ME30 Group Project

By: Victor Tolentino, Hector Barahona, Jose Hernandez, Ethan Hua

Project Proposal

Using the various inputs available through Circuit Playground Express and our group’s collective Python experience, we aimed to create a game inspired by the popular handheld toy, “Bop-it.” Our game utilizes the CPX board’s LEDs, speakers, buttons, touchpads, and accelerometer to create a fun and interactive experience. It works by providing instructions through the serial monitor in Mu editor and the circuit’s onboard speakers. First, the CPX speakers prompt the player to start the game by pressing A. Then, under the serial monitor, a message is printed that gives the player a random action to complete. The player must either shake the board, press buttons A or B, or touch a touchpad. Each action has a time limit, and failing to respond results in “Game Over.” Every successful action adds a point to the player’s score, and the goal of the game is to achieve the highest score possible.

Game Rules

- Actions
  - Press button A / B
  - Press a touchpad
  - Shake the circuit
- How to Play
  - Press button A to start
  - Follow the instructions (actions will be presented in the serial console)
  - Get as many points as possible
- Scoring
  - Each successful action adds a point and decreases the time for each turn
  - Game ends when the player fails to perform an action within the time limit

Victor’s Slide

Pseudocode

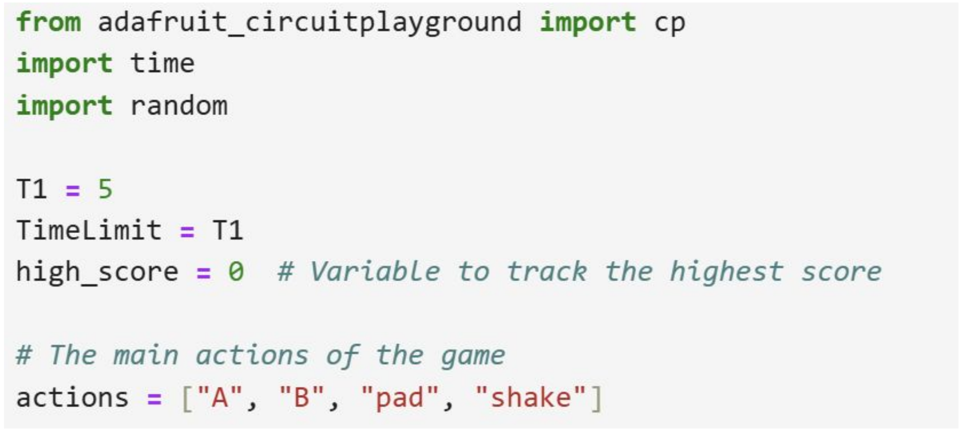
Libraries:

- Adafruit_circuitplayground enables the CPX Board and allows us to use the sensors on the board
- The time library is used for the LEDs and the timer during the game
- Random library is used to select a random action during the game

Variables:

- T1 is the initial time of the round (5 seconds)
- TimeLimit is set to T1 to be manipulated inside the maingame function
- high_score is used to store the highest score
- 4 possible actions for the game:
    - Press A, Press B, Touch a Pad, and Shake the Board

Code



```python
from adafruit_circuitplayground import cp
import time
import random

T1 = 5
TimeLimit = T1
high_score = 0  # Variable to track the highest score

# The main actions of the game
actions = ["A", "B", "pad", "shake"]
```

