying attributes at the heart of Oak's curriculum and lessons.

Lessons and units are **knowledge and vocabulary rich** so that pupils build on what they already know to develop powerful knowledge.

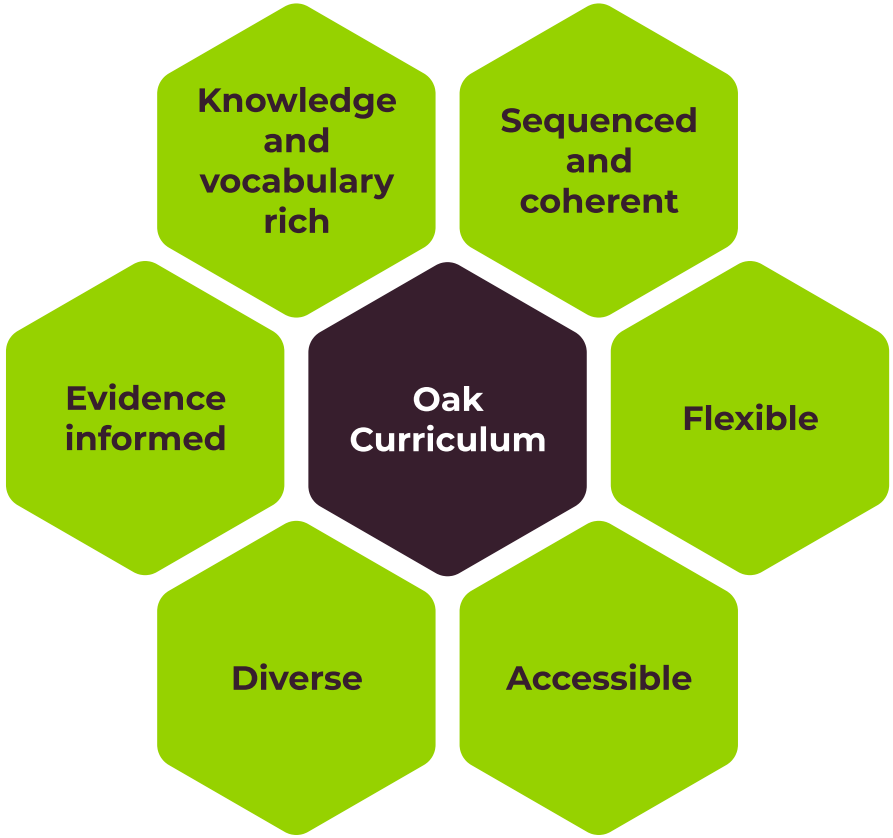
Knowledge is **sequenced** and mapped in a **coherent** format so that pupils make meaningful connections.

Our **flexible** curriculum enables schools to tailor Oak's content to their curriculum and context.

Our curriculum is **evidence informed** through rigorous application of best practice and the science of learning.

We prioritise creating a **diverse** curriculum by committing to diversity in teaching and teachers, and the language, texts and media we use, so all pupils feel positively represented.

Creating an **accessible** curriculum that addresses the needs of all pupils is achieved to accessibility guidelines and requirements.



2. Units

KS4 Computing is formed of 15 units and this is the recommended sequence:

Unit Title	Recommended year group	Number of lessons
1 Data Representation	Year 10	9
2 Computer Systems	Year 10	12
3 Networks	Year 10	6
4 Security	Year 10	7
5 Impacts on society	Year 10	7
6 Algorithms	Year 10	11
7 Programming 1: Sequence	Year 11	5
8 Programming 2: Selection	Year 11	6
9 Programming 3: Iteration	Year 11	5

10	Programming 4: Subroutines	Year 11	6
11	Programming 5: Strings and lists	Year 11	10
12	Programming 6: Dictionaries and datafiles	Year 11	12
13	Databases and SQL	Year 11	5
14	HTML	Year 11	6
15	Object-oriented programming	Year 11	5

3. Lessons

Unit 1 Data Representation — 9 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**What is representation?**	• In this lesson, we will consider two-state systems, such as bulbs and switches, and investigate how combining these in groups provides more combinations. Finally, we will learn how this relates to computers, and will be introduced to binary.
2.	**Number bases**	• In this lesson, we will use decimal representation to explore the concept of place values in number bases and the values of digits. Then, we will explore the representation of binary values, and converting these back to decimal.
3.	**Binary maths**	• In this lesson, we will discover how to perform binary shifts, binary addition and develop an understanding of the term 'overflow'.

