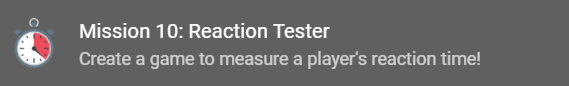


**Mission 10:**

**Reaction Tester**

**Student Workbook**



****

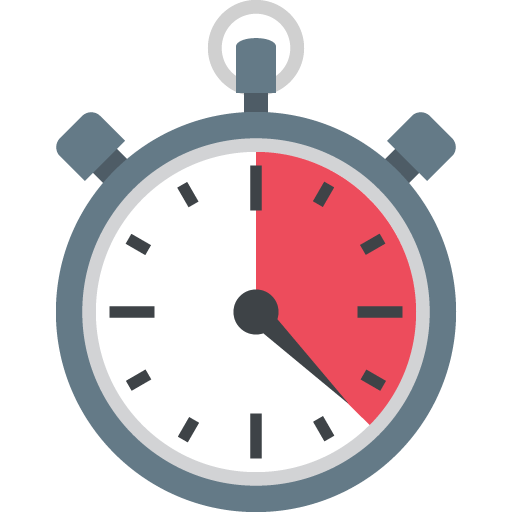
**Let’s get physical!**

In the last mission, the program used functions, parameters and arguments. For this mission, you tap into the power of CodeX by using the built-in capabilities of its powerful clock.

Go to the Mission 10 Log and fill out the   
Pre-Mission preparation.

* In this mission you will use a computer clock to measure time. What are some things you use a timer for?

**Mission 10: Reaction Tester**



**How fast is your reaction time?**

**In this project you will make a device to measure your reaction time. This project will:**

* Give a 3-2-1 countdown
* Wait a random delay
* Turn the pixels GREEN
* Measure the reaction time for the button press
* Loop and do the countdown again

**Mission 10: Get started**

* Go to <https://make.firialabs.com/> and log in. 
* Go to Mission 10



* Click and start Mission 10.

**Objective #1: Milliseconds**

This mission will require you to turn on all the pixels the same color.

The code so far turned on a single pixel at a time:

* pixels.set(0, RED)

Using a list, there is an easier way:

* pixels.set(**[RED, RED, RED, RED]**)
* Do you notice the list with four items?
* The pixels.set() command needs parenthesis, and the list needs [ ]
* Make sure you use both, in the correct order

**Objective #1: Milliseconds**

CodeX’s powerful clock can work in milliseconds -- that’s 1,000 times per second!

You will want a random time in milliseconds, so you just have to do a little math.

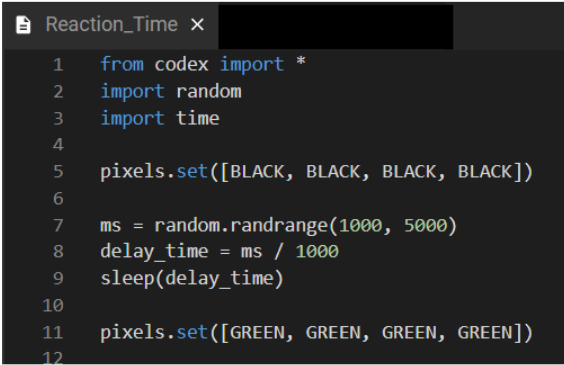
random.randrange(1, 5) gives a random number between 1 and 4

random.randrange(1000, 5000) gives a random number between 1000 and 4999.

* This gives you a good range of milliseconds, but sleep() uses seconds
* 1000 milliseconds = 1 second, so
* Divide the random number by 1000!

**Objective #1: Milliseconds**

**DO THIS:**

* Start a new file named **Reaction\_Time**
* Import the codex module
* Import the random module
* Import the time module
* Turn all pixels BLACK
* Get a random number using 1000 and 5000 as the range
* Divide the random number by 1000
* Use the random number in sleep()
* Turn all pixels GREEN

**Objective #2: The Countdown**

To make this into a game, you want to give a countdown.

