MISSION 14: Line Art	Time: 60-90 minutes

Overview:

This mission will lead students on a journey to discover the magic of computer graphics. It all starts with a pixel drawn on the screen. But things get much more interesting when code is looped to create patterns of logic, sounds, and **light!**

Cross Curricular:

- **MATH:** This mission uses some very advanced math. If your students are in an upper math class, you can do more with the math in the lesson, specifically deltas.
- **ART:** Have students design and create their own string art. It doesn't have to follow the same rules, but it could.
- **LANGUAGE ARTS:** Have students write step by step instructions on how to create computer string art.
- Supports **language arts** through reflection writing.

Materials Included in the learning portal Teacher Resources:

Mission 14 Slidedeck

The slide deck is for teacher-led instructions that let you guide students through the material using the slides. It is an alternative to the students reading a lot of instructions in CodeSpace. The slides mirror the instructions, with simplified language that is chunked into smaller sections at a time. The information is shown on slides with "Objective". The tasks to complete are on slides with "Mission Activity''.

Mission 14 Workbook

The workbook can be used instead of slides for student-led or independent work. It is an alternative to students reading a lot of instructions in CodeSpace. It mirrors the instructions (and the slide deck), with simplified language that is chunked into smaller sections at a time. Each objective is on its own page. The tasks to complete are labeled "DO THIS" and have a robot icon next to it.

Mission 14 Log

This mission log is the worksheet for students to complete as they work through the mission. It should be printed and given to each student before the mission starts. They write on the mission log during the assignment and turn it in at the completion of the mission (assignment).

Mission 14 Lesson Plan

The lesson plan comes from the original CodeX Teacher Manual and is included here for easy reference.

Mission 14 Remix Folder

Following Mission 14 students should complete a remix of their code.

Additional Resources:

- **Mission 14 Solutions**
  - All code solutions to LineArt in a text file
- **Review Kahoot** (Mission 14 Obj. 1-5)
- **Review Kahoot** (Mission 14 Obj. 6-9 & review)

Formative Assessment Ideas:

- Exit ticket(s)
- Mission log completion
- Completed program
- Obj. 1-5 Kahoot Review
- Obj. 6-9 Kahoot Review
- Student Reflection

Vocabulary:

- **Bitmap:** Graphics bits – drawing images and text. A bitmap is an object that can hold a 2D image of a given width and height; a list of pixel RGB values.
- **Pixel**: Elements of a picture, short for "picture element." They are the tiny dots that make up larger images.
- **Magic Number**: Numbers that just appear in code with no explanation. When something changes in the future, the number doesn't work anymore and you have to change it.
- **Literal**: A specific value, like 120 or 239; also known as a magic number.
- **Envelope:** In geometry, a curve created by straight lines moving down and across a grid.

Preparing for the lesson:

This mission will create string line art. It uses functions from bitmap, such as display.draw_line() and display.set_pixel(). The last three objectives are heavy in math. There is a link to a wikipedia article. If the math is too advanced for your students, they can still complete the mission using the code that is given. They are not required to understand all the math equations and concepts.

Also, students will learn and use **for** loops as an alternative to **while** loops. You may want to give a lot of practice on for loops before, during and after the mission.

Students will use the Codex throughout the lesson. Decide if they will work in pairs or individually.

- Look through the slide deck and workbook. Decide what materials you want to use for presenting the lesson. The slide deck can be projected on a large screen. The workbook (if used) can be printed or remain digital through your LMS.
- Be familiar with the Mission Log (assignment) and the questions they will answer.
- Print the Mission Log for each student, or upload digitally to your LMS.
- The mission program does not need to be portable. If you want students to use the CodeX without a cable, then have batteries available.

Lesson Tips and Tricks:

💡 **Teaching tip:**

You can use a variety of discussion strategies to get the most engagement from your students. For example, you can have students write their answers before asking anyone for an answer. You can use one of many think-pair-share methods. You can have students write their answer and share with someone, and then have other students share answers they heard from their peers. You can randomly select students to answer.

