MISSION 4: Display Games	Time: 90 minutes

Overview:

In Mission 4, students will program the CodeX to display text and get input from the user by pushing buttons to create a game.

Cross Curricular:

- **MATH:** Students learn that computers can do math really fast. Use the computer as a calculator to check their work.
- **COMPUTATIONAL THINKING:** Students must make a plan for their game and follow the plan. They have to pay attention to details.
- Supports **language arts** through reflection writing

Materials Included in the learning portal Teacher Resources:

Mission 4 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 4 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 4 Log (and answers)

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 4 Lesson Plan

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

Mission 4 Remix Folder

Following Mission 4, students should complete a remix of their code. Get supplemental materials from the folder.

Additional Resources:

- **CodeX mission reminders** – in Unit 1
- **Mission 4 Display Solution** – in Answers section)
- **Kahoot** (Mission 4)

Formative Assessment Ideas:

- Exit ticket
- Mission log completion
- Completed program
- Kahoot Mission 4 Review

Vocabulary:

- **Integer:** A whole number that can be positive, negative or zero
- **String:** A sequence of characters, like words or sentences
- **Conversion Function:** a built-in function that converts a value to a different (and specific) data type
- **Branching:** Decision points in code; a condition
- **Boolean:** True or False data type (values that can be True or False)
- **Indention:** Structuring blocks of code in Python; statements ending with a colon (:) execute the block of code indented four spaces beneath it

Preparing for the lesson:

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.
- Decide if you want to discuss CodeX mission reminders. This mission follows a remix, so you may feel like they are good reminders to go over right from the start.
- Be familiar with the Mission Log (assignment) and the questions they will answer.
- Print the Mission Log for each student.
- If you have a word wall, or another form of vocabulary presentation, prepare the new terms.

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:**

Students can write in their log first and then share, or discuss first and then write in their log. There are two questions for the pre-mission. You could discuss them together or one at a time. There aren't any "right" answers here. The purpose is to get them thinking about displaying text on the screen as well as images. Also, there are real-world applications to what they are learning.

- What are some things that have a display or screen?
  - Possible answers: car dashboard, alexa, stadium billboards, kitchen appliances
- What are some things you might want to display on a screen?

💡 **Teaching tip – Reminders for the beginning of mission: (go over these if you think it will help your students)**

These reminders are organized on a short slide deck that can be shown to students at the beginning of class

- Always start a new program by creating a new file and naming it appropriately. If you don't, you will lose all your previous work! Using descriptive file names is essential to finding the program later!
- You are making a project – not just working random problems. Focus on the *project-based* objectives and avoid rushing through the material too quickly.
- Test your understanding along the way by "coloring outside the lines". Try stuff!
- Collect all the Tools you find! (They're indicated with a **wrench** icon)
- Read carefully – usually the answer is right there in front of you!

💻 **Mission Activities:**

Most of this lesson is on the computer, writing code to display text, light pixels, and detect a button press.

- 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: Objective #1** -- Slides 3-5, Pages 3-5
Students review displaying a CodeX image on the screen.

📝**NOTE:** in the CodeSpace instructions, the term “argument” is used. This isn’t emphasized in the lesson or included in the vocabulary at this time. It will appear in a later mission. The mission can be completed without going over the term. It isn’t used in the slide deck or workbook.

💡 **Teaching tip: Objective #2** -- Slides 6-8, Pages 6-7
Students learn about data types. This is a new concept. You may want to pause here and practice data types with students. This objective mentions three data types. You could go over some examples with your students and make sure they are clear on what they are and what values represent different data types. There is a workbook question for this objective.

💡 **Teaching tip: Objective #3** -- Slides 9-11, Page 8
This is also a new concept for students: the computer can do math. One example is given. You might want to pause here and go over other examples with your students so they have a broader idea of the math a computer can do.

📝 **NOTE:** The code students type will cause an error. Students will write about the error in their log.

💡 **Teaching tip: Objective #4** -- Slides 12-14, Pages 9-10
This objective is about converting one data type to another, specifically an integer to a string. You can pause here and do more practice with your students.