4.	**Hexadecimal**	• In this lesson, you will be introduced to the hexadecimal number system and learn how to convert between hexadecimal and decimal numbers. We will also learn why hexadecimal is used.
5.	**What can you remember?**	• In this lesson, we will recap our previous learning on data representation within this unit to see how much we can remember. We will review number bases, binary maths and hexadecimal.
6.	**Representing text**	• In this lesson, we will look at character coding systems. First, we will consider character coding systems that you might have come across before, such as semaphore and Morse code, then we will be introduced to ASCII, a standard coding system for representing text. Next, we will learn about the need for Unicode to extend ASCII. Finally, we will learn how to work out the size of a plain text file.
7.	**Representing bitmap images**	• In this lesson, we will learn how bitmap images are represented in binary. We will learn about resolution, bit depth and how to calculate the file size of a bitmap image.

8. Representing sound

- In this lesson, we will look at sound and think about how it could be represented for storage or transmission. We will investigate the effect of sound waves in the air and how this relates to speakers and microphones. We will sample a short sound wave in order to understand this representation and how it can be improved on. We will also calculate representation size from sample rate, size, and duration.

9. Units of measurement

- In this lesson, we will look at units of measurement and learn how to convert between these units.

Unit 2 Computer Systems

12 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Computer systems and system software**	• In this lesson, we will discover two types of computer systems: general purpose and embedded. Following this, we will explore the need for system software to facilitate communication between software and hardware in computer systems. We will explain the role of an operating system in controlling a computer while it is running.
2.	**Introduction to the CPU**	• In this lesson, we will be introduced to the CPU and von Neumann architecture. We will learn about the individual components of the CPU and their roles in computation, along the way finding out about von Neumann and his theories that form the basis of modern computer architecture.
3.	**The FDE cycle**	• In this lesson, we will extend our knowledge of the components that make up the CPU by introducing the fetch-decode-execute cycle (FDE). We will observe a program running and will connect the parts of the CPU to their role in executing instructions.

4.	**Main memory**	• In this lesson, we will be introduced to main memory, RAM and ROM, as well as cache. This lesson builds on the core knowledge from the previous lesson about CPU components.
5.	**Secondary storage**	• In this lesson, we will be introduced to secondary storage and take an in-depth look at solid-state storage. We will discover the need for secondary storage, through assessing the devices we have learnt about already. By the end of the lesson, we will be able to explain how solid-state storage works, and describe the advantages and disadvantages of such devices.
6.	**Optical and magnetic storage**	• In this lesson, we will build on what we learnt in the previous 'Secondary Storage' lesson; this lesson involves exploring optical and magnetic storage devices. We will learn how each type of storage operates, and explain how data is written and read from each device. We will then actively rank the storage device in a number of key areas of comparison.
7.	**Selecting a storage device**	• In this lesson, we will systematically compare and select storage devices for a given situation. The second half of the lesson explores the limits of physical storage and how cloud storage can fill the gaps. We will examine cloud storage and explore questions about the impacts of cloud storage.

8.	**Computer specifications**	• In this lesson, we will learn how to evaluate a computer based on its specifications. We will discover the factors that limit a CPU's performance: clock speed, cache, and the number of cores. The end of this lesson will involve us choosing the right computer for a given task.
9.	**Logic gates**	• In this lesson, we will discover logic gates: the building blocks of processors at the heart of a computer system. Through the activities we will build an understanding of how logic gates are used to address real-world problems.
10.	**Logic problems**	• In this lesson, we will be introduced to the concept of three-input logic problems, and will be taught how to construct a three-input logic diagram, truth table, and expression.
11.	**Assembly language programming I**	• In this lesson, we will learn how to write and debug our own assembly language program. The lesson will build us up to this task, first modelling how to translate a piece of Python code into assembly language and examining the types of commands used in assembly language.