* This will let the player know the game is starting.
* It also indicates when to start the timer.
* Use display.clear() to clear the display
* Use display.print() to countdown from 3 to 2 to 1 (with a sleep delay in between)
* You can scale the number bigger on the display for easy viewing
  + display.print(“3”, scale=6)
  + sleep(1)

**Objective #2: Click to flick**

**DO THIS:**

* Clear the display & the pixels
  + Set all pixels to BLACK
* Countdown from 3 to 2 to 1
* Clear the screen again
* Then continue the rest of your code to get a random number and light all pixels GREEN



**Objective #3: The Fourth Dimension**

Computers relay on electronic clock circuits

* Clock circuits are used to move through code
* They are used as time delays in the sleep() command
* When you turn on CodeX, its clock is continuously running.

So far you have used the time module for sleep()

* The time module also has a function that returns the current time on the computer clock

If you want to use more than one function from a module, you need to import the entire library, not just one function

* from time import sleep
* This imports only one function
* import time
* This imports the entire library

**Objective #3: Fun functions**

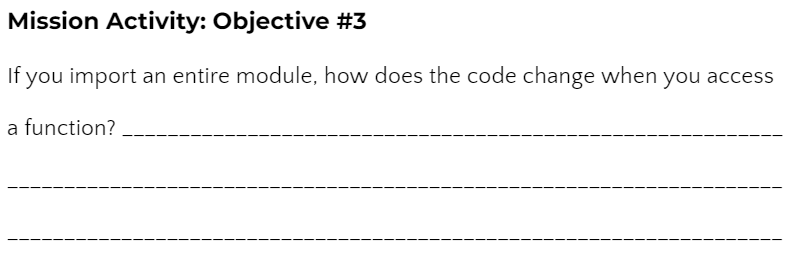
When you import the entire library, you must reference it when calling one of its functions.

* time.sleep(1)
* time.ticks\_ms()
* This returns the current time
* It returns a value, so the value needs to be assigned to a variable
* start\_time = time.ticks\_ms()



**DO THIS:**

* Go to your Mission Log and answer the question about importing a module

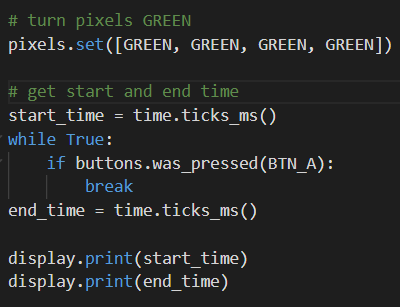
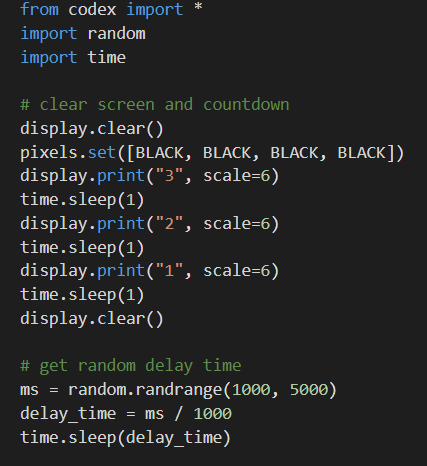


**Objective #3: Fun functions**

**DO THIS:**

* Change from   
  time import sleep to import time
* Change all the sleep(1) commands to time.sleep(1) commands
  + HINT: There are four sleep() commands

After the pixels turn GREEN:

* Assign **start\_time** the value from time.ticks\_ms()
* Wait until BTN-A was pressed
* Assign **end\_time** the value from time.ticks\_ms()
* Print **start\_time** and **end\_time**

**Objective #4: Time Differential**

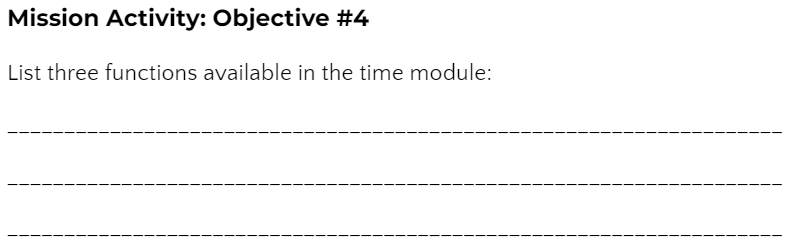
You have the **start\_time** and **end\_time**.

The reaction time is the difference of the two variables.