💡 **Teaching tip: Objective #5** -- Slides 15-16, Page 11
Students display two messages. But, like the pixel experiment in Mission 3, only the last message will be displayed because of the speed of the computer.

💡 **Teaching tip: Objective #6** -- Slides 17-18, Page 12
A new way to display text. Instead of display.show(), students can use display.print(), where more than one line can display at a time.

📝 **NOTE:** The instructions in CodeSpace talk about Escape Sequences. This isn’t necessary for the lesson. Unless you have a student that wants to print something weird, they won’t need any escape sequences.

💡 **Teaching tip: Quiz** -- Slide 19, Page 12
Students take a ❓short quiz making a prediction. The 2 Quiz questions are below. You can decide if you need to go over the question with your students.

💡 **Teaching tip: Objective #7** -- Slides 20-24, Pages 13-15
Another new concept! Branching. An example is given for coding. You might want to pause here and give several examples of branching in everyday life. If it is a weekday I go to school, otherwise I stay home. If I like dinner I eat it, otherwise I make a sandwich. If I have chores to do, I get to work; else I play games. Have the students come up with some examples.

The objective will have students use a button press for the condition. You could have students think what they want to happen if a button is pressed: light a pixel, display an image, show a text or play a sound are examples.

💡 **Teaching tip: Objective #8** -- Slides 25-26, Page 16
Students use the simulator to find all six CodeX buttons.

💡 **Teaching tip: Objective #9** -- Slides 27-28, Page 17
Students use an actual button press for the branching. They need to be careful with their typing; it needs to be exact. The button names use an underscore (_) and not a dash(-).

💡 **Teaching tip: Quiz** -- Slide 29, Page 18
students take a second ❓ short quiz about button presses and branching. Quiz questions below. You can decide if you need to go over the question with your students.

💡 **Teaching tip: Objective #10** -- Slides 30-35, Pages 18-20
Students complete their game by copying and pasting code and making slight modifications to indicate the button to press and the pixel to light. Students write in their log the buttons they want to use (they can choose any 4 of the 6 buttons).

Students can share their games with each other. They should have slightly different games since they can pick which buttons to press and in which order.

💻 Mission Complete:

This mission ends with a completed, working game. 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:

Several new concepts were introduced in this lesson. The post-mission reflection asks students to review three of the concepts. You can change the question if there is something else you want to emphasize with your students. You might want students to share their answers, especially the last question.

- What are the ways you can detect the press of a button (there are 2 ways)
- What are the four data types used in the mission?
- What do you remember about branching?

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

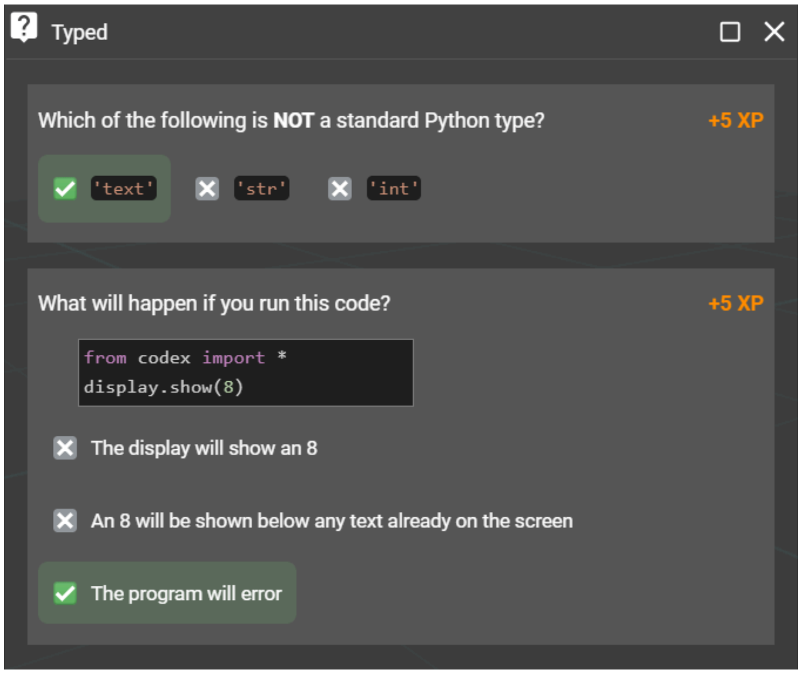
💻 IMPORTANT Clearing the CodeX:

