Pong

Instructions:

1. Read the directions for each task very carefully before you begin working on it.
2. Every time the directions say something like "take a screenshot", do it and then copy it into the proper space in this document.

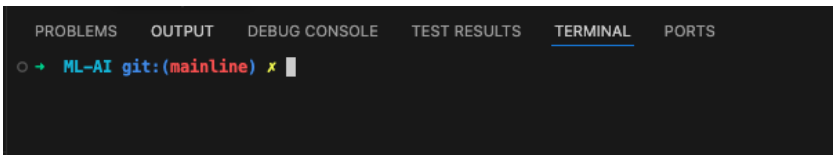
This worksheet is intended to guide you towards creating your first Python game using the `pygame` library. Assuming you accomplish all the problems successfully, you should, in the end, have a fully working version of Pong.

Problem 0: Visual Studio Setup

1. Download the `Pong.zip` file from Schoology
2. Unzip `Pong.zip` and put the resulting `Pong` folder into a special spot
    a. Suggestion: make a folder like `cs_workspace` on your Desktop and put `Pong` there
3. Go into the `Pong` folder and double click on `pong.code-workspace`
4. This should result in the program `Visual Studio Code` opening with several pong files for you to read/edit

Problem 1: Pygame Setup

1. Open `vscode`'s terminal window by, from the menu, selecting `Terminal` -> `New Terminal`. Upon clicking this, a new pane should appear in `vscode` that looks like the one below.



2. Prepare the workspace for installing libraries by running: `python3 -m venv .venv`

3. Install the required libraries by running: `python3 -m pip install pytest pygame`

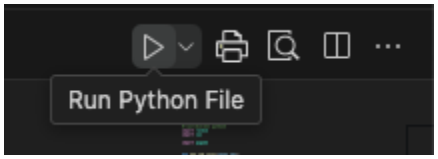
4. After the installation command finishes running, take a screenshot of the last few lines and paste it into the space below:

   ```

   ```

Problem 2: Verify it Runs

You now should be able to edit and run the program in `vscode`. To do this, open the `src/pong/main.py` file and click the "play" button at the top right (as seen in the picture below).



You should now see a paddle and a ball. Take a screenshot of the result and paste into into the space below:

```

```

Problem 3: Interpret

Now that everything is set up, you can start with understanding/developing the game. The file you will be editing is `src/pong/main.py`.

1. Looking at the code, what is the difference between `ball_x` and `ball_x_speed`?

   >

2. Explain in detail what the line `ball_x_speed = random.randint(2, 4)` does in terms of the game? Be sure to use the keywords “function” and “inputs”.

   >

3. Why is it important to have lines like `ball_x_speed = random.randint(2, 4)` and `ball_x = width // 2` outside the `while` loop?

   >

4. Pygame’s coordinate system is flipped over the y-axis (as seen here). How does knowing this change your understanding of the line `paddle_y = paddle_y - 5`?

   >

5. What line of code is responsible for detecting when the ball should bounce off the right side of the screen? Briefly explain what is happening.

   >

6. Once a collision does happen, what line of code is actually responsible for making the ball go in the opposite direction (aka bounding)? Briefly explain what is happening.

   >

7. Why can the paddle move up when the up arrow is pressed but not down when the down arrow is pressed?

   >

Problem 4: Modify

Now that you understand the code a little bit, work to accomplish each of the following tasks in order:

1. Have the ball move up/down and left/right.

- Hint: examine the `ball_x_speed` and `ball_y_speed` variables

2. Change the color of the ball from white to blue.
    - Hint: change the `get_rgb_color()` function.

3. Have the ball bounce off the top, bottom, and right walls. Again, note that the coordinate system pygame uses is a little weird.
    - Hint: examine how the ball detects/reacts to the right wall and copy the logic

Problem 5: AI Help

AI is a very powerful tool when writing a program. Unfortunately, like all tools, how you use it determines whether that power is put to good or to ill. It can be good if it helps you learn: almost like a bridge. It can be bad, however, if you become overly reliant on it: almost like cheating on an exam you're giving yourself.

In this section, you can use AI (preferably claude.ai) to help you when you get stuck. Then, you'll reflect on your experience and see whether you were building bridges or cheating yourself. In my opinion, **learning how to use a ChatBot to help you tackle difficult, open-ended problems is just as important as the actual material in this worksheet**.

1. Have the ball bounce off the paddle. This is the hardest part of programming the game. The way I think about it is the following: *if* the ball is to the left of the right side of the paddle **AND** to the right of the left side of the paddle **AND** to below the top of the paddle **AND** above the bottom of the paddle, then it's hitting the paddle. You can represent all this with a complex `if` statement that uses three `and` connectors.

2. Have a "score" variable that shows how many hits in a row the user has had.
    - Hint: research `pygame.font.Font`

3. Reset the game and the "score" if the player misses the ball and it touches the left wall.

4. Take a screenshot of your game and paste it into the space below:

    ```
    ```

```

5. In 2-3 sentences, describe your experience with the ChatBot.

   >

6. Do you think you could redo this section from scratch without the aid of a ChatBot? How do you know?

   >

7. What are two or three signals that someone is becoming overly reliant on the ChatBot and not actually learning?

   >

8. You can ask a ChatBot to adopt various roles such as tutor or coach. Here is a list of possible prompts that you can use to start your conversation with AI to increase the chance you're learning and not "cheating yourself". Which one do you think is a good one to use for future assignments like this? Why?

   >

# Problem 6: Make

Pick two of the following three extensions and add them to your game:

1. Two-player pong with a left AND right paddle
2. Have the ball become a new, "random" color every time it hits a paddle
3. Have the ball bounce off the paddle at an angle dependent on where it hits the paddle

Once you are done, copy **ALL** of the source code for `main.py` and paste it into the space below:

```

```
```

Problem 7: Reflection

1. What does the `pip3 install pygame` command do?

   >

2. What was the most difficult part of programming the game?

   >

3. PRIMM is a common paradigm for learning code. It stands for “predict, run, interpret, modify, make”. How did following this paradigm help you scale to the point where you could complete the game?

   >

4. “Learning with ChatBots requires discipline”. In 4-5 sentences explain why this is true and some techniques you will use to prevent yourself from being overly reliant on this tool.

   >

5. Did you notice any obvious errors with this worksheet that should be fixed for future students? If so, what were they?

   >

6. Were there things in the worksheet that were overly confusing and that you think could be improved? If so, what were they?

   >