Year 9

Curriculum Checkpoints: What do students know and what can they do?

Computer Science		**Developing**	**Securing**	**Flourishing**	**Excelling**
Algorithms	Knowledge	Students can define basic concepts of abstraction and decomposition and provide simple examples of algorithms in the real world. They understand the purpose of sorting algorithms and can apply the bubble sort algorithm to small arrays of integers and words, though they may require assistance with more complex examples.	Students can effectively define and use abstraction and decomposition in problem-solving. They can explain the purpose of sorting algorithms and apply the bubble sort algorithm to larger unsorted arrays, both with integers and words. These students can recap the steps of bubble sort and understand how the insertion sort algorithm operates conceptually.	Students demonstrate a solid understanding of abstraction and decomposition, applying these concepts to solve real-world problems effectively. They can apply both the bubble sort and insertion sort algorithms to various unsorted arrays and articulate the steps involved. They also can explain the differences between sorting algorithms and discuss the effectiveness of bubble sort, insertion sort, and merge sort in various contexts.	Students can critically analyze and define advanced concepts of abstraction and decomposition, applying these to complex real-world problems. They proficiently apply multiple sorting algorithms, including bubble sort, insertion sort, and merge sort, to various unsorted arrays, demonstrating a thorough understanding of how each algorithm operates. These students can effectively evaluate and compare the efficiency of different sorting algorithms, providing detailed insights into their strengths and weaknesses in various scenarios.
Python	Knowledge/Practical	Students can define basic programming concepts like variables, print, and input. They can form simple sentences using print and input, and place variables in various parts of a sentence. They understand the basic purpose of if statements and can solve basic problems with them using numbers and strings, but need more support for boolean operators or loops.	Students can use variables in sentences efficiently and are comfortable working with input and output. They understand how to apply if statements with boolean operators and solve problems using different data types. These students can recognize when to use for or while loops and can perform basic string manipulations to solve common problems.	Students demonstrate a solid understanding of programming fundamentals, using variables dynamically in sentences. They can confidently use if statements and boolean operators to solve problems with strings and numbers, manipulate data types, and apply string functions effectively. They understand when to use for or while loops and can solve more advanced problems involving lists and arrays.	Students can skillfully solve complex programming problems using if statements with boolean operators and can manage a variety of data types and string manipulations. They can implement both for and while loops effectively, recognizing the most efficient loop for each situation. They also demonstrate a deep understanding of lists and arrays, using them to create well-structured solutions for advanced programming challenges.
Networks	Knowledge	Students can define basic network hardware and protocols, explain the difference between the World Wide Web and the internet, and understand how packets are sent across the internet. They can also identify different types of networks like LAN and WAN, but may need guidance to explain more complex topics like network topologies and addressing.	Students can describe the physical formation of the internet, explain the differences between star and mesh network topologies, and discuss some advantages and disadvantages of both. They can define MAC and IP addressing, explain the purpose of the Domain Name System (DNS), and understand the impact of bandwidth on network performance.	Students are able to evaluate the advantages and disadvantages of Wi-Fi networks and 2.4GHz vs. 5GHz wireless frequencies. They can explain the ethical implications of AI in network environments and provide examples of IoT devices. Additionally, they demonstrate a solid understanding of different network types, key protocols, and the components that help the internet function.	Students can critically evaluate the positive and negative ethical impacts of AI on networks, accurately explain the complexities of network protocols, and compare full mesh and star topologies in depth. They can also explain how bandwidth is influenced by various factors, justify their choice of network topology in different scenarios, and provide a thorough analysis of MAC and IP addressing, DNS, and how packets move across the internet.
Spreadsheets	Knowledge/Practical	Students can format basic data in Excel, sort and filter it, and use simple formulas. They are familiar with basic functions like VLOOKUP and can create simple graphs and charts, though they may require support with more advanced features such as pivot tables or conditional formatting.	Students can confidently use Excel formulas, sort and filter data effectively, and present relevant data using appropriate charts and graphs. They understand how to create and format pivot tables, slicers, and apply basic conditional formatting to their data. They can also use VLOOKUP and start to explore more complex formulas like IF statements.	Students can use advanced Excel features like creating and formatting data bars, combo boxes, and applying conditional formatting using formulas. They can present data clearly through well-structured charts, graphs, and pivot tables and are able to format them appropriately. They also show proficiency with VLOOKUP, IF statements, and can begin using macros to automate tasks.	Students can efficiently define and create macros to automate complex tasks in Excel. They can design fully interactive and well-formatted data presentations using graphs, slicers, pivot tables, and combo boxes. These students demonstrate an advanced understanding of conditional formatting, using data bars, and applying complex formulas like nested IF statements. They can critically select relevant data and present it effectively in various contexts.

Data Representation	Knowledge	Students can define basic concepts such as bitmap resolution, colour depth, and ASCII. They can calculate simple file sizes of bitmaps and understand how basic colours work within them. They can also perform straightforward conversions of ASCII numbers to characters but may need guidance with more advanced concepts like sound sampling or compression techniques.	Students can accurately define and calculate the file size of bitmaps, understand colour depth, and explain the differences between ASCII, extended ASCII, and Unicode. They can discuss the advantages and disadvantages of different character encoding systems, understand basic concepts of sound sampling, and calculate compression file size savings using the Run Length Encoding (RLE) algorithm.	Students can confidently define and calculate file sizes for more complex bitmaps, explaining how resolution and colour depth impact the size and quality of images. They can convert ASCII and Unicode values with ease and discuss how different encoding schemes affect file size and compatibility. They demonstrate a solid understanding of sound sampling, adjusting the sample rate and resolution to influence sound quality, and can apply compression algorithms to reduce file sizes efficiently.	Students can critically evaluate the trade-offs between lossy and lossless compression, apply RLE to compress text files, and calculate accurate file size savings. They can define and apply complex concepts like sound sampling and adjust sound characteristics to produce high-quality audio. These students can also deeply analyze the impact of different resolutions and colour depths in bitmaps and discuss the implications of using ASCII vs. Unicode in various contexts, providing detailed insights into encoding efficiency and compatibility.
Game creation in Python	Practical Skills	Students can define basic project specifications and create simple designs and plans for their gaming project. They can write small sections of Python code but may struggle with understanding more complex program structures. They can identify basic programming errors but may need guidance in resolving them.	Students can create well-structured project specifications and designs for their gaming project. They understand the advanced structure of Python code and can independently build larger sections of their program. They are comfortable debugging their code and demonstrate improved problem-solving skills, applying different types of test data (normal, boundary, and erroneous) in their test plans.	Students confidently create detailed project specifications, designs, and plans for their gaming project, ensuring that their code is well-organized. They can build substantial Python programs with complex structures and fix a variety of programming errors effectively. They also create comprehensive test plans, use normal, boundary, and erroneous data efficiently, and begin to develop an instruction manual for their project.	Students can define, design, and plan their gaming project with exceptional clarity, using highly organized specifications. They are adept at building large and complex Python programs, demonstrating an advanced understanding of code structure. These students can solve intricate programming problems, apply testing methods rigorously, and develop a clear and user-friendly instruction manual for their project. They can also critically evaluate and improve their project’s overall functionality and design.