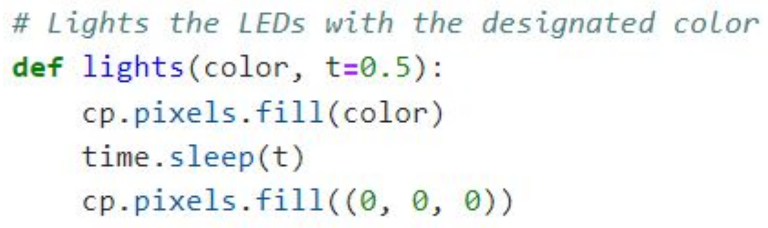
Ethan’s Slide

Pseudocode

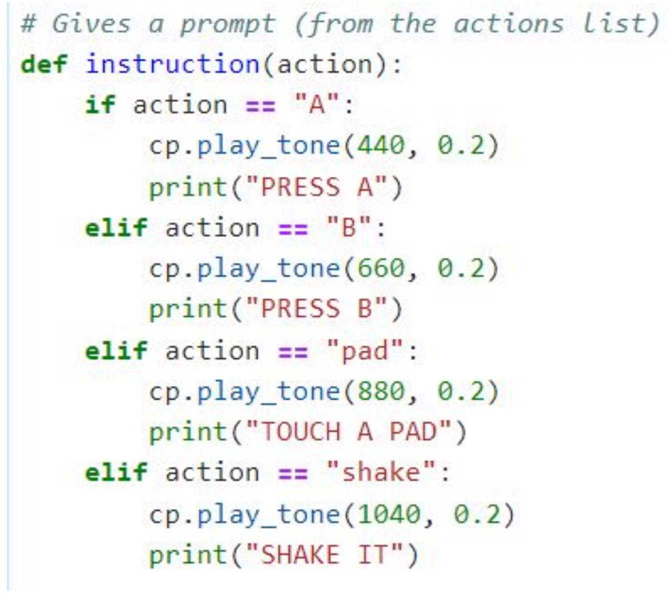
```
Function lights(color, duration = 0.5):
    ●    Set all LEDs to the specified color
    ●    Waits for the specified duration
    ●    Turns off all LEDs

Function instruction(action):
→If action equals "A":
    ●    Plays tone at 440 Hz for 0.2 seconds
    ●    Display "PRESS A"
→Else if action equals "B":
    ●    Plays tone at 660 Hz for 0.2 seconds
    ●    Display "PRESS B"
→Else if action equals "pad":
    ●    Plays tone at 880 Hz for 0.2 seconds
    ●    Display "TOUCH A PAD"
→ Else if action equals "shake":
    ●    Plays tone at 1040 Hz for 0.2 seconds
    ●    Display "SHAKE IT"
```

Code



```python
# Lights the LEDs with the designated color
def lights(color, t=0.5):
    cp.pixels.fill(color)
    time.sleep(t)
    cp.pixels.fill((0, 0, 0))
```



```python
# Gives a prompt (from the actions list)
def instruction(action):
    if action == "A":
        cp.play_tone(440, 0.2)
        print("PRESS A")
    elif action == "B":
        cp.play_tone(660, 0.2)
        print("PRESS B")
    elif action == "pad":
        cp.play_tone(880, 0.2)
        print("TOUCH A PAD")
    elif action == "shake":
        cp.play_tone(1040, 0.2)
        print("SHAKE IT")
```

Hector’s Slide

Pseudocode

#initializes the main part of the game as a function; maingame()
#global: changes made to the variable inside the function are changed outside of the function

<u>def maingame():</u>

- Make totpoints, TimeLimit, high_score global
- Variables
- Initialize TimeLimit and totpoints
- playing=True

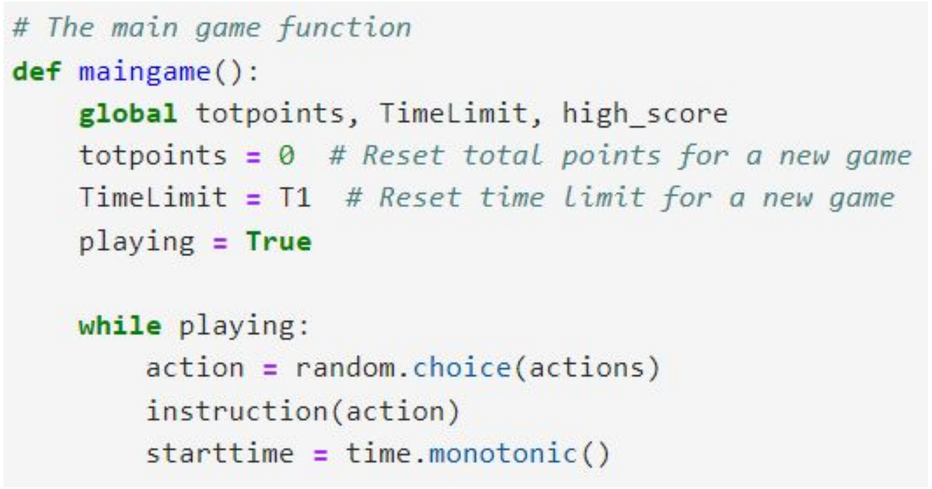
→ When playing:

  - Select an action from the actions list
  - Calls previous instruction function

#time.monotonic measures time that only moves forward (used to measure time elapsed)

  - Records current time using monotonic
  - Clock (reference time for player response)

Code



```
# The main game function
def maingame():
    global totpoints, TimeLimit, high_score
    totpoints = 0  # Reset total points for a new game
    TimeLimit = T1  # Reset time limit for a new game
    playing = True

    while playing:
        action = random.choice(actions)
        instruction(action)
        starttime = time.monotonic()
```