12. Assembly language programming II

- In this lesson, we will complete one last project for the unit: we will be given a set of requirements and tasked with designing an assembly language program to meet the requirements.

Unit 3 Networks

6 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**What are networks?**	• In this lesson, we will be introduced to computer networks and will learn about the different pieces of computer hardware involved in creating computer networks. We will then identify some of the advantages and disadvantages of using computer networks in the modern world.
2.	**Basic networks**	• In this lesson, we will be introduced to the ways that computers can be connected together in a network that is either wired or wireless. We will be introduced to MAC addresses and relate this to the need for having unique identifiers for every device on a network.
3.	**Real-world networks**	• In this lesson, we will further develop our understanding of networks. We will look at the three main types of networks, LAN, WAN, and PAN, and at the different topologies that these networks can use in their design.We will carry out an activity in which we will plan a business network to meet the requirements of a fictional company, 'Fabgadget Inc.'.

4.	**Configuring networks**	• In this lesson, we will compare peer-to-peer networks with client-server networks. We will also build and configure three networks using Packet Tracer.
5.	**The IP Suite and packet switching**	• In this lesson, we will get to know application protocols and the TCP/IP model. Finally, we will understand how data travels through a network.
6.	**Network speed and performance**	• In this lesson, we will look at how we measure network speed and performance, and what some of the key considerations are when designing networks, to ensure that they can function as well as possible. We will also learn about network broadcast traffic and how this affects network performance, and about the use of virtual LANs as a way to control the impact of broadcast traffic. Finally, we will look at the different ways and reasons that criminals attack computer networks, and what methods we have to defend ourselves against these attacks.

Unit 4 Security

7 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**The cost of cybercrime**	• In this lesson, we will explore the difference between cybersecurity and network security and understand that although networks are wonderful inventions, they can make an organisation vulnerable to attack. Through interactive activities we will gain a sense of the size of the problem and learn how important it is that we understand a hacker's motivation and take the subject seriously.
2.	**Non-automated cybercrime**	• In this lesson, we will distinguish between non-automated and automated cybercrime. We will be introduced to the idea that humans are the weakest link in the security chain before exploring different types of social engineering.
3.	**Automated cybercrime**	• In this lesson, we will explore terms and techniques related to automated cybercrime and the vulnerabilities of a network or software that can make companies or individuals vulnerable to such attacks.

4.	**Fighting fire with fire**	• In this lesson, we will discover different ways to protect software systems. This will include approaches to design and approaches to access.
5.	**Network defence**	• In this lesson, we will find out about ways to protect network systems from cybercrime. We will learn about the importance of network security before looking at a number of ways to achieve network security.
6.	**Where is the danger?**	• In this lesson, we will begin to understand how companies can and do test their own vulnerability to cybercrime.
7.	**Being part of the solution**	• In this lesson, we will explore different careers available in cybersecurity before taking an end of unit quiz.

Unit 5 Impacts on society

7 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**How does technology impact us?**	• In this lesson, we will be introduced to the five broad areas of impact (i.e. privacy, legal, ethical, environmental, and cultural). We will identify and discuss the impact areas through genuine examples. Finally, we will begin to understand the legal impact of technology specifically, and work through an example relating to the Data Protection Act (1998).
2.	**The law, data protection and copyright**	• In this lesson, we will further exercise our knowledge about data protection and clarify the definition of the word 'stakeholder' and examples of impact that a specific technology has had on them. We will then learn about the relationship between data protection, General Data Protection Regulation (GDPR), and the right to be forgotten. Finally, we will learn about copyright, Creative Commons licensing and comparing open source and proprietary software.

