KS2 Progression in Computing
Iqra Primary School

CS	Year 3	Year 4	Year 5	Year 6
Problem solving	Design, write and debug programs that accomplish specific goals. Design and write a program using a block language, without user interaction. *A typical program might be a scripted animation for a joke, part of a story, or linked to another area of the curriculum. Programs could use pre-built sprites or ones designed by the child. Expect programs to include movement and dialogue; they may also include sound effects and some use of costumes to allow for animated movement. There may be more than one sprite in the animation.*	Design, write and debug programs that accomplish specific goals. Design and write a program using a block language to a given brief, including simple interaction. *Write a program in Scratch (or similar) in which the user has to provide some input, perhaps as an answer to a question on screen, or by using key presses or the mouse. The program could be a simple game or a set of questions and typed responses.*	Design, write and debug programs that accomplish specific goals. Design, write and debug a program using a block language based on their own ideas. *Can design a program of their own and write this in a block-based language such as Scratch. The child can test and debug their code, explain what bugs they found and how they fixed them. The program need not be complex (a simple game or a turtle graphics program would suffice) but it should be accomplished with a degree of independent working.*	Design, write and debug programs that accomplish specific goals. Design, write and debug a program using a second programming language based on their own ideas, using iterative development to make improvements. *Can design a program of their own and write this in a programming language other than Scratch (or whichever language has formed the focus for their programming in other years), such as TouchDevelop or App Inventor. The second language does not need to be text based, but Logo or Python could be used. The child can test and debug their code, explain what bugs they found and how they fixed these. The child can review their code, decide for themselves how this might be extended or improved, and then implement, test and debug these modifications. At this level, expect the child to be able to develop relatively complex apps with a degree of independence.*
	Controlling or simulating physical systems. Explore simulations of physical systems on screen. Experiment with some on-screen simulations of physical systems, *linked to topics from other curriculum areas, e.g. a ball bouncing on a bat or a car moving around a track.* Many computer games include elements of computer simulations. Discuss what they have learned from using the simulation.	Controlling or simulating physical systems. Develop their own simulation of a simple physical system on screen. *Create a Scratch (or similar) program to simulate a simple physical system. This could be in the form of a simple animation or an on-screen prototype for a product made in design and technology*	Controlling or simulating physical systems. Experiment with computer control applications. *Can use simple computer control and/or sensors with products they make in design and technology, perhaps using Lego WeDo kits, MaKey MaKey or similar.*	Controlling or simulating physical systems. Design, write and debug own computer control application, using iterative development to make improvements. *Can add computer control and/or sensors to a smartphone app or to products they design and make in design and technology, perhaps using Lego WeDo kits, MaKey MaKey or similar. The child can show evidence of designing, writing and debugging their program, ensuring that this functions correctly on the available hardware platform. The child can review their code and, perhaps, their hardware, decide for themselves how this might be extended or improved, and then implement, test and debug these modifications.*

	Solve problems by decomposing them into smaller parts To plan a project. Develop an outline plan for a project in computing, involving multiple steps and resources, *e.g. creating an animation, filming a video or conducting a survey. In video work, the plan might include identifying a subject; storyboarding the video; sourcing media; recording video; filming; editing; exporting.*	Solve problems by decomposing them into smaller parts Work with others to plan a project. Given a particular project, can work as part of a team to plan how to accomplish their goal, breaking the project down into a set of tasks. *Examples of projects could include creating an educational game, developing a wiki or monitoring the weather.*	Solve problems by decomposing them into smaller parts Plan a solution to a problem using decomposition. *Can take a complex problem, identify component parts, use decomposition to break this problem down and then plan how they can solve the problem by working through the elements they have identified. Projects could include developing a computer game, creating a website or designing a building.*	Solve problems by decomposing them into smaller parts Apply decomposition to help understand complex systems. *Can apply the principle of decomposition to help them to understand how complex systems operate. This could be software or combined hardware/software systems such as a smartphone. In this case, the child could consider input, processing, memory, output and connectivity hardware, operating system, application software and data as separate, interconnected component systems.*