🧑‍🤝‍🧑 Pre-Mission Discussion (Slide 2, page 1):

Students can write in their log first and then share, or discuss first and then write in their log.

There is one question for the pre-mission. There isn't a "right" answer here. The purpose is to get them thinking about the need for selecting something random. Also, there are real-world applications to what they are learning.

- In this mission you will use loops to create beautiful and interesting art. In previous missions, you learned how to draw with lines, circles and rectangles. How do you think you can use loops to create art?

💻 Mission Activities:

Most of this lesson is on the computer, writing code to create string line art using for loops.

- Each student will complete a Mission Log.
- Students could work in pairs through the lesson, or can work individually.
- Students will need the CodeX and USB cable.

💡 **Teaching tip: Mission Introduction** -- Slides 3-4, Page 2
This objective introduces the mission and reviews bitmap functions like draw_rect() and draw_text().

💡 **Teaching tip: Objective #1** -- Slides 5-11, Pages 3-6
This objective introduces setting a single pixel on the screen. It reviews Bitmaps and the 240x240 screen.

Students will click on “Bitmap” to add it to their toolbox and answer a question in their mission log.

Then students will learn about getting information from the screen using a “get” command. You may want to practice or review “set” and “get”. Students will open their Console and type 4 lines of code directly into the console and see the result with a print. They will write the result in their mission log.

Students will create a new file and display at least 7 pixels on the screen.

💡 **Teaching tip: Quiz** -- Slide 12, Page 7

Students take a ❓ short quiz. The 2 Quiz questions are below. Both are about pixels. The second question can be tricky if they haven’t completed Objective 1 by typing the code on the slide in the console panel. The code in the instructions is for a different color than the quiz question, so following the instructions on the slide (or workbook) will help students understand and answer the question.

💡 **Teaching tip: Objective #2** -- Slides 13-1,5 Pages 7-8
Students replace their code with a for loop. They will delete all prior code except the import statement. The number for the range is not given. They need to remember the width of the screen is 240 pixels, which will be the range.

Students will click on “pixel” to add it to their toolbox and write the definition in their mission log.

💡 **Teaching tip: Objective #3** -- Slides 16-22, Pages 9-11
Students will add another loop to draw a vertical line. They must use variables for the range, not a literal value.

Students will answer two questions in their mission log.

🗝️ **NOTE:** Students will answer the question “what can you do to avoid magic numbers?” The answer can be straight from the textbook – get the display width and height straight from the source, or they can be more general and say something like using variables and constants to represent the numbers. The goal in the objective says to eliminate magic numbers by using the functions rather than literal numbers. You can remind students that a literal number IS a magic number.

🗝️ **NOTE:** Running the code WILL cause an ERROR.

💡 **Teaching tip: Objective #4** -- Slides 23-25, Pages 12-13
Students fix a bug by converting the division into integers.

Students will answer two questions in their mission log.

🗝️ **NOTE:** You may want to review the data types **int** and **float**

💡 **Teaching tip: Objective #5** -- Slides 26-32, Page 14-16
Students will use nested for loops to create a grid. They will use the "step" parameter in the for loop. During the objective, have students experiment with the step parameter so they can make a mental model of what it is doing before going to the nested for loops (see slide 27).

Students will answer two questions in their mission log.

💡 **Teaching tip: Quiz** -- Slide 33, Page 17

Students take a ❓ short quiz. The 2 Quiz questions are below. Both are questions on a nested for loop. You may want to go over this with the students, and even trace through it on the board.

💡 **Teaching tip: Objective #6** -- Slides 34-40, Page 17-20
This objective introduces display.draw_line() and reviews display.draw_rect(). Go over any of this material as needed for your students.

Students will answer two questions about the built-in functions in their mission log.

🗝️ **NOTE:** Students are asked to do a "Save As" and give the file a new name. This will keep the original file and start a new file with code already there. They will delete some code from the new file before adding more. If they don't do the "Save As", it is okay. They just won't keep their original code.

🗝️ **NOTE:** The horizontal and vertical line code in the slide is SLIGHTLY different than CodeTrek. The code doesn't have to have "display.width - 1" or "display.height - 1" to work. It will work without the subtraction.