3.	**Freedom of Information and Computer Misuse**	• In this lesson, we will be introduced to the Freedom of Information Act, and given the opportunity to study genuine examples. We will also learn about the Computer Misuse Act and categorise case studies by the different levels of offence.
4.	**Cultural impacts**	• In this lesson, we will understand what the term 'cultural impact' means. We will learn about downtime and its effect on businesses and individuals and examine the concept of 'digital divide' and work through examples of its impact. Finally, we will analyse the cultural implications of the growing use of mobile technology and the responsibility of stakeholders in mitigating any risks explore globalisation.
5.	**Privacy and surveillance**	• In this lesson, we will discover the reality and limitations of privacy and surveillance. We will learn the laws that protect us directly, and others that enable the security services to protect us from harm. We will then think about the tension that exists between these two sets of laws. In addition, we will consider the different technology that we encounter in our daily lives and the privacy implications of these. Finally, we will learn about how intrusive social media is in its harvesting of data, and become more informed about the data that they are making available through our engagement with technology, as well as the, environmental impacts of social media.

6.	**Environmental impact**	• In this lesson, we will discover the reality of technology's impact on the environment and about how technology, if utilised strategically, can be used to protect the environment and can be a preserver rather than a destroyer.
7.	**Ethical impact**	• In this lesson, we will review what the word 'ethical' means generally. We will then be introduced to the main ethical impacts of technology, and engage in activities to help them become more aware of how important acting ethically is.

Unit 6 Algorithms

11 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Computational thinking**	• In this lesson, we will be introduced to three computational thinking techniques: decomposition, abstraction, and algorithmic thinking. We will explore how these skills can be applied when solving a wide range of problems, both computer-based and in their everyday lives.
2.	**Representing algorithms**	• In this lesson, we will be developing flow charts. This lesson assumes that learners have already covered the flow chart lesson in the KS4 Programming unit, although this lesson can also be used to introduce the flow chart symbols if required.
3.	**Tracing algorithms**	• In this lesson, we will be shown examples of tracing a Python program and a flow chart. Trace tables are great for walking through an algorithm and are often used to locate logic errors

4.	**Linear search**	• In this lesson, we will be introduced to one of two searching algorithms we need to know about: linear search. We will go over the steps of carrying out a linear search, and perform a linear search in real life and with a sample of data.
5.	**Binary search**	• In this lesson, we will be introduced to binary search: the second searching algorithm we need to know about. We will go over the steps of carrying out a binary search and perform a binary search with playing cards and with a sample of data.
6.	**Comparing searching algorithms**	• In this lesson, we will compare the features of linear search and binary search and the suitability of each algorithm in different contexts. We will also interpret the code of both algorithms in Python, as well as analysing the efficiency of two implementations of the linear search algorithm.
7.	**Bubble sort**	• In this lesson, we will learn the first sorting algorithm in this unit: bubble sort. We will discuss why and where sorting is used in real life, become familiar with performing a bubble sort on a set of data, and investigate the efficiency of bubble sort.

8.	**Insertion sort**	• In this lesson, we will explore another sorting algorithm: insertion sort. Some exam boards do not require learners to know insertion sort, so do check the specification first. We will start by discussing how to sort objects in real life, which will help us to describe something akin to an insertion sort.
9.	**Coding sorting algorithms**	• In this lesson, we will analyse and evaluate code for bubble sort and insertion sort in Python, as well as comparing different implementations of the bubble sort algorithm.
10.	**Merge sort**	• In this lesson, we will explore the final sorting algorithm in this unit: merge sort. We will start by considering how we might go about combining two groups of sorted items into one sorted group before being taken through the steps of one merge of a merge sort.
11.	**Algorithms review**	• In this lesson, we will practise and cement our knowledge of what we they have learnt in the algorithms unit. The worksheets contain a range of questions on flow charts, searching algorithms, and sorting algorithms that will help prepare us for the summative assessment for the unit.

