Computing Curriculum Intent	
Our courses are designed not just to give students necessary programming skills but also introduce broader, multidisciplinary concepts, including ways to think about technology and how it will continue to impact society. Upon completion of our courses, students develop their transferable, technical and soft skills. Computational thinking and mathematical algorithms are two fundamental pillars of Computer Science. Our students develop/ enhance their existing transferable/ soft skills such as communication, presentation, team Building and problem-solving techniques. Regardless of CS/ IT, our students build on their fundamental programming skills in a range of programming languages. Our students are prepared for the apprenticeship or university pathways upon completing the course. We actively promote innovation – both in the teaching and learners' thinking – and we strive to build independent learners. Topics are generally divided into projects. All projects aim to take students through the process of learning, developing, applying and evaluating. Assessment is always focused on improvement. We actively encourage different pathways within our courses and the curriculum area - to ensure the needs of all learners are met.	
KS3	The key stage 3 curriculum provides challenges and new experiences in computing, digital literacy and digital media (regardless of their prior knowledge of using computers) and is designed to ensure students studying GCSE Computer Science have a basis of knowledge, skills and understanding in the fundamental concepts covered at KS4. Over time, students learn to, and develop proficiency in, program, starting with block-based languages before progressing to High-Level Languages. The development of programming skills is also built into physical Computing tasks using Micro:Bits for example coding LED lights to effectively apply the knowledge learnt in earlier Algorithm and Programming units. The curriculum journey connects to other curriculum areas holistically to ensure learning contexts are authentic, meaningful and provide opportunities for application of skills, investigation and purposeful play. In addition, references to key events and developments through the history of technology using role models from all aspects of society are used in an inspirational and motivational way for students. In year 9, we have computing core and enrichment as two different aspects of the course.
KS4	The increasing importance of information technologies means there will be a growing demand for professionals who are qualified in this area. Learners who've taken a GCSE in Computer Science and who then progress to study the subject at A Level or university will have an advantage over their colleagues who are picking up the subject at these levels. This course will develop critical thinking, analysis and problem-solving skills through the study of computer programming. For many learners, it'll be a fun and interesting way to develop these skills, which can be transferred to other subjects and even applied in day-to-day life. In computer science students will leave KS4 having knowledge, understanding and be able to apply this to computer systems. This includes PC's, tablets and all computing devices, how they are built, the processes used by computers, how the components work together/communicate, algorithms and programming, networks, cyber security, impacts of ethics, legal and environmental aspects of computing on society. Link to spec: https://www.ocr.org.uk/Images/558027-specification-gcse-computer-science-j277.pdf
KS5	In year 12 students have the opportunity to study an OCR A-Level Computer Science. This is a follow through for the GCSE and is designed for students to have a deeper understanding and master other topics such as how data is represented in computers, computer systems, consequences of using computer systems, communication and networks, computer organisation and architecture, databases, big data and functional programming. It is an intensely creative subject that combines invention and excitement, and can look at the natural world through

	a digital prism. Learners will develop an ability to analyse, critically evaluate and make decisions. The project approach is a vital component of 'post-school' life and is of particular relevance to Further Education, Higher Education and the workplace. Each learner is able to tailor their project to fit their individual needs, choices and aspirations. Link to spec: https://www.ocr.org.uk/Images/170844-specification-accredited-a-level-gce-computer-science-h446.pdf

For curriculum map - the work for sections below Years 12 and 13 can be found here

