THE HAT FEATURES IN DEPARTMENTS: Computer Science and Creative i-Media

1	**We review learning:** The lesson begins with a brief review of recent and previous learning, and learning is reviewed systematically, for example through quizzes and tests.	KS3: We start every lesson with a 5-minute review - to explore recall skills and basic understanding. KS4: We start every lesson with a 5–10-minute review of prior learning – to help us establish understanding; this is often used to explore linkages between concepts, etc. KS5: We start every lesson with a 5–10-minute review of prior teaching to establish if what we have taught previously is understood and whether students have completed lesson-preparation work is needed to support the next steps in their learning. Example: Coding exercises as "Fill in the blanks" or "Spot the errors". Theory lessons as "Explain key words" or "Name the description of Key Word". Coursework lessons as "Checklist check", "Reminder of resources location" or "Checking / Responding to feedback from last lesson."
2	**We make the learning clear:** Students are told what they will be learning (learning intentions) and are shown how they can make progress (success criteria).	For all Key Stages, the Learning Objectives are communicated to students at the beginning of each lesson, and we use (mini) plenaries in the form of Q&A throughout to check progress and understanding towards the LO. In Theory lessons Learning Objectives are clearly displayed and explained at the start. In Coding lessons, comments in the code are used to help structure the code ("Functions, then Variables, then Main Code etc.."), identify success criteria ("Successfully created a function for Encrypting etc..") and identify Learning Objectives "This program will allow the user to encrypt / decrypt a message they input".)
3	**We present new learning in small steps:** Students are given the opportunity to practise each step thoroughly, to obtain a high success rate.	KS3: CS students are coding regularly with our online Coding platform "REPL.IT", and we monitor closely how they use problem-solving skills in Google Classroom. We spend lessons preparing students for projects by working through relevant examples that students can then use as reference. We also present our Blue Peter ("here's one I made earlier") working program/solution with comments to explain each section in plain English and then guide students through a sequence of lessons (with these code segments) to the final working solution – as in the Adventure Game. KS4: I-Media. Report writing templates are provided in Google Classroom (together with checklists) to help guide students, step-by-step to a final report for each piece of Coursework. Theoretical tasks that support formal coursework are taught before students get to practice elements of the theory and complete the practical tasks. This follows a structure that starts with Research/Analysis, then Design and Planning, Development then Review. KS4: CS. Extensive coding is undertaken. Students get to design solutions to problems using a standard linear structure starting with Analysis of Features, Design, Coding and Testing – with a self-review. KS5: CS. Students are taught programming theory to underpin their own Programming Project before or during the completion of their NEA.
4	**We explain clearly and directly:** Explicit and detailed instructions and explanations are given throughout the lesson.	Through varied questioning approaches, we draw-out common misunderstandings across a group as well as individual ones. This ensures they understand what needs doing and if required some help with the "how". We regularly interrupt lessons deliberately to check understanding and allow students to clarify if we notice common misunderstandings across multiple students. In Theory lessons, instructions are left on the board and/or broadcasted to students' screens so they can see clearly. Example code is also broadcasted to students' screens to enable them to identify errors in their own code easier than trying to read off the board.
5	**We ask questions of everyone:** For example, through no-hands-up, cold calling and Think-Pair-Share, EVERYONE is involved and encouraged to think.	We deliberately avoid students that we know struggle with anxiety speaking in front of class but use any opportunity to get students to participate in discussions either 1:1 (desk visits), messaging on Google Classroom/NetSupport or via open classroom discussions. Students are also able through their coding able to express their views/responses in safety which can be shared with us or monitored through NetSupport. No hands is used from time to time so that depending on the group, students can expect to be asked to contribute.

THE HAT FEATURES IN DEPARTMENTS: Computer Science and Creative i-Media

6	**We provide models:** Evidence of modelling by thinking aloud, by using WAGOLLs, worked examples and partially worked examples, and by demonstrating (in practical work).	We use Google Rubrics publicly and extensively with students at KS4 which enable them to understand how their efforts meet expectations and explains what they need to do to succeed. All coding projects are constructed as collection of objects/procedures/functions that need to be combined at a later to date which we often give worked examples ("What do we start with? What goes next etc..") Where students struggle, they are given code "fragments" to help them, but they also have access to extensive online resources to help them code effectively via Google Classroom. I-Media has posts dedicated to resources related to the tasks they're currently completing as well as previously completed example tasks which are relevant.
7	**We guide students' practice:** Evidence of scaffolds (examples, models and writing frames) and teacher's movement whilst students are working to support and to provide corrections and feedback.	In I-Media and other coursework, we provide checklists which outline the major sections we expect as well as content required within each section. For each section, students first complete theory tasks which are more structured such as fill in the blanks, guided questions that can be applied to act as writing frames to coursework. Across all ages in Computer Science, we have students create "Example code" when we teach which allows them to store and apply the relevant coding techniques later. During Projects we would normally interact personally by circulating the room and spotting errors in code or providing support to those with their hands up. However, recently we have had to rely more on NetSupport to monitor students screens remotely. When students are completing programming projects, mostly in KS4 and 5, we set a Rubric - which students can read to ensure they hit all criteria that we use to produce a final score as well as adding specific comments relating to their project on Google Classroom. During theory lessons we provide a mixture of examples, models and writing frames depending on the age, ability and SEN of the students. Feedback is applied via Google Classroom as comments on their work or as marked assessments which are then returned to students.
8	**We require students to practice independently:** Clear opportunities for students to work alone, in order to thoroughly practice, for example through timed and un-scaffolded tasks in silence, while monitoring their progress.	After providing worked examples, WAGOLL or related tasks, students will then complete their work independently. In Theory lessons these will be part of a workbook or specific exercise on Google Classroom that they would have previously completed exercises on and had resources given to assist them. In KS4 and KS5 this is usually a project with a set deadline that requires students to go through with a specific methodology (Design, Code, Test, Review) or a mock assessment / singular exam question as a plenary etc... Students in KS3 are also set End of Topic assessments which allow them to demonstrate and apply their knowledge un-scaffolded within a time-limit.
9	**We check for understanding:** Evidence of questioning to check all students understand by asking them to explain what they have learned and by using all-students' response systems such as quizzes and mini whiteboards; evidence d adaptive teaching in response to the checking of understanding; students are retaught if they haven't got it.	As well as End of Topic Assessments that we use for KS3, we will often use Kahoot or Quizizz to check for understanding in the middle or towards the end of a topic. In KS4 and KS5 we set short, low-stake Exam Questions that we then go through as a class as well as Kahoot and Quizizz as we do for KS3. NetSupport also allows for gauging understanding using questioning tools that appear on each students' screen.
10	**We use retrieval practice systematically:** Evidence of retrieval practice to make the learning stick.	Repetition is our most common form of retrieval practice, ensuring students repeat key words and skills. These are initially recapped at the beginning of the lesson but then also applied throughout. In KS4 and KS5 we will repeat topics every year but also throw in Programming Projects that are related to theory topics, boosting their problem solving and coding skills as well as their retrieval of the theory topic. In KS3 we will rely on knowledge from previous topics or the previous years through recaps, class discussions, mind maps / spider-diagrams of everything they remember, posters / displays around the room, and reintroduction tasks which refamilarise them with the basics so that they can then deepen their knowledge with new content. KO's are used in lessons to practice retrieval. Knowledge from prior lessons is tested when appropriate but connected to current topic.