Unit 7 Programming 1: Sequence

5 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Translators**	• In this lesson, we will learn that computers need clear and precise instructions in order to perform the expected task. We will also be taken on a journey from machine code to high-level languages in order to discover how a computer interprets instructions.
2.	**Sequence**	• In this lesson, we will be introduced to a Python IDE. We will learn about the function of an IDE and why programmers use these to write programs. We will be given some simple code to predict, run, investigate, and modify. Whilst we take our first steps in Python programming, we will also learn about common errors and error types.
3.	**Variables**	• In this lesson, we will find out about variables. We will learn about the purpose of variables, but also the technical aspects of creating variables to a uniform standard. Variable declaration is not used in Python, so a wider look at this through other programming languages will help us gain an insight into its meaning.

4.	**Input**	• In this lesson, we will start to add interactivity to our programs by using the input() function. Whilst learning about input, we will be introduced to functions and data validation techniques. These will be covered in more detail later on in the course.
5.	**Flowcharts**	• In this lesson, we will focus on interpreting and creating flowcharts. We will use our knowledge of writing simple sequences and subroutines to follow a flowchart, and write the code that it represents. We will be given time to write our own simple flowcharts in order to practise using the symbols that we have learnt during the lesson. This is an introduction to flowchart design and will be built upon throughout the unit.

Unit 8 Programming 2: Selection

6 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Randomisation**	• In this lesson, we will be introduced to the concept of random numbers using Python documentation. We will determine what the random module is capable of, and how random numbers can be generated in Python. In computer science, random numbers are something that you are likely to use regularly. They are also used in areas such as cryptography, while pseudorandom numbers are used in video games, modelling, and simulations.
2.	**Arithmetic expressions**	• In this lesson, we will understand the rules of operator precedence when evaluating arithmetic expressions. We will be reminded of BIDMAS, before investigating code that uses various arithmetic expressions. This lesson will prepare us for the next lesson, where we will begin to use conditions in programming.

3.	**Selection**	• In this lesson, we will move on to the next big programming construct: selection. We will be introduced to it initially through a flowchart that demonstrates how a condition can be used to control the flow of execution in a program. We will then learn about definitions for logical expressions and conditions. A short activity has been included to allow us to grasp how logical expressions evaluate. Next, we will complete a PRIMM activity where we investigate and modify a chatterbot.
4.	**Selection challenge**	• In this lesson, we will complete the 'make' part of PRIMM. We will complete an activity to create a joke machine. This will allow us to apply our new knowledge to a new, but similar scenario.
5.	**Logical expressions**	• In this lesson, we will deepen our understanding of logical expressions by introducing the operators AND and OR. It will begin with a Parson's Problem to check our understanding of selection. It will then move on to an unplugged activity that introduces AND and OR. We will walk through code, and investigate it before writing our own logical expressions.

6. Nested selection

- In this lesson, we will be introduced to the concept of nesting 'if statements'. We will walk through some basic nested statements to check our understanding. We will then follow the PRIMM approach and investigate a 'guess the animal' game. We will then modify the game to improve the functionality of it.

Unit 9 Programming 3: Iteration

5 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**While loops**	• In this lesson, we will investigate the world of iteration in programming. We will learn how to create and use a while loop in Python.
2.	**Trace tables**	• In this lesson, we will learn what a trace table is and how to use one. Trace tables help check logic errors in a program and help us learn how to read and follow programs.
3.	**For loops**	• In this lesson, we will be introduced to 'for loops'. For loops allow us to iterate through a sequence. We will find out how to create and use a for loop in this lesson.
4.	**Data validation**	• In this lesson, we will develop our understanding of data validation by applying data validation through iteration.

5. Pseudocode

- In this lesson, we will learn how to write our own pseudocode. We will also complete a challenging project that we will need to design using pseudocode before creating in Python.

Unit 10 Programming 4: Subroutines

6 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Subroutines**	• In this lesson, we will formalise our definition of a subroutine. We will find out why they are used in programming and we will also learn how to pass values to them.
2.	**Functions**	• In this lesson, we will explore how to create our own functions. Functions are a type of subroutine that allow us to return a value. We have used them before when we have used code like print() and input(). We will also explore the difference between a function and procedure.
3.	**Scope**	• In this lesson, we will learn about the scope of variables. When a variable is initialised within a subroutine it cannot be easily accessed and modified by other subroutines. This lesson will teach us about the different levels of scope and show us how to access variables that are initialised within subroutines.