Type subject here Curriculum Implementation						
	Autumn 1	**Autumn 2**	**Spring 1**	**Spring 2**	**Summer 1**	**Summer 2**
Year 7	**Using Computers Safely Effectively & Responsibly/Google Induction** • use basic file management techniques to create folders, save, copy, move, rename and delete files and folders and make backup copies of files • recognise extensions for common file types such as .doc or .docx, .ppt, .jpg etc • keep their files in well organised and appropriately named folders • explain what constitutes a "strong" password for an online account	**First Steps in Small Basic** • Write and run programs in Small Basic using For…EndFor loops, variables, input-output and selection statements • Create a simple quiz game and identify and correct syntax errors in a program • Use a While…EndWhile loop in a program • Find and correct logic errors in a program • Use the graphics window to draw different shapes in random colours	**Computational Thinking** • Be able to ask logical questions to solve problems • Know the common Boolean operators: – AND – OR – NOT • Understand how Boolean operators can be represented in written expressions and Venn diagrams • Be able to complete truth tables for logic gates and circuits with up to three inputs	**Artificial Intelligence** • Understand the origin and uses of AI • Understand how rules are used in AI decision making • Understand what ethics is • Consider some simple ethical hypothetical problems • Understand how intelligence can be measured in humans and computers • Know what the Turing test is and how it works • Discuss the strengths and weaknesses of machine learning	**Introduction to Python** • Run simple Python programs in Interactive and Script mode • Write pseudocode to outline the steps in an algorithm prior to coding • Write programs using different types of data (e.g. strings and integers) • Correctly use different variable types (e.g. integer and floating point), assignment statements, arithmetic operators • Distinguish between syntax and logic errors and be able to find and correct both types of error	**App development** • Evaluate a simple GUI (Graphical User Interface) • Create a simple GUI (Graphical User Interface) within a web application • Explain the processes involved in building an app • Understand the term 'Home Screen' • Build a photo gallery • Use the map building tool • Create a sophisticated, error-free and attractive app suitable for its target audience • Write a good App description

	• describe a code of conduct • list some of the dangers and drawbacks of social networking sites • list some possible responses to cyberbullying • send and reply to emails, send attachments • use a search engine to find information		• Understand how loops can be used to reduce the amount of code required for a solution • Understand what an algorithm is • Create a sequence of instructions to achieve a goal	• Understand how bias can be introduced into AI algorithms and machine learning • Describe the opportunities and problems of using AI for sentiment analysis • Understand why interpreting patterns is not as useful a skill as 'thinking'	• Describe the purpose of pseudocode in designing algorithms • Use comments to document their programs and explain how they work • Write an error-free, well-documented program involving sequence, selection and iteration, but with some help given	
Year 8	**<u>Computer Crime and Cyber Security</u>** • Name the major Acts concerning computer use • Describe briefly some of the dangers of putting personal data on social networking sites • Describe briefly ways of protecting online identity • Identify some of the signs of fraudulent emails and respond appropriately • Adhere to Copyright Law	**<u>FLOWOL</u>** • Identify everyday situations where computer control is used • Identify common types of sensors used by control systems • Identify control flowchart symbols and understand how they are used to break down problems • Produce flowchart-based solutions for control	**<u>Understanding Computers</u>** • Distinguish between hardware and software • Give examples of computer hardware and software • Draw a block diagram showing CPU, input, output and storage devices • Name different types of permanent storage device • Suggest appropriate input and output devices	**<u>Python, Next Steps</u>** • Use data types correctly and convert between them when necessary • Write programs that use a loop to repeat a section of code • Write programs that use lists (known as 'arrays' in some languages) • Create and use a function with or without parameters • Find and debug syntax errors	**<u>Website Development</u>** • Understand the basics of website development and its importance and define what makes a website effective and user-friendly. • Learn how to identify and analyse the target audience for a website. • Use storyboards to plan the content and user interactions on each page of the website.	**<u>Microbits/Scratch</u>** • can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation • can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems

	when using written text, downloading music etc. • List some of the Health and Safety hazards associated with computer use • Describe how to safely dispose of an old computer	systems that include sequences and loops • Explain why control systems might fail and how this might impact on safety • Produce control solutions for problems that include subroutines • Produce control solutions for problems that include variables	for a simple scenario • Explain what RAM and ROM are used for • Show how numbers and text can be represented in binary • Explain the impact of future technologies	• Look at a given section of code and describe its function	• Learn the principles of UI design, including simplicity, consistency, and visibility. • Create a basic website layout using paper prototypes. • Identify key elements of a user-friendly interface (navigation menus, buttons, forms, etc.).	• are responsible, competent, confident and creative users of information and communication technology
Year 9 Core	**<u>Binary</u>** • Convert binary to denary values • Convert denary to binary values • Adding binary numbers together • Correctly identify the output of complex logic gate problems. • Complete truth tables for complex problems combining AND, OR, XOR and NOT logic gates. • Decode binary using ACSII conversion • Explain how image size, colour depth and resolution can alter the quality of an	**<u>Practical; Python Programming</u>** • The use of variables, constants, operators, inputs, outputs and assignments • The use of data types: Integer, Real, Boolean, Character, String and Casting • The use of the three basic programming constructs used to control the flow of a program: Sequence, Selection & Iteration The common arithmetic operators • The common Boolean operators AND, OR and NOT	**<u>Systems Architecture</u>** • Understand what systems architecture is and its importance in computing. • Identify the main components of a computer system. • Learn about the function and structure of the CPU. • Understand the concepts of clock speed, cores, and cache and explore the FDE cycle. • Identify various input, output devices and their functions.	**<u>Networks</u>** • State that the Internet is a wide area network and the world wide web is part of the Internet • Define the meaning of the terms “domain name”, http protocol • Explain the basic principle of packet switching • Give examples of LANs and WANs • State three different network topologies • Describe what is meant by a client-server network and state some of its advantages	**<u>Spreadsheet Modeling</u>** • Give examples of how computer models are used in the real world • Format a simple spreadsheet model • Use simple formulae and functions • Name cells in a spreadsheet model • Use a simple spreadsheet model to explore different “what if” scenarios • Create a basic pie chart to display results	**<u>3D modeling</u>** • Understand the basics of 3D modelling and its applications in various industries. • Identify different types of 3D modelling software and their interfaces. • Learn how to create and manipulate basic 3D shapes (cubes, spheres, cylinders, etc.). • Understand and apply transformations such as scaling, rotating, and translating objects. • Learn how to apply textures and

	image and the file size ● Describe the difference between vector and bitmap images		● Understand the role of these devices in the computer system.	● State why some transmissions are encrypted, and use a simple algorithm to encrypt and decrypt a message		materials to 3D models. ● Understand the difference between procedural and image-based textures.
Year 9 **Enrichment**	**<u>Programming</u>** ● The use of variables, constants, operators, inputs, outputs and assignments ● The use of data types: Integer, Real, Boolean, Character, String and Casting ● The use of the three basic programming constructs used to control the flow of a program: Sequence, Selection & Iteration The common arithmetic operators ● The common Boolean operators AND, OR and NOT	**<u>Networks protocols and Security</u>** ● Network threats ● Identifying and preventing vulnerabilities ● Operating systems ● Utility software ● The Internet ● Types of networks ● Network components, Topologies and factors affecting networks ● Wired and Wireless networking ● Client-server & Peer to Peer ● Standards, Protocols and Layers	**<u>Data representation</u>** ● Binary and Denary representation, ASCII ● Bitmap ● Numbers and characters ● Characters ● Images ● Sound ● Pixels ● Sampling ● Processing analogue & digital data	**<u>Algorithms and programming</u>** ● Computational Thinking ● Designing, creating and refining algorithms ● Linear search ● Binary search ● Bubble sort ● Merge sort ● Insertion sort	**<u>Systems architecture, memory and storage</u>** ● The purpose of CPU ● Common CPU components and their function ● Von Neumann architecture ● CPU performance ● Embedded systems ● Primary Storage(RAM & ROM) ● Secondary storage(Optical, Magnetic and Solid State)	**<u>Logic & languages</u>** ● Logic diagrams and truth tables ● Defensive design ● Errors and testing ● Translators and facilities of languages ● IDEs
Year 10	**<u>Impacts of digital technology</u>** ● Ethical issues ● Legal issues ● Cultural issues	**<u>Boolean logic/Programming languages and IDEs</u>** ● Boolean ● Logic Languages ● IDEs	**<u>Programming Fundamentals</u>** ● Programming fundamentals ● Data types	**<u>Algorithms</u>** ● Computational Thinking	**<u>Algorithms + Practical Programming</u>** ● Programming techniques ● Analysis ● Design	**<u>Summer exam revision</u>** **<u>Summer exam revision</u>**