Programming	Use sequence, selection and repetition in programs; work with variables. Use sequence in programs. In on-screen programming, the child's program should include a sequence of commands or blocks in an appropriate order. *A typical program could be a simple scripted animation, e.g. telling a joke, a story or explaining an idea taken from across the curriculum. The child's program might include multiple sprites; instructions could include movement, on-screen text, sound and/or costume changes.*	Use sequence, selection and repetition in programs; work with variables. Use sequence and repetition in programs (Scratch). Should include sequences of commands or blocks and some repetition. *Repetition would typically be for a fixed number of times, but might also include exit conditions (e.g. repeat...until...). Programs might include turtle graphics, simple music or a simple game.*	Use sequence, selection and repetition in programs; work with variables. Use sequence, selection and repetition in programs. Should include sequences of commands or blocks, some repetition and selection. *Repetition might include exit conditions (e.g. repeat...until...). Selection would normally be of an if...then or if...then...else type. At this level, expect the child to be able to combine repetition with selection. Programs might include a computer game or a turtle graphics design.*	Use sequence, selection and repetition in programs; work with variables. Use sequence, selection, repetition and variables in programs. *Program should include sequences of commands or blocks, repetition, selection and variables. Repetition might include exit conditions (e.g. repeat...until...) and perhaps a counter-variable for iteration. Selection would normally be of an if...then or if...then...else type. At this level, expect the child to be able to combine repetition with selection and variables. Programs might include a simple smartphone app.*
	Work with various forms of input and output Write a program to produce output on screen. Create a program that produces output on screen, such as moving sprites or displayed text, *e.g. a simple animation program.*	Work with various forms of input and output Write a program that accepts keyboard input and produces on-screen output. In Scratch (or similar), can write a program that displays a question, accepts typed input and responds in an appropriate way to what is typed. *This might be used as the basis for a dialogue program or a simple maths game.*	Work with various forms of input and output Write a program that accepts keyboard and mouse input and produces output on screen and through speakers. *In Scratch (or similar), the child can create a computer game using the keyboard or mouse for input and the screen and speakers for output.*	Work with various forms of input and output Write a program that accepts inputs other than keyboard and mouse and produces outputs other than screen or speakers. *Could create a smartphone app, using the touch screen and the GPS sensor or accelerometer for input, and the screen and speakers or headphones plus vibration motor or network connection for output.*

Logical thinking	Use logical reasoning to explain how some simple algorithms work. Explain a simple, sequence based algorithm in own words. Give an explanation for a simple algorithm based on a sequence of instructions. *The algorithm could be one of their own, or a simple one with which they have been provided. The algorithms could be recorded graphically, e.g. as a storyboard.*	Use logical reasoning to explain how some simple algorithms work. Explain an algorithm using sequence and repetition in their own words. Given an algorithm using both sequence and repetition, can give a coherent, logically reasoned explanation of what it does and how it works. *Repetition is likely to be 'forever' or for a set number of times, although end conditions (e.g. repeat...until...) could be used.*	Use logical reasoning to explain how some simple algorithms work. Give a clear and precise explanation of a rule-based algorithm. *When provided with a rule-based algorithm (e.g. for a computer game), the child should draw on logical reasoning to give a clear and precise explanation of what it does and how it works.*	Use logical reasoning to explain how some simple algorithms work. Give clear and precise logical explanations of a number of algorithms. *Given an algorithm, can describe what it does and, using logical reasoning, give precise explanations of how it works. Algorithms could be linked to programming projects, but might include a key algorithm such as binary search.*
	Use logical reasoning to detect and correct errors in algorithms and programs. Use logical reasoning to detect errors in programs. Give well-thought-through reasons for errors found in programs. *Find errors by reasoning logically about the program code, but they might also be able to use logical reasoning to identify errors in programs when they are executed. The programs do not have to be written originally by the child.*	Use logical reasoning to detect and correct errors in algorithms and programs. Use logical reasoning to detect and correct errors in programs. *The child can give well-thought-through reasons for errors they find in programs and explain how they have fixed these. The child can find and correct errors by reasoning logically about the program code; they might also be able to use logical reasoning to identify errors in programs when executed and confirm that they have fixed these by testing the new version of their program. The programs do not have to be written originally by the child.*	Use logical reasoning to detect and correct errors in algorithms and programs. Use logical reasoning to detect and correct errors in algorithms – identify patterns and explain why the algorithm is incorrect. *When given an algorithm for a particular purpose, e.g. a rule-based algorithm for a computer game or a sequence of steps to draw a geometric pattern, the child can use logical reasoning to identify possible errors in the algorithm, explaining why they believe the algorithm is incorrect. The child can use logical reasoning to suggest possible corrections to the algorithm, explaining* why these would correct the bug they identified.	Use logical reasoning to detect and correct errors in algorithms and programs. Suggest ways in which the efficiency of algorithms and programs can be improved. *Consider alternative algorithms for particular problems, using logical reasoning to compare these for efficiency. Examples might include comparing linear and binary search, or comparing exhaustive search and Euclid's algorithm for finding highest common factors.*