4.	**XOR**	• In this lesson, we will be introduced to a new operator called XOR. We will discover how an XOR works before writing a function to perform the XOR operation.
5.	**Structured programming**	• In this lesson, we will be introduced to the paradigm: structured programming is programming. This lesson will also demonstrate how we can apply the paradigm to our own programs.
6.	**Create a program**	• In this lesson, we will follow the structured programming approach to create our own program.

Unit 11 Programming 5: Strings and lists

10 Lessons

Lesson number	Lesson question	About the lesson
1.	GUIs	**Pupils will learn** • In this lesson, we will learn how to create a program that uses a Guizero, which is a third party module that enables us to use Python code to create windows, buttons and more. We will also learn that GUI stands for Graphical User Interface.
2.	String handling I	**Pupils will learn** • In this lesson, we will learn about string handling techniques that can be used in Python.
3.	String handling II	**Pupils will learn** • In this lesson, we will further explore different string handling techniques that can be used in Python.
4.	String handling III	**Pupils will learn** • In this lesson, we will create a program that uses some of the string handling techniques that we have learnt in the lessons 'String handling I' and 'String handling II'.

5. Arrays and lists

Pupils will learn

- In this lesson, we will be introduced to the data structures: arrays and lists. We will then define them and explain the differences between the two. We will then focus on lists in Python. We will use lists to create a 'Simon says...' game, which randomly selects instructions from a list of items.

6. List methods

Pupils will learn

- In this lesson, we will explore some common list methods through the creation of a program that populates a deck of cards. We will also learn how lists can be returned from a function.

7. Sense HAT I

Pupils will learn

- In this lesson, we will learn about the world of the Sense HAT, a physical device that sits on top of a Raspberry Pi computer and can be used to sense the environment.

Guidance warnings

- Equipment requiring safe usage.

8. Sense HAT II

Pupils will learn

- In this lesson, we will continue to explore the Sense HAT and will complete 3 small projects that will demonstrate the different capabilities of the HAT.

9. 2D Arrays and lists

Pupils will learn

- In this lesson, we will learn about 2D arrays and lists. We will discover new syntax for creating and using them in our programs.

10. 2D Lists challenge

Pupils will learn

- In this lesson, we will use what we have learnt about 2D lists in previous lessons to complete a challenging project.

Unit 12 Programming 6: Dictionaries and datafiles

12 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Records and dictionaries**	• In this lesson, we will be introduced to two new data structures: a record and a dictionary. The focus of this lesson is on records and how these can be created and used in Python to form a database. We will be shown how to use a dictionary as a record before creating a 'database' using dictionaries within a list.
2.	**Dictionary challenge**	• In this lesson, we will have the opportunity to use a dictionary data structure in a new context. We will create a Caesar cipher encryption program using a dictionary as the cipher wheel. This mini project will allow us to develop our programming skills through an appropriate challenge.
3.	**Reading text files**	• In this lesson, we will be introduced to text files. The focus will be on reading text files, and how the data from a text file can be used within a program. We will be stepped through the key methods that are used for reading text files in Python, before we complete two text file challenges.

4.	**Writing to text files**	• In this lesson, we will continue to explore text files by looking at how to write text files and how to append text files. Live coding will be used to introduce the two new concepts and mini challenges are used to allow us to test our understanding of them.
5.	**Work with CSV files**	• In this lesson, we will learn what CSV files are and how to use them. We will also learn how to read data into a list and a 2D list. Finally, we will complete a challenge working with the data.
6.	**Write to CSV files**	• In this lesson, we will learn how to write to CSV files. We will work with 1D and 2D lists, before converting them to string and writing them to CSV files. There are lots of list challenges this lesson, which should help to enhance skills in preparation for the final project.
7.	**Being a programmer**	• In this lesson, we will discuss the good habits of a programmer before being reminded of why some of the key aspects are good habits. We will also hear from industry programmers about their own good practice. We will then look at alternative approaches to programming solutions.

