MISSION 10: Reaction Tester

Time: 50 minutes

Overview:

In this project, students will create a game that measures the time between the display lighting up and a button being pressed. After the measurement is complete, this time will be scrolled across the display until a button is pressed to restart the game. This lesson will combine many concepts used in earlier programs into one complete game. It will include a countdown, a random delay, turning on neoPixels, using a button press, and variables for calculating the wait time. Also, the opposite of a kill switch, a "don't start until pressed" function.

Objectives:

- I can write a function to make code more efficient and readable.
- I can utilize multiple variables to a new program and describe their purposes.
- I can utilize loops to make my code more efficient.

Standards:

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-IC-26 Demonstrate ways a given algorithm applies to problems across disciplines.

CSP Framework:

Computational Thinking Practices:

3.B Use abstraction to manage complexity in a program.

4.C Identify and correct errors in algorithms and programs, including error discovery through testing.

6.A Collaborate in the development of solutions.

Key Concepts:

- Computers are driven by internal clocks.
- Use the time.ticks_ms() function to determine how long the CodeX's clock has been running.
- Functions can have named parameters, like loop=True and wait=False.
- The DRY concept. Never write the same code twice!

Preparation:

Make a copy of the assignment or put it in the LMS.
Prepare formative assessments

Links:

- Assignment
- Daily reflection form
- Mission 10 code (with more functions
- Mission 10 (from lesson)

Agenda:

- Warm-up (5 minutes)
- Mission 10 (40 minutes)
- Wrap-up (5 minutes)

Vocabulary:

- **Argument (review from #9):** The value passed into a function – information the function needs to complete its task. An argument can be a literal value, a variable, or an expression.
- **Computer Clock:** Electronic clock circuits; the heartbeat of the computer. The tick of the clock moves through the code one line at a time. It is also used in the sleep function, scheduled activities within the CPU, and everything timing related on the computer.
- **Monotonic**: Always increasing (a computer's electronic clock is not monotonic; like an odometer it will wrap-around once the highest value is reached)

Assessment:

- Daily reflection journal or form
- Exit ticket or group discussion

Teaching Guide

Warm-up (5 minutes)

The actual coding part of this Mission is about one normal class period. The extensions for this mission are fairly easy and don't take much time. Encourage the students to do them if they have time. The challenge is included only if some students work very quickly. It is OPTIONAL!

🧑‍🤝‍🧑 **Discuss** – Use a discussion strategy, like journaling, working at boards, selecting random students, or a form of think-pair-share.

- **Topic:** Two choices here: Ask the students how they think a computer keeps track of time, like in the sleep command. Talk about the computer clock and its uses (see vocabulary). OR decide on a topic that may need review with your students before starting the mission. You may want to review functions or arguments and parameters. Or you can go over common programming errors. Spend five minutes or less doing a quick review and then get right to the mission.

Activity – Mission #10 with extensions (40 minutes)

💻 Randomly group students into pairs for pair programming.

Students log in to one computer. Two computers can be used if they want to see instructions on one computer and work on the other computer. However, the assignment document requires snippets, so it will need to be open on the same computer as CodeSpace.

💡 **Teaching tip – Before they start:**

Optional: Review the Mission Reminders slides.

Important! Remind students that they need to document their errors and how they fixed them. There is a table at the end of the assignment document for this.

Students go to sims.firialabs.com and should be at the beginning of Mission 10

💡 **Teaching tip – Objective 1:**

The objective reviews random numbers and the argument for randrange. There are some questions to answer on the document students need to answer. They should read the instructions carefully.

💡 **Teaching tip – Objective 2:**

This objective also is a review – displaying a countdown. A suggestion is given that students create a function for this instead of typing directly in the main program. It is optional, but suggested.

💡 **Teaching tip – Objective 3:**

This objective has students use **import time** instead of **from time import sleep**. This means whenever they use sleep in their code, they need to use time.sleep(). If they are getting errors, be aware of this.

💡 **Teaching tip – Objective 6:**

Students create a function for the wait button. This is similar to the kill switch but does the opposite – wait for the button press before beginning. It is used twice in the code, so the instructions talk about reducing repetition by creating a function. This is a good concept to reinforce – a reason for functions, and also an example of abstraction.

💻 **Teaching tip – Extension #1:**

Complete this extension if at all possible. It doesn't take very long. Students create a "kill switch" for the program. Also, encourage the students to include a message after the loop – not indented – that displays the program has ended.

💻 **Teaching tip – Extension #2:**

Complete this extension if at all possible. It doesn't take very long. Encourage students to look over their code and create one or two more functions to chunk the code and make it readable. This is also a great example of abstraction. In my mission 10, I created a function for the random part and a function for the looping part (getting the reaction time).

💻 **Teaching tip – Challenge:**

This is completely optional. It is only for students who work very fast so they have something to work on throughout the class period. Students shouldn't end early and have free time. For the challenge, students complete the feel of a game by using a counter and if statements (selection) to create a win or lose. They should put the code in a function.

✅ Assignment is complete and ready to turn in. Both students should include their names on the document.

✅ Determine how you want to check-off the student program (turn in text file, submit through LMS, observe on student computer, etc.)

✅ **IMPORTANT!!**

Students should clear their CodeX by running their ClearCodeX program.

Wrap-Up (5 minutes)

👩‍🏫 **Vocabulary** – Review the vocabulary for today's lesson:

- **Argument (review from #9):** The value passed into a function – information the function needs to complete its task. An argument can be a literal value, a variable, or an expression.
- **Computer Clock:** Electronic clock circuits; the heartbeat of the computer. The tick of the clock moves through the code one line at a time. It is also used in the sleep function, scheduled activities within the CPU, and everything timing related on the computer.
- **Monotonic**: Always increasing (a computer's electronic clock is not monotonic; like an odometer it will wrap-around once the highest value is reached)

🧑‍🤝‍🧑 **Discuss** – Use a discussion strategy, like journaling, working at boards, selecting random students, or a form of think-pair-share.

🧑‍🤝‍🧑 **REVIEW the TOPIC:** Discuss real world applications for detecting time. Computers measure time in all types of applications.

- Football play clocks, and stop watches for other sports.
- Electronic Drum Machines
- Microwave Oven timers
- Alarm clocks

Formative Assessment:

- Daily reflection journal or form
- Completion of assignment and mission
- Exit ticket on vocabulary
- Group review on vocabulary
- Programming journal
- Students create a vocabulary canvas with vocabulary words.

SUCCESS CRITERIA:

- ❑ Give the player a 3-2-1 countdown.
- ❑ Program a random delay so the player can’t “guess” the timing.
- ❑ Measure the time until a button press occurs.
- ❑ Scroll the reaction time across the display.
- ❑ Wait for a button press, then restart the game.
- ❑ Use functions to reduce repetition and to organize your code.