	Understand computer networks including the internet. Understand that computer networks transmit information in a digital (binary) format. *Explain that any information has to be converted to numbers before it can travel through computer networks. Understand that this conversion happens according to an agreed system or code.*	Understand computer networks including the internet. Understand that the internet transmits information as packets of data. *When working online, can explain that the information they send and receive is automatically broken down into packets of data, and that these sometimes take different routes across the internet.*	Understand computer networks including the internet. Understand how data routing works on the internet. *The child can give a coherent explanation of how data packets are routed from one computer to another on a separate network, which is also connected to the internet.*	Understand computer networks including the internet. Understand how mobile phone or other networks operate. Understand differences between network technologies. *Compare and contrast different network technologies, discussing differences in topology, range, bandwidth and fault tolerance.*

	Understand how networks can provide multiple services, such as the world wide web. Understand that email and videoconferencing are made possible through the internet. *Know that email messages are sent and received through servers connected to the internet. Know that Skype and other videoconferencing systems also work through the internet, but these services may be direct, peer-to-peer connections rather than via servers.*	Understand how networks can provide multiple services, such as the world wide web. Understand how the internet makes the web possible. *The child can give an explanation of how requests for web pages, and the HTML for* those pages, *are transmitted via the internet.*	Understand how networks can provide multiple services, such as the world wide web. Understand how web pages are created and transmitted. *Explain how HTML is used to create a web page and how it is transmitted as packets of digital data over the internet. The child should have an awareness of simple HTML tags (such as <h1> and <p> for marking up a web page.*	Understand how networks can provide multiple services, such as the world wide web. Understand how domain names are converted into IP addresses on the internet. *Give some explanation of how a domain name is converted into an IP address using the distributed domain name system (DNS) using something similar to a set of phone books. The child should show an awareness of the looked-up addresses (DNS records) being copied (cached), and that more local records are used in preference to more authoritative records in most circumstances.*
Creating content	Select, use and combine a variety of software (including internet services) on a range of digital devices. Use a range of programs on a computer. *The child can use a range of software on laptop or tablet computers with some degree of independence. Software might include video editing, diagnostic tools, email clients, videoconferencing (with the teacher or another adult), survey design software, spreadsheets and presentation software.*	Select, use and combine a variety of software (including internet services) on a range of digital devices. Use and combine a range of programs on a computer. *Can use multiple programs on laptop or tablet computers to achieve particular goals. E.g. They might record audio and then use this as samples in a composition; create HTML content in a text editor and preview it in a browser; analyse data in a spreadsheet and then create a presentation to show the results of their analysis.*	Select, use and combine a variety of software (including internet services) on a range of digital devices. Use and combine a range of programs on multiple devices. *Can use multiple digital devices (such as tablets and laptops or digital cameras and laptops) to achieve particular goals. The devices might include web servers, allowing them to use cloud-based applications. E.g. They might use local media in conjunction with a cloud-based programming platform, such as Scratch; digital cameras and video cameras to capture content to use on an externally hosted website or blog; a digital camera to take photos they could import into 3D design software on a laptop.*	Select, use and combine a variety of software (including internet services) on a range of digital devices. Select, use and combine a range of programs on multiple devices. *Can choose for themselves from a range of available programs on laptops, tablets or cloud-based services to achieve particular goals. E.g. They might choose which image editors and presentation software to use when making a presentation; which image and audio editors to use when creating media content for an app; which DTP, video editor and website tools to use when developing marking materials for an app.*

	Design and create a range of programs, systems and content that accomplish given goals. Design and create content on a computer. *Children can plan and execute a project in which they use software on a laptop or tablet to create digital content with some degree of independence. E.g. They could plan and shoot a video, plan and create a presentation on a given topic or plan and then create an online survey.*	Design and create a range of programs, systems and content that accomplish given goals. Design and create content on a computer in response to a given goal. *With a given goal, can plan and execute a project in which they use software on a laptop or tablet to create digital content with some degree of independence. E.g. They could plan and compose original music using sequencing software; plan and create a web page; plan how they could contribute to a shared wiki and then do so; plan and create a presentation about the weather. They should evaluate how effectively they have met the requirements of the original goal.