8.	**Battle boats**	• In this lesson, we will be introduced to the final programming project of this course: programming a game called Battle Boats.
9.	**Battle boats design**	• In this lesson, we will design our Battle Boats game using either Pseudocode or Flowchart in preparation for programming the solution in the next lesson: 'Battle boats code'.
10.	**Battle boats code**	• In this lesson, we will code our Battle Boats game. Demonstrations and solutions are provided to give us support if needed, and it is likely to take more than an hour to complete the coding.
11.	**Battle boats test**	• In this lesson, we will focus on creating and using a test plan to test our Battle Boats program.
12.	**Battle boats evaluate**	• In this lesson, we will complete an evaluation of our Battle Boats program.

Unit 13 Databases and SQL

5 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Database essentials**	• In this lesson, we will explore the key terminology required to be able to use SQL to search and update a relational database. Together we will step through the key concepts using a music database as an example. We will complete the lesson by gaining some hands on experience of using a database management system.
2.	**SQL searches**	• In this lesson, we will use the music database from the previous lesson, 'Database essentials', to write our first SQL commands to access and manipulate the data.
3.	**Insert, update, delete**	• In this lesson, we will explore INSERT, UPDATE, and DELETE queries, before being given the opportunity to implement these queries using our music database.
4.	**Swim challenge (Part 1)**	• In this lesson, we will use our SQL skills to build, interrogate, and update a database to manage swimming lessons.

5. Swim challenge (Part 2)

- In this lesson, we will use our SQL skills to build, interrogate, and update a database to manage swimming lessons.

Unit 14 HTML

6 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Introduction to HTML**	• In this lesson, we will take our first footsteps in web development. We will start by gaining an understanding of what a website is and how it ends up in the web browser. Then we will edit content in a website before creating our very own web page from scratch.
2.	**Images and links**	• In this lesson, we will look at the design considerations necessary to make a website as accessible as possible. We will then add images and links to a wesbite.
3.	**Mini project**	• In this lesson, we will build upon a laser tag website by designing and adding new pages.
4.	**Introduction to CSS**	• In this lesson, we will focus on the look and feel of a laser tag website by applying CSS (a cascading style sheet) to it.

5.	**DIVs and classes**	• In this lesson, we will look at how we can make use of a DIV tag to apply CSS to different sections of each page on a website.
6.	**The CSS box model**	• In this lesson, we will learn how to apply the CSS Box Model to achieve the exact appearance we would like to on a webpage.

Unit 15 Object-oriented programming

5 Lessons

Lesson number	Lesson question	Pupils will learn
1.	**Programming Paradigms**	• In this lesson, we will focus on how a programmer chooses to write programs. We will investigate how they use a programming paradigm, which is a set of conventions, a set pattern or set way of doing things.
2.	**Classes and Objects**	• In this lesson, we will learn how to create a class in Python and then use that class to create objects.
3.	**Creating a class**	• In this lesson, we will use our experience with object-oriented programming to step into the shoes of a video game programmer. We will create a class that will hold the information about monsters in a puzzle game.
4.	**Inheritance**	• In this lesson, we will learn about a key principle of object-oriented programming - Inheritance. We will use it to create new monster classes for the Monster Quest puzzle game. These new classes will inherit all the attributes and methods of the original monster class and add some new ones all their own!

5. Exploring OOP

- In this lesson, we will explore a program written using OOP. We will use our experience to first investigate and then modify the program. Finally, we will add a new subclass to the program and apply the principle of inheritance.

4. Learn More

Contents

1. Coherence and flexibility

The computing curriculum is structured in units. For the units to be coherent, the lessons within them must be taught in order. However, the curriculum is flexible in terms of the order in which you teach units within a year group, except for programming, where concepts and skills rely on prior knowledge and experiences.

