**Nature art**

**Lesson 2: Art algorithms**

**Introduction**

In this ‘unplugged’ lesson, pupils recap their understanding of algorithms before writing their own algorithms to show how they created their nature representations in lesson 1 of the ‘Nature art’ unit.

**Time:** @60 minutes

**Materials needed:** Art materials used in lesson 1, lesson presentation, printouts of *Algorithm Word Bank* and *Algorithm Support Sheet*, paper to write algorithms on.

**Learning objectives**

* To know and understand what algorithms are
* To write algorithms with clear instructions
* To test and debug algorithms

**Lesson summary**

* Recapping representations (5 minutes)
* Constructing algorithms (20 minutes)
* Paired algorithmic writing (20 minutes)
* Reviewing algorithms (15 minutes)

**Introduction: Recapping Representations (5 minutes)**

* Display **slide 3** and use the questions to encourage pupils to recap the representations they created in the previous lesson.
* Use **slide 4** to recap the computing concepts covered. Explain you will be focusing on one more important computing concept today.

**Constructing Algorithms (20 minutes)**

* Use **slide 5** to display a sequence of instructions (an algorithm) and ask pupils to think/pair/share what it is for (getting ready for school in the morning).
* Ask pupils if they know the word for a sequence of instructions in computing (an algorithm) and explain that they are going to write an algorithm that shows how they created their nature representations in the previous lesson (**slide 6**).
* Invite pupils to share their current level of understanding of algorithms, if possible, showing algorithms they constructed in previous years (discuss with the computing subject leader and/or staff from earlier year groups, e.g. England KS1).
* Draw pupils’ attention to the equipment and materials used last lesson along with large pieces of paper to pairs and ask them to write down the first 3 steps they took last lesson.
* Discuss these as a class to correct any misunderstandings and ask pupils to consider if the statements they have written would give instructions to people. Allow pupils time to change their statements to instructions and invite suggestions on the instructions they have created, recording them on the whiteboard and focusing on the use of the imperative verb.
* Use **slide 8** to recap on the term debugging and discuss the importance of pupils do this as they work.

**Paired Algorithmic Writing (20 minutes)**

* Give each pair a copy of the word bank containing useful verbs in the imperative form and the names of the equipment and materials used.
* Give pupils time to work in pairs to create their algorithm, using a fresh piece of paper if they wish.
* Encourage pupils to read and act out their instructions to each other in order to test and debug their algorithm as they go.

**Reviewing Algorithms (15 minutes)**

* When their algorithms have been completed, invite pairs to share their algorithm with another pair. Use **slide 10** to guide pupils’ evaluation of the other pair’s algorithm. Pairs should feedback their evaluations to each other.
* Invite pupils to share with the rest of the class the changes they might need to make to their algorithms more accurate. Recap on the name of the process pupils are undertaking (debugging).
* Use **slide 11** to review the learning outcomes of the lesson and invite pupils to think/pair/share how they have met these.

**Extension ideas:**

* Pupils could use presentation software and combine text-based instructions with digital images to write their algorithm.

**Differentiation**

**Support:**

* The algorithm support sheet can be given to pupils to help them sequence statements into an algorithm.

**Stretch & challenge:**

* Pupils could write an additional algorithm that explains how to use the art materials in one of the art-zones to produce a representation e.g. clay for sculpting, or be challenged to create a more detailed algorithm with additional steps.

**Opportunities for assessment:**

* Informal observations of pupils understanding of algorithms, debugging and evaluation during class discussions.
* More formal assessment if wished of the final algorithms.