💡 **Teaching tip: Objective #7** -- Slides 41-44, Page 21-22
A lot of information here about creating string art with envelopes. A link to wikipedia is included. This is really advanced math!! Don't worry about the math. Students can understand the concept without knowing the details. You can even skip the part about the envelope if you think it will be frustrating, and just use the term.

💡 **Teaching tip: Objective #8** -- Slides 45-50, Page 23-25
This objective helps students understand the math by using their fingers to "trace" the lines and see the movement. You might want to try this yourself first and demonstrate for the students. They can use their own CodeX, or you can print the webbing on paper and have them "trace" on the paper.

This objective shows students how to use a for loop to draw the lines to create the web. After they create the for loop, run the program. Then they will need to change the value of WEB_SPACING and run again to meet all the goals.

The CodeTrek uses literal numbers for 120 and 239. The slide instructions stay true to the “no magic numbers” and use the variables and functions for the values instead.

💡 **Teaching tip: Objective #9** -- Slides 51-60, Pages 26-30
This is a fairly long objective. It starts with adding another for loop to the code. Then students learn about creating a function that will do all the work for them. There is a lot of math! Don’t stress about it. Just teach the math at their level, and they don’t have to understand all of it, just that it changes the values so the lines create a curve.

Then students create the function and replace the for loops with function calls. Four function calls are provided. Students must call the function two more times. They can try to complete the webs that are drawn, but they don’t have to. Any two function calls will meet the goal. The solution provided shows the completed webbings.

💻 Mission Complete:

This mission ends with a completed, working program that will draw string art. You need to decide how you will use the program for assessment. You could:

- Go to each student and check-off their code
- Have the students download their code to a text file and turn it in using your LMS
- Have students print their code (either download and then print the text file, or print a screenshot)
- Have students switch computers and run each other’s code. Fill out a simple rubric and turn in to teacher
- Any other way that works for you

👫 Post-Mission Reflection:

The post-mission reflection asks students to think about using their creativity and working through frustration. You can change the questions if there is something else you want to emphasize with your students.

- You have learned a lot about pixel art! How did you use your creativity to complete the program?
- This program can be frustrating. How did you manage your frustrations and work through problems?

End by collecting the Mission Log and any formative assessment you want to include.

💻 IMPORTANT Clearing the CodeX:

The students have already created a “Clear” program. Students should open and run “Clear” at the end of each class period.

SUCCESS CRITERIA:

- ❏ Draw an x- and y- axis on the display screen
- ❏ Draw a white grid of dots on the display screen
- ❏ Draw a blue square as a border around the screen
- ❏ Use built-in screen information for drawing lines (display.width and display.height)
- ❏ Create a function for drawing a web
- ❏ Call the web function at least 6 times

❓ Quiz Questions

Quiz 1

Pixel Basics

Answer the questions below about the following code:

```
display.set_pixel(200, 200, CYAN)
c = display.get_pixel(200, 200)
print("Color = ", c)
```

Where on the CodeX 🔧display does the CYAN pixel appear? +5 XP

- ✗ Upper Right
- ✗ Upper Left
- ✓ Lower Right
- ✗ Lower Left

What is printed on the 🔧console? +5 XP

- You either need to consult the 🔧RGB Colors documentation, or just try it for yourself!

- ✗ `Color = (0, 255, 255)`
- ✗ `Color = (144, 210, 48)`
- ✗ `Color = (0, 0, 0)`
- ✓ `Color = (0, 252, 248)`

Quiz 2

Graphics Ranger

The following code draws a dashed line across the screen.

```
for d in range(0, 240, 20):
    for x in range(d, d + 10):
        display.set_pixel(x, 120, WHITE)
```

What is the orientation of the *dashed line?* +5 XP

- ✓ Horizontal
- ✗ Vertical
- ✗ Diagonal

How many pixels *long* is each dash? +5 XP

- ✗ 40
- ✗ 20
- ✗ 1
- ✓ 10