2. Knowledge organisation

The curriculum applies to the National Centre for Computing Education's computing taxonomy. This has been developed through a review of the KS1-4 computing programme of study, and the GCSE and A Level computer science specifications, across all awarding bodies. All learning outcomes can be described through a top-level taxonomy of ten topics, ordered alphabetically as follows:

- Algorithms
- Computer Networks
- Computer Systems
- Creating Media
- Data & Information
- Design & Development
- Effective use of tools
- Impact of technology
- Programming
- Safety & Security

The taxonomy categorises and organises content into strands which encapsulate the discipline. Whilst all strands are present at all phases, they are not always taught explicitly.

3. Inclusive and ambitious

We want Oak to be able to support all children. Our units will be pitched so that pupils with different starting points can access them. Our lessons will be sequenced so that each builds on prior learning. Our activities will be scaffolded so all children can succeed. We use unplugged or real world activities to unpack difficult concepts in computing as part of a semantic wave of learning. We also use a range of scaffolding approaches when teaching programming, ranging from copying code, exploring some commands or functions, fixing code with bugs to solving specific problems with code.

4. Application through software

We need pupils to be thinking during their lessons - both to engage with the subject and to strengthen memory of what is being learnt. Some of our lessons require practical application of concepts and skills on a computer using appropriate software. We supplement our lessons with guidance on how to use such software to reinforce the learning from the lesson.

5. Motivation through learning

We believe that computing is inherently interesting, and seek to motivate pupils through the subject matter. Where possible, we draw on real world experiences to provide an engaging viewpoint on computing concepts. Every pupil should have the opportunity to implement their skills and knowledge and ultimately feel a sense of achievement. We provide opportunities for pupils to be creative and solve problems by building their own programmes and applications for example.

6. Key stage 4 computing curriculum themes

Below are the key stage 4 GCSE Computing and Non-GCSE Computing units organised by 10 curriculum themes.

Algorithms

- Computing Systems (Year 10)
- Algorithms (Year 10)
- Programming (Year 11)

Programming

- Computing Systems (Year 10)
- Networks (Year 10)
- Algorithms (Year 10)
- Programming (Year 11)
- Object-oriented programming (Non-GCSE Computing course)
- HTML (Year 11)

- Databases and SQL (Year 11)

Data and Information

- Data Representation (Year 10)
- Computer Systems (Year 10)
- Databases and SQL (Year 11)
- Spreadsheets (Non-GCSE Computing course)

Computing Systems

- Computer Systems (Year 10)

Network

- Networks (Year 10)

Creating Media

- Project Management (Non-GCSE Computing course)
- Media (Non-GCSE Computing course)

Design and Development

- Programming (Year 11)
- Object-oreinted programming (Year 11)
- HTML (Year 11)

Effective use of tools

- Project management (Non-GCSE Computing course)
- Spreadsheets (Non-GCSE Computing course)

- IT and the world of work (Non-GCSE Computing course)
- Media (Non-GCSE Computing course)

Impact of Technology

- Impacts on society (Year 10)
- IT and the world of work (Non-GCSE Computing course)

Safety and Security

- Security (Year 10)
- IT and the world of work (Non-GCSE Computing course)

7. Key stage 4 computing unit summaries

Unit title	Summary of unit content
Data Representation	Binary, Hex Conversions & Ops Text Images & Sound Data Capacity Compression
Computer Systems	Components Architecture Storage Software Boolean logic
Networks	Components Classifications Protocols Layers
Security	Vulnerabilities Forms of Attack Techniques for: Identification and protection
Impacts	Ethical Legal Environmental (inc. privacy and cyber security)
Algorithms	Tracing & Exec. Representation Searching Sorting Efficiency

	Comp. Thinking
Programming	Tracing & Exec. Prog. constructs Data types, structs Modularity Quality Translators
Databases & SQL	Relational databases SQL (Select, Insert, Update, Delete)
HTML	Images, links, CSS (DIVS, classes and box model)
Object Oriented Programming	Classes, Objects, Attributes, Methods, Encapsulation, Inheritance, Structured programming, Software design