## Multitasking Mania!

**Lesson 3: Scratch Introduction**

**Problem Statement:**

Your task is to create both a computer-based task and non-computer-based task that helps employees evaluate the effectiveness of multitasking and helps them plan their work efficiently.

**Lesson Overview:**

In previous lessons, students have been introduced to the concept of multitasking and have been asked to come up with a means of evaluating its effectiveness. Here, students will acquire programming skills that will help them accomplish this goal. They will receive an introduction to coding using the Scratch platform.

**Learning objectives:**

* Students will be able to: Understand the basics of coding
* Students will be able to: Understand the Scratch programming environment (stage, sprites, etc.)

**Lesson standards (NGSS, CCSS, CTE):**

|  |  |
| --- | --- |
| Computer Science (CTE) | |
| CTE 2-A-5-6 | Develop programs, both independently and collaboratively, that include sequences with nested loops and multiple branches. [Clarification: At this level, students may use block-based and/or text-based programming languages.] |
| Science (NGSS) | |
| MS-ETS1-1. | Define the criteria and constraints of a design problem with sufficient precision to ensure a successful solution, taking into account relevant scientific principles and potential impacts on people and the natural environment that may limit possible solutions. |
| Mathematics (CCSS) | |
| 6.NS.C.8 | Apply and extend previous understandings of numbers to the system of rational numbers. |

**Soft skills (21st Century Skills):**

* Technology Literature
* Learning and Innovation
* Life and Career

**Locally and/or personally relevant for students:**

* Many students play video games and use apps that are built on programming
* Many current and future jobs require programming skills

**Connections to Career and Educational Pathways:**

* Introduction video from code.org details career opportunities in programming

**Materials:**

* Computers - ideally 1:1, but could be 1:2 computer:student ratio
* Scratch Teacher Account with Student Scratch Accounts (see appendix)
* A copy of Scratch 101 Project instructions for each student/pair
* A copy of the Scratch Exit Ticket for each student

**Lesson preparation:**

* Knowledge of the Scratch Platform (see appendix for more information)

**Time required:** 45 minutes at a minimum

* 5 minutes: Introduction Video
* 10 minutes: Demonstrate common blocks (teacher)
* 20 minutes: Practice with Coding using Scratch 101 Worksheet (student)
* 5 minutes: Exit Ticket

**Grouping of students for instruction:**

Students will ideally have their own computer, but if not, you can pair students with mixed ability levels.

OPTION: Let students “self-select” out of the mini-lesson if they have prior experience with Scratch; they can independently code during that time (i.e. start the “practice session” early.

**Understanding the Problem**

|  |  |
| --- | --- |
| **Teacher** | **Student** |
| Introduce computer programming using a video. Providing instruction on using the Scratch platform, which will be used to solve the BleepBlorp problem later in the unit. | Do a think/pair/share about their prior understanding and experience with computer programming. |

1. Show the video “What Most Schools Don't Teach”. (<https://www.youtube.com/watch?v=nKIu9yen5nc>)
2. Have students do a Think-Pair-Share for the question “What is computer programming? Why is it important for the future?”
3. Let students know that they will need to learn to program in order to complete the task for Company BleepBlorp (design a computer-based multitasking activity).
4. Introduce Scratch as a language and its features using the following video: <https://www.youtube.com/watch?v=3ginlE868XI> (2:24 minutes).
5. Model first steps on the Scratch 101 Project.

**Exploring the Problem**

|  |  |
| --- | --- |
| **Teacher** | **Student** |
| Support students as they explore the common blocks with the Scratch programming language | Demonstrate their ability to use common blocks. |

1. Release students to computers to complete the Scratch 101 Project.
2. Administer Exit Ticket-Student Self-Assessment of the Scratch 101 Worksheet.
3. Review Student Scratch Programs. Reteach this lesson if needed before teaching the next Scratch lesson (Lesson 5).

**Accommodations:**

If needed, you can pair students with a computer to support learning.

**Extensions:**

Students who have prior knowledge of Scratch/movement blocks will be exempt from the demonstration so they can use that time to independently develop their programming skills. They will be expected to make a plan for how they will use that time and a goal.

**Assessment:**

Exit Ticket-Student Self-Assessment of the Scratch 101 Worksheet

Standard: Finish all steps in the Scratch 101 Worksheet

Extension Group: Demonstrate that they met their individually set goal

**References/Resources:**

Optional: Set-up a Scratch teacher account to manage student projects. Otherwise, students can save their work daily. A Scratch Teacher Account FAQ (includes 10-minute video) can be found at <https://scratch.mit.edu/educators/faq>.