Hector’s Slide

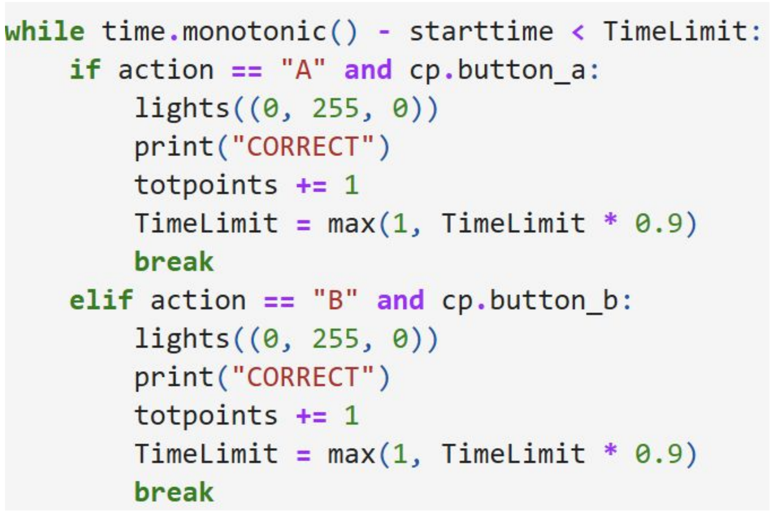
Pseudocode

#time elapsed is the current time - starttime
While time elapsed is less than the time limit:

- If the action is press A and A is pressed
- Call light function (green)
- Print CORRECT
- Add 1 point to the score
- Lowers the time limit 10% and never goes
- below 1 second (max picks largest)
- Break →return to the previous slide aka
- Back to the previous while loop

<u>REPEAT FOR EACH ACTION IN THE ACTIONS LIST</u>
#If action is (action) and action is completed

Code



```
while time.monotonic() - starttime < TimeLimit:
    if action == "A" and cp.button_a:
        lights((0, 255, 0))
        print("CORRECT")
        totpoints += 1
        TimeLimit = max(1, TimeLimit * 0.9)
        break
    elif action == "B" and cp.button_b:
        lights((0, 255, 0))
        print("CORRECT")
        totpoints += 1
        TimeLimit = max(1, TimeLimit * 0.9)
        break
```

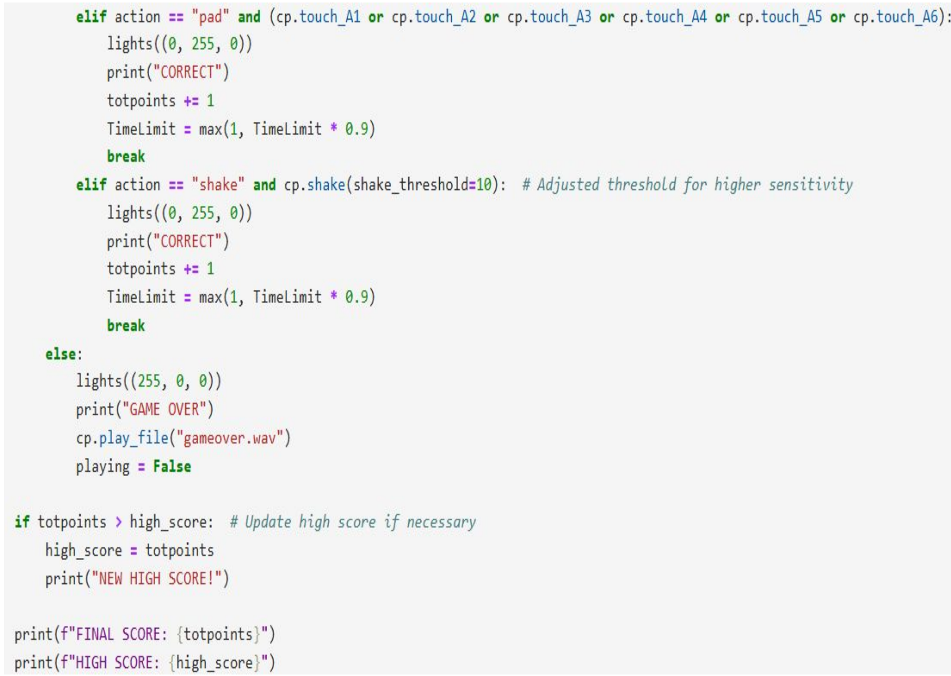
```python
while time.monotonic() - starttime < TimeLimit:
    if action == "A" and cp.button_a:
        lights((0, 255, 0))
        print("CORRECT")
        totpoints += 1
        TimeLimit = max(1, TimeLimit * 0.9)
        break
    elif action == "B" and cp.button_b:
        lights((0, 255, 0))
        print("CORRECT")
        totpoints += 1
        TimeLimit = max(1, TimeLimit * 0.9)
        break
    elif action == "pad" and (cp.touch_A1 or cp.touch_A2 or cp.touch_A3 or cp.touch_A4 or cp.touch_A5 or cp.touch_A6):
        lights((0, 255, 0))
        print("CORRECT")
        totpoints += 1
        TimeLimit = max(1, TimeLimit * 0.9)
        break
    elif action == "shake" and cp.shake(shake_threshold=10):  # Adjusted threshold for higher sensitivity
        lights((0, 255, 0))
        print("CORRECT")
        totpoints += 1
        TimeLimit = max(1, TimeLimit * 0.9)
        break
```