	• Environmental issues • Privacy issues • Legislation relevant to Computer Science		• Additional programming techniques • Defensive design Testing	• Designing, creating and refining algorithms • Searching & sorting algorithms	• Development • Testing • Evaluation and conclusions	**Summer exam revision**
Year 11	**Networks protocols and Security** • Network threats • Identifying and preventing vulnerabilities • Operating systems • Utility software • The Internet • Types of networks • Network components, Topologies and factors affecting networks • Wired and Wireless networking • Client-server & Peer to Peer • Standards, Protocols and Layers	**Systems architecture, memory and storage** • The purpose of CPU • Common CPU components and their function • Von Neumann architecture • CPU performance • Embedded systems • Primary Storage(RAM & ROM) • Secondary storage(Optical, Magnetic and Solid State)	**Programming** • Defensive design considerations: Anticipating misuse & Authentication • Input validation Maintainability • Use of sub programs Naming conventions • Indentation • Commenting	**Revision of core J277/01 and /02 paper content**	**Revision of core J277/01 and /02 paper conte**	**GCSE exams**
Year 12	**Software Development** • Systems analysis methods • Writing and following algorithms • Programming paradigms	**Networks and Web technologies** • Structure of the Internet • Internet communication • Network security and threats	**Exchanging data** • Compression, Encryption and Hashing • Database concepts	**Boolean Algebra** • Logic gates and truth table • Simplifying Boolean expressions • Karnaugh maps	**Ethical, legal and moral issues** • The Data Protection Act 1998 • The Computer Misuse Act 199 • The Copyright Design and Patents	**Year 12 Mock Exams + NEA**

	• Assembly languages **Components of a computer** • Processor components • Processor performance • Types of processor • Input devices • Output devices • Storage devices Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	• HTML and CSS • Web forms and Javascript • Search engine indexing • Client server and peer to peer **Software development** • Functions of an operating system • Types of operating system • The nature of applications • Programming language translators Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	• Relational databases and normalisation • Introduction to SQL • Defining and updating tables using SQL • Transaction processing **Data types** • Primitive data types, integer, real/floating point, character, string and Boolean • Represent positive integers in binary • Use of sign and magnitude and two's complement to represent negative numbers in binary • Addition and subtraction of binary integers • Represent positive integers in hexadecimal • Convert positive integers between binary hexadecimal and denary	• Adders and D-type flip-flops **Programming Techniques** • Programming basics • Selection • Iteration • Subroutines and recursion • Use of IDE • Use of OOP Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	Act 1988 The Regulation of Investigatory Powers Act 2000 • Computers in the workforce • Automated decision making and Artificial intelligence • Environmental effects and Censorship and the Internet **Computational thinking** • Thinking abstractly • Thinking ahead • Thinking procedurally • Thinking logically • Thinking concurrently • Problem recognition • Problem solving Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	

			• Representation and normalisation of floating point numbers in binary			
Year 13	**<u>Data structures</u>** • Arrays (of up to 3 dimensions), records, lists, tuples • The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table • How to create, traverse, add data to and remove data from the data structures mentioned above **<u>Algorithms</u>** • 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	**<u>Data Structures (continued)</u>** • Arrays (of up to 3 dimensions), records, lists, tuples • The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table • How to create, traverse, add data to and remove data from the data structures mentioned above **<u>Algorithms (Continued)</u>** • 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	**<u>Programming Project(20% of overall grade)</u>** • Analysis of the problem • Design of the solution • Developing the solution • Evaluation Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	**<u>Programming Project(20% of overall grade)</u>** • Analysis of the problem • Design of the solution • Developing the solution • Evaluation Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	• Revision and past paper practice to embed knowledge and apply skills • Programming project completion Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	• Revision and past paper practice to embed knowledge and apply skills • Programming project completion Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.