* You can just subtract the two:
  + reaction\_time = end\_time - start\_time
* OR use another time module function that finds the difference:
  + reaction\_time = time.ticks\_diff(end\_time, start\_time)

**DO THIS:**

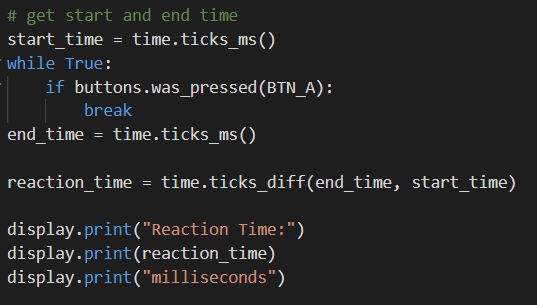
* Go to your Mission Log and answer the question about functions in the time module



**Objective #4: Time Differential**

**DO THIS:**

* Assign **reaction\_time** the difference between **end\_time** and **start\_time**
* Change the **display.print()** statements to print the **reaction\_time** instead of start\_time and end\_time



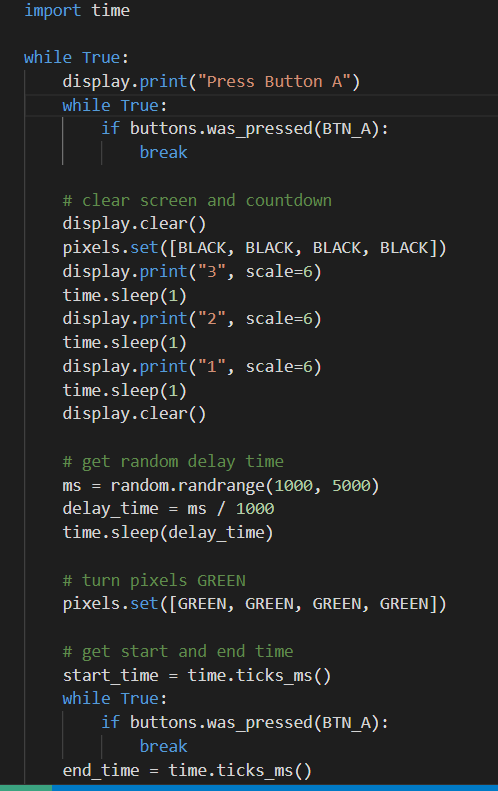
**Objective #5: Let’s Keep Playing**

Great job so far! The reaction game is fun, but what if you want to play more than once?

* Make the game wait for a button press, and then play again
* You will need an infinite loop with most of the code in it
* You will need to wait for a button press after displaying the reaction time
* You already have code for waiting for a button press, so you can copy and paste it