Jose’s Slide

Pseudocode

- Pad:
  The Action pad checks for touch sensors, Which turn on a green light and print the correct
- Shake:
  Action shake check: if the user shakes the cpx board using the sensitivity of 10, it will also turn green if you succeed and will print “correct”
- Game over:
  If the action does not match the pad or the shake, the game is over; then the light will turn red and play the game over the sound. It ends by setting the play variable to false
- Update the high score:
  Check for the player who got a new high score and update it, which will print the new high score.
- The score display:
  will show the high score along with the user score

Code



```
        elif action == "pad" and (cp.touch_A1 or cp.touch_A2 or cp.touch_A3 or cp.touch_A4 or cp.touch_A5 or cp.touch_A6):
            lights((0, 255, 0))
            print("CORRECT")
            totpoints += 1
            TimeLimit = max(1, TimeLimit * 0.9)
            break
        elif action == "shake" and cp.shake(shake_threshold=10):  # Adjusted threshold for higher sensitivity
            lights((0, 255, 0))
            print("CORRECT")
            totpoints += 1
            TimeLimit = max(1, TimeLimit * 0.9)
            break
    else:
        lights((255, 0, 0))
        print("GAME OVER")
        cp.play_file("gameover.wav")
        playing = False

if totpoints > high_score:  # Update high score if necessary
    high_score = totpoints
    print("NEW HIGH SCORE!")

print(f"FINAL SCORE: {totpoints}")
print(f"HIGH SCORE: {high_score}")
```

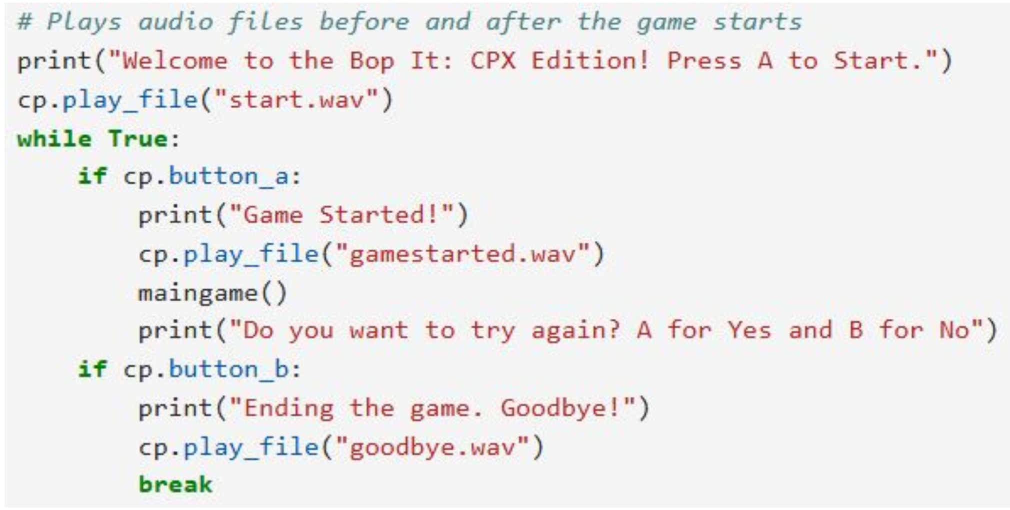
Victor’s Slide

Pseudocode

Audios:

- Prints Welcome message and plays audio file
- If the A button is pressed, it prints that the game has started, plays an audio file, and runs the maingame() function
- Once game ends, it prompts the user to either start a new game or end the game
- If the B button is pressed, it prints a message, plays the corresponding file, and exits the loop

Code



```
# Plays audio files before and after the game starts
print("Welcome to the Bop It: CPX Edition! Press A to Start.")
cp.play_file("start.wav")
while True:
    if cp.button_a:
        print("Game Started!")
        cp.play_file("gamestarted.wav")
        maingame()
        print("Do you want to try again? A for Yes and B for No")
    if cp.button_b:
        print("Ending the game. Goodbye!")
        cp.play_file("goodbye.wav")
        break
```

Thank You.
Questions?