	determine the efficiency of different algorithms, 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* algorithm, binary search and linear search) Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.	the efficiency of different algorithms, 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* algorithm, binary search and linear search) Students will complete weekly practical programming practice Students will be preparing for their exams throughout the term by learning exam techniques and practicing exam questions.				

Subject **Computer Science KS3**		**FUNCTIONS OF ASSESSMENT**		
		FORMATIVE; The instructional guidance that identifies central points of learning and plans for the progression of individuals students.	**SUMMATIVE;** This describes individuals learning at the end of an instructional unit by comparing it against a standard or bench mark. (High Stakes Assessment)	**EVALUATIVE;** This is about institutional accountability and comes after terminal exams.
TIMESCALE	**Annually**	KS3 Baseline Test during first/second week in September Quizzes, low stake testing, discussions and final prototypes of practical work.	End of year assessment during Assessment Week	Department Review Data analysis of KS3 results KS3 SOL scrutiny and restructure Assessment Week in summer
	Interim Could be termly or half termly	End of Unit Test every 6 weeks	**<u>Year 7 Topic Assessment List</u>** UCSER Small Basic Computational thhinking Python Beginners Artificial Intelligence App Development **<u>Year 8 Topic Assessment List</u>** Computer Crime and Cyber Technology Flowol Understanding Computers Python Next steps Web Development Microbits/Scratch **<u>Year 9 Topic Assessment List</u>** Binary Practical Programming Systems architecture Networks Spreadsheet modelling 3D modelling	
	Weekly	Homework-Google Classroom and in workbooks		

		Verbal feedback. Questioning. Suggestions of clubs to go to extend learning further.	
	Hourly	Key Questions Student presentations Lesson objectives Every lesson, every lesson last lesson, last week, last year Teacher, peer and self assessment – verbal feedback Questioning Success criteria explained	

Subject **Computer Science & IT KS4**		**FUNCTIONS OF ASSESSMENT**		
		FORMATIVE; The instructional guidance that identifies central points of learning and plans for the progression of individuals students.	**SUMMATIVE;** This describes individuals learning at the end of an instructional unit by comparing it against a standard or bench mark. (High Stakes Assessment)	**EVALUATIVE;** This is about institutional accountability and comes after terminal exams.
TIMESCALE	**Annually**	Baseline Assessment testing on basic computational thinking, programming concepts, algorithms and flowcharts. This enables for a starting point for making early judgements and informing subsequent formative assessment. Data logged into departmental trackers to monitor attainment.	Years 9 and 10 students will sit a GCSE style CS paper for their End of Year Exam to measure progress and outcomes from their starting points. Year 11 students will have their Trial GCSE exams in December which are internally marked by class teachers.. Results in January with feedback forms. Year 11 students will have their GCSE exams in May/June which are externally marked by OCR. Results in August.	The IT /CS department produces analysis of examination results at KS4 to identify strengths and areas to improve on to inform teaching and intervention strategies. Results data/final outcome: Data is used to identify students not making adequate progress Verbal and written evaluation of exams and progress