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

SUCCESS CRITERIA:

- ❑ Understand and use variable data types, converting types when needed
- ❑ Use a Boolean condition in an if..then statement
- ❑ Receive input from the user through a push-button
- ❑ Program a push-button to make a fast-click game.
- ❑ Debug any errors in the code
- ❑ Write a program, run it, and save it to the CodeXs
- ❑ Clear the CodeX of meaningful code

❓ Quiz #1 Questions



Typed

Which of the following is **NOT** a standard Python type? +5 XP

✓ 'text' ✗ 'str' ✗ 'int'

What will happen if you run this code? +5 XP

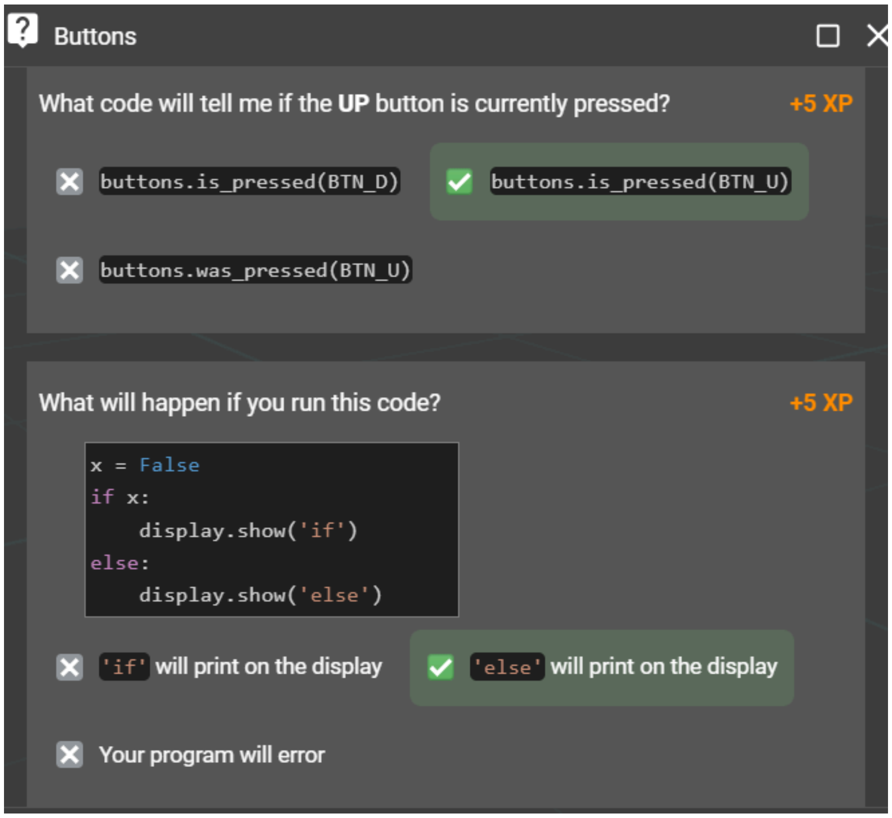
```
from codex import *
display.show(8)
```

✗ The display will show an 8

✗ An 8 will be shown below any text already on the screen

✓ The program will error

❓ Quiz #2 Questions


Buttons

What code will tell me if the **UP** button is currently pressed? +5 XP

- ✗ `buttons.is_pressed(BTN_D)`
- ✓ `buttons.is_pressed(BTN_U)`
- ✗ `buttons.was_pressed(BTN_U)`

What will happen if you run this code? +5 XP

```
x = False
if x:
    display.show('if')
else:
    display.show('else')
```

- ✗ `'if'` will print on the display
- ✓ `'else'` will print on the display
- ✗ Your program will error