|  |
| --- |
| ***Overview*** |
| *Topic/Theme*Please list a brief title for the task | Programming Variables and Functions |
| *Lesson/Activity Goal* | Students will be able to create a computer program that uses variables and functions to draw an infinitely growing shape. |
| *Rationale and Unit Placement*Please provide a few sentences that describe how this lesson or activity might fit within an existing unit. | Students working on this unit should have encountered [growing linear functions](https://bhi61nm2cr3mkdgk1dtaov18-wpengine.netdna-ssl.com/wp-content/uploads/2018/09/Seeing-and-Describing-FInal-copy.pdf) and will likely be familiar with the ideas of input, output, variables, and rate of change from their work in mathematics. This background knowledge will align with the performance task of drawing an infinitely “growing” square shape. Students working on this unit should have already learned how to move a sprite, draw using the sprite pen, and learned looping behavior. This multi-day lesson activity focuses on the technical skills of drawing in Scratch and thinking through how variables and functions work together in a program. This lesson plan features possible cross-curricular mathematical conversations that could take place either in the mathematics or computer science classroom. |

|  |
| --- |
| ***Standards Alignment***Please list the standards aligned with this task (e.g. K-2PA.2) |
| CS Standards6-8.DI.1 Use the basic steps in algorithmic problem-solving to design solutions (e.g., problem statement and exploration, examination of sample instances, design, implementing a solution, testing, and evaluation).6-8.PA.2 Implement problem solutions using a programming language that includes looping behavior, conditional statements, logic, expressions, variables, and functions.6-8.PA.3 Demonstrate dispositions amenable to open-ended problem solving and programming (e.g., comfort with complexity, persistence, brainstorming, adaptability, patience, propensity to tinker, creativity, accepting challenge). 6-8.DI.4 Understand the notion of hierarchy and abstraction in computing including high-level languages, translation, instruction set, and logic circuits.Math StandardsPS.1: Make sense of problems and persevere in solving them.PS.2: Reason abstractly and quantitatively.PS.4: Model with mathematics. PS.7: Look for and make use of structure.PS.8: Look for and express regularity in repeated reasoning.8.AF.6: Construct a function to model a linear relationship between two quantities given a verbal description, table of values, or graph. Recognize in y = mx + b that m is the slope (rate of change) and b is the y-intercept of the graph, and describe the meaning of each in the context of a problem7.GM.4: Solve real-world and other mathematical problems that involve vertical, adjacent, complementary, and supplementary angles. |

|  |
| --- |
| ***Attributions*** |
| *Created by:* | Catherine Floyd |
| *Date:* | May 21, 2019 |

---------------------------------------------------------------------------------------------------------------------

|  |
| --- |
| ***I. Introduction/Anticipatory Set***How might you make connections to students’ own experiences/ideas or other content to set the stage for the lesson/activity? |
|  Many students have done Mad Libs, and so it might be fun to start with a quick game of Mad Libs to motivate this lesson. There are also many examples of [square spirals](https://www.google.com/search?q=square+spirals&source=lnms&tbm=isch&sa=X&ved=0ahUKEwikjJ_9qa_iAhUFP6wKHTSRB7oQ_AUIDygC&biw=930&bih=1071) in art that could be a motivating conversation for this lesson.  |

|  |
| --- |
| ***II. Summary Description***Please describe the procedures or parts of the lesson/activity. If you are using an existing activity, you can include a link to the instructions. Feel free to provide any further instructions for how this lesson/activity might be adapted for be integrated with other parts of the unit.  |
| **Task 1: (Unplugged) Mad Libs**Use the following [instructions](https://curriculum.code.org/csf-1718/coursef/16/) to have students use [unplugged blocks](https://drive.google.com/file/d/0B-uvt08wYSQqdG8tMGlyNWlHelk/view) to do an activity similar to Mad Libs. Students plug in words for different variables and then plug them into a story or sequence. Class discussion:* What is a variable in computer science, and how is it the same/different compared to a variable in mathematics?

**Task 2 (Plugged) Create a Counter**Introduce the variable blocks to students. It might be helpful to have pictures of some examples of variable blocks and have them interpret what the outcome will be. There are some examples [here](https://nostarch.com/download/samples/Learn-Scratch-05.pdf). Have students create a counter, start at zero, and count up by 1 each second (solution [here](https://cs.harvard.edu/malan/scratch/variables.php)). Have students annotate (using the comments function) to describe important aspects of their program.Class discussion: * How can functions and loops work together?

*Extension*: Students can work on counters that start at different numbers and count up at different rates.*Cross-curricular connection:* Students can have discussions concurrently in their math classes (or in their CS class) using the idea of counting in terms of how else this could be represented (in tables or graphs) and how else it could be interpreted mathematically (as slope/y-intercept, an equation). **Task 3 (Unplugged) Walk a Maze**Offer students the following picture: Image result for spiral square Class discussion: * What do you notice about this picture?
* What is changing? What is staying the same?

Have them think about ways that they would need to walk to draw this shape as a sprite. Have students write down directions that would serve as an algorithm to walking and drawing this shape. Have students compare their algorithms with a partner and highlight similarities/differences in their ideas.Class discussion:* What programming tools do you think that we will need to use to create a program that draws this shape? [pen tool, motion tool]
* What tool(s) will we need to use to continue this pattern forever? [loops, functions]

*Cross-curricular connection:* Students can have discussions concurrently in their math classes (or in their CS class) using the idea of the changing length of side in terms of how else this could be represented (in tables or graphs) and how else it could be interpreted mathematically (as slope/y-intercept, an equation). **Task 4 (Plugged) Program a Sprite to Draw a Never-ending Spiral Square Pattern**Offer students the following picture: Image result for spiral square Grouping students in pairs, ask them to write a program where the sprite draws a square. Have students annotate (using the comments function) to describe important aspects of their program. Once students have finished their programs, have them do a short gallery walk.* How were the programs of the other students similar/different from your own program?
* What (optional) tools helped customize the squares? [change colors, backgrounds, etc.]
* What tools helped make some programs more efficient than others?
* What might you do differently in your program if you were to do this again?
 |

|  |
| --- |
| ***III. Whole Group Discussion Central Questions***What 1-2 central questions might be used with the whole class to solidify the main idea of the lesson/activity? |
| How can repetition be used to write algorithms efficiently? |
| How do variables and functions work together? |

---------------------------------------------------------------------------------------------------------------------

|  |
| --- |
| ***Evidence of Proficiency***Please list 3-5 categories and brief descriptions of what proficiency looks like for those categories (this can serve as the precursor to a rubric). You do not have to use all of the rows. |
| Category | Description |
| 1. Variables | Students create and use variables in their program. |
| 2. Functions | Students use functions correctly, as evidenced by programming their sprite to draw a never-ending spiral square pattern. |
| 2. Collaboration | Students work together to create a program, sharing the responsibility of thinking and coding. |
| 3. Annotations | Students provide annotations/comments to prove that they understand the purpose and utility of their chosen program functions. |