*	Design and create a range of programs, systems and content that accomplish given goals. Design and create programs on a computer in response to a given goal and paying attention to the needs of a known audience. *With a given audience in mind, can design a program of their own in response to a given goal and write this in a block- based language such as Scratch. E.g. The child could design and create a computer game for a particular, known audience.*	Design and create a range of programs, systems and content that accomplish given goals. Design and create systems in response to a given goal, paying attention to the needs of a known audience. Plan, design and implement a system with multiple, interrelated components with a given goal and a known audience in mind. *E.g. They could develop a smartphone app, taking into account input, output and connectivity, the operating system, the algorithms, code and user interface of their own program. They should evaluate how effectively their system meets the specified goal and the needs of their audience.*
	Collecting, analysing, evaluating and presenting data and information. Collect and present information. *Use computers to collect information and present this to an audience. E.g. They could shoot and then show a video, read and respond to an email or conduct an online survey and present the results. They should be able to do this with a degree of independence.*	Collecting, analysing, evaluating and presenting data and information. Collect, analyse and present data. *Can use computers to collect numerical data, analyse this (typically in a spreadsheet) and present this to an audience. E.g. They could collect, analyse and present data about the weather over a period of time. They should be able to do this with a degree of independence.*	Collecting, analysing, evaluating and presenting data and information. Analyse and evaluate information from multiple sources. *Working with text, audio, images or video, can analyse information, perhaps summarising this or looking for common features or exceptions. They should evaluate the quality of the information, looking for bias or questioning assumptions that have been made. E.g. They could work with a number of sources of information on e-safety, evaluating their quality and providing a clear and coherent summary, drawing on multiple sources.*	Collecting, analysing, evaluating and presenting data and information. Analyse, evaluate and interpret data, being aware of the limitations of any conclusions drawn. *Can evaluate the quality of numerical data, deciding the extent to which it is affected by systematic or random errors. They should analyse their data, perhaps producing summary statistics, looking for relationships, trends and exceptions. They should provide an interpretation of their data and discuss the limitations of their findings. E.g. They could conduct market research for a smartphone app, and evaluate, analyse and interpret the data they obtain.*

Searching	Use search technologies effectively. Search for information within a single site. *Use browser-specific tools (e.g. the Find command) and site-specific tools (such as the search tools for Wikipedia or YouTube) to locate particular information on a web page or within a website.*	Use search technologies effectively. Use a standard search engine to find information. *Can use a common search engine (such as Google with safe search mode locked in place) effectively, to search for particular information on the web, such as answers to questions they identify in a research project.*	Use search technologies effectively. Can use advanced search options to make more effective use of a standard search engine. *Can use a common search engine (such as Google with safe search mode locked in place) effectively, to search for particular information on the web, such as answers to questions they identify in a research project. They should use advanced search options to filter their results, perhaps searching for a key phrase rather than keywords, using alternate keywords, or restricting their search to particular locations or domains.*	Use search technologies effectively. Appreciate that much information cannot easily be found using search engines. *Should be aware that not all questions can be answered using search engines. They should be able to give examples of 'ungoogleable' questions and consider some other ways in which these could be answered.*
	Appreciate how search results are selected and ranked. Understand that search engines select pages according to keywords found in the content. *When using search engines, the child should demonstrate their understanding that the pages shown include the keywords they have specified. The child can use this knowledge by thinking of good keywords appropriate for what they are searching.*	Appreciate how search results are selected and ranked. Understand that search engines use a cached copy of the crawled web to select and rank results. *Can explain how a search engine creates an index from a cached copy of the web and uses this to select and rank results. The child might also show an awareness of the Page Rank algorithm in which results are ranked according to the number and quality of in-bound links.*	Appreciate how search results are selected and ranked. Understand how search engines build a cached copy of the web using HTTP and web-crawler programs. *Can explain how a search engine creates a cached copy of the web using automated HTTP GET requests, follows links found, indexes results and uses the resulting index to select and rank results. The child might also show an awareness of the Page Rank algorithm in which results are ranked according to the number and quality of in-bound links.*	Appreciate how search results are selected and ranked. Appreciate that search engines now use many additional 'signals' to provide more relevant results. *Should be aware of the Page Rank algorithm used for ranking search results, but should also be able to discuss other signals used in ranking algorithms, such as bounce back rates, accessibility indicators, localisation and personalisation of search results.*