**Objective #5: Let’s Keep Playing**

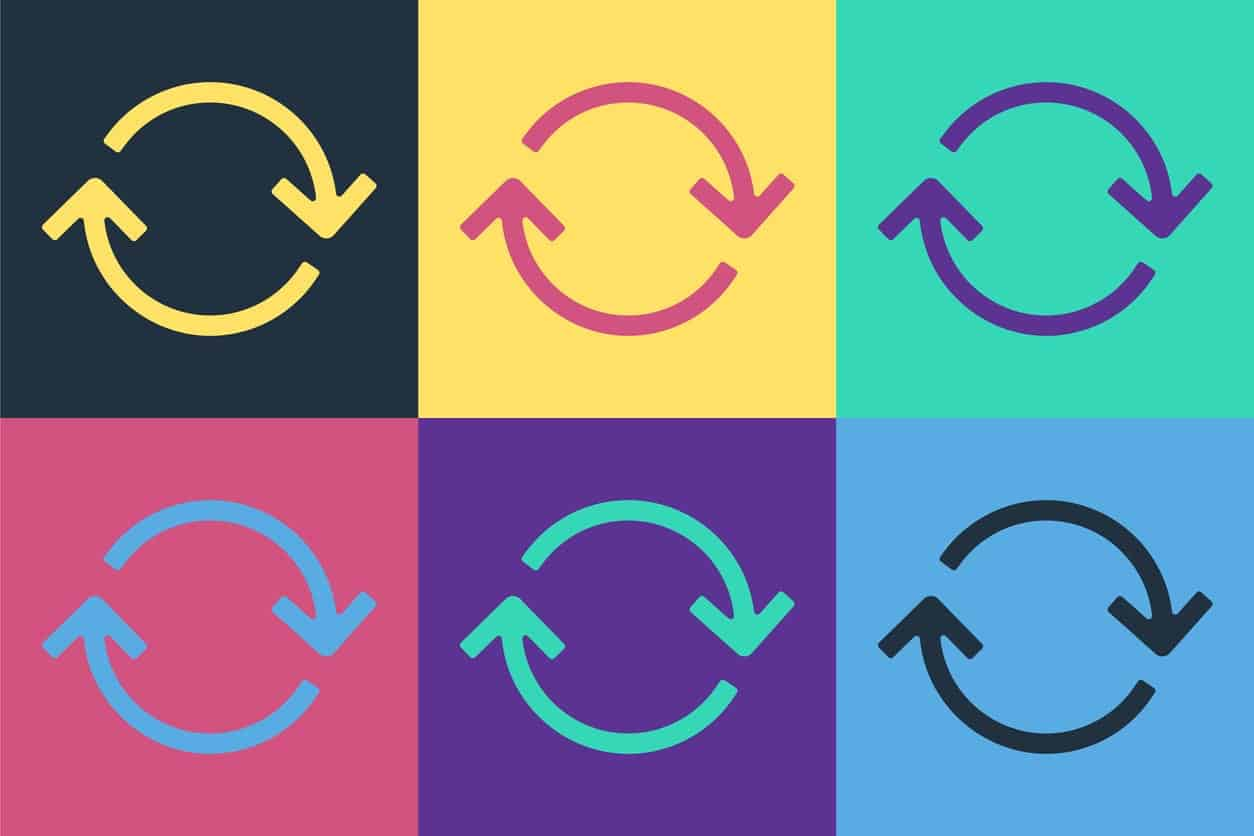
**DO THIS:**

* Add an infinite loop after the import statements
* Indent all the code inside the loop
* Add another wait loop at the beginning of the loop

**Objective #6: Reduce Repetition**

Take a look at your code. Do you notice a block of code that is repeated?

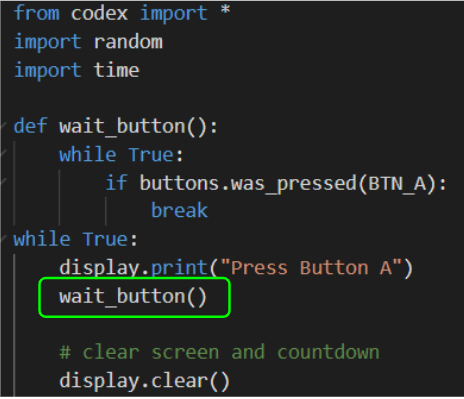
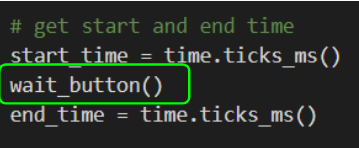
* You learned in Mission 9 that you can write a function instead of copy-paste or repeating code, you can write a function instead.
* There are two places in your code that wait for BTN-A to be pressed



**Objective #6: Reduce Repetition**

**DO THIS:**

* Write a **wait\_button()** function.
  + HINT: A function goes near the top of your code
* Delete the code that waits in the while loop.
* Call the **wait\_button()** function two times in the while loop.



**Mission Quiz: Quiz Timing**

Test your skills by **taking the quiz**.

**Objective #7: No Cheating**

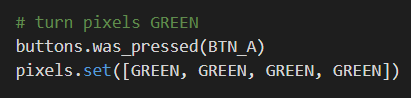
Fix a bug. Oh no! Players are pressing the button during the delay and getting ultra fast times.

* The buttons.was\_pressed() is always listening
* Even during the random delay
* Solve this problem by resetting the buttons.was\_pressed() just before starting the timer



**DO THIS:**

* Reset **buttons.was\_pressed(BTN\_A)** just before the pixels turn GREEN



**Mission Complete**

You have completed the tenth mission. 

**Do this:**

* Read your “Completed Mission” message
* Complete your Mission 10 Log
  + Post-Mission Reflection
* Get ready for your next mission!

**Wait! Before you go … Clear the CodeX**

Go to FILE -- BROWSE FILES

Select the “**Clear**” file and open it

Run the program to clear the CodeX

**Okay. Now you can go.**