			There are there components in IT: Comp1 and 2 controlled assessment/coursework, comp 3 external exam.	
	Interim Could be termly or half termly	End of unit assessments(8 units in Total) Peer and self-assessment on Google classroom and worksheets. Re-ACT written feedback and student response.	End of Unit assessments with ReACT written feedback and student response Updating of Department Trackers to monitor students after every unit. **Year 11** 8 unit assessments altogether 1 Trial examination in December 1 Trial examination in March Past Papers from January until May examination. From January all students receive personalised learning checklists (PLCs) for every examination paper they complete. IT - trial exam **Year 10** 6 unit assessments End of unit assessment Trial Examination in April IT - Complete comp 1&2 assignments and prep for comp 3. **Year 9** 4 unit assessments + Python Essentials End of unit assessment End of year assessment in July. IT - Prep comp1&2 and mock assignments	
	Weekly	Formative assessment strategies take place including the following strategies: ● Worksheets/Homework on Google Classroom ● Exam questions, mark schemes and model answers on Google Classroom ● Lesson Ready PowerPoints/video links and articles on Google Classroom		

		• Coursework where applicable all students to proofread their work (ReACT)	
	Hourly	Lesson Outcomes are shared with students on PowerPoints- Google Classroom. Every lesson the following formative assessment takes place using the following strategies: • Python Challenges • Direct and Targeted questioning • Tiered questioning to clarify understanding using Bloom's Taxonomy • Last lesson, last week, last year	

Subject **Computer Science & IT KS5**		**FUNCTIONS OF ASSESSMENT**		
		FORMATIVE; The instructional guidance that identifies central points of learning and plans for the progression of individual students.	**SUMMATIVE;** This describes individuals learning at the end of an instructional unit by comparing it against a standard or benchmark. (High Stakes Assessment)	**EVALUATIVE;** This is about institutional accountability and comes after terminal exams.
TIMESCALE	**Annually**	Baseline testing for external Year 12 students GCE Alps and trial exam data is used to make judgement for assessment Year 13 public exams	Years 12 and 13 will sit an A- level style CS paper for their End of Year Exam to measure progress and outcomes from their starting points. Year 13 will have their Trial exams which are internally marked. Results in January with feedback forms. Year 13 will have their exams in May/June which are externally marked by OCR. Results in August.	The IT /CS department produces analysis of examination results at KS5 to identify strengths and areas to improve on to inform teaching and intervention strategies. Feedback is given to students throughout the year based on their folder organisation and the quality of work submitted.

			H446 (Coursework) will be carried out over 2 year and will be submitted to OCR for final marks by March/April.	ReAct is completed consistently to bridge any gaps. Data is used to identify students not making adequate progress.
	Interim Could be termly or half termly	End of unit assessments(12 units in Total) Student PLC's for each topic used on Google classroom Peer and self-assessment on Google classroom and Worksheets Re-ACT written feedback and student response	End of Unit assessments with ReACT written feedback and student response Completion of subject progress trackers / personalised learning checklists (PLCs) Updating of Department Trackers to monitor students after every unit. **Year 12** 12 unit assessments altogether 1 Trial examination in December (2hr) 1 Trial examination in March (2hr) Past Papers from January until May examination. From January all students receive personalised learning checklists (PLCs) for every examination paper they complete. H446 Coursework (Analysis and Design) **Year 13** 12 unit assessments altogether 1 Trial examination in December (2hr 30) 1 Trial examination in March (2hr 30) Past Papers from January until May examination. From January all students receive personalised learning checklists (PLCs) for every examination paper they complete. H446 Coursework (Development, Testing and Evaluation) Year 12/13 IT BTEC Single award - 4 Units (2 exams and 2 coursework) double award - 8 Units (3 exams and 5 coursework)	
	Weekly	Formative assessment strategies take place including the following strategies:		

		● Worksheets/Homework on Google Classroom ● Past Paper Exam questions, mark schemes and model answers on Google Classroom ● Lesson Ready PowerPoints/video links and articles on Google Classroom ● Coursework where applicable all students to proofread their work (ReACT)	
	Hourly	Lesson Outcomes are shared with students on PowerPoints- Google Classroom. Every lesson the following formative assessment takes place using the following strategies: ● Python Challenges ● Direct and Targeted questioning ● Tiered questioning to clarify understanding using Bloom's Taxonomy. ● last lesson, last week; last year	