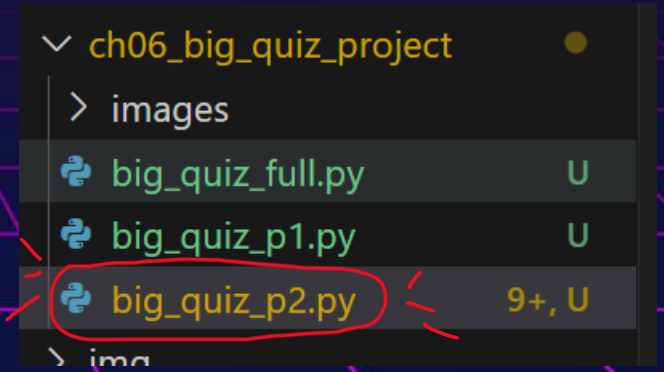
6.2
Big Quiz Project

Part 2 of 3

6.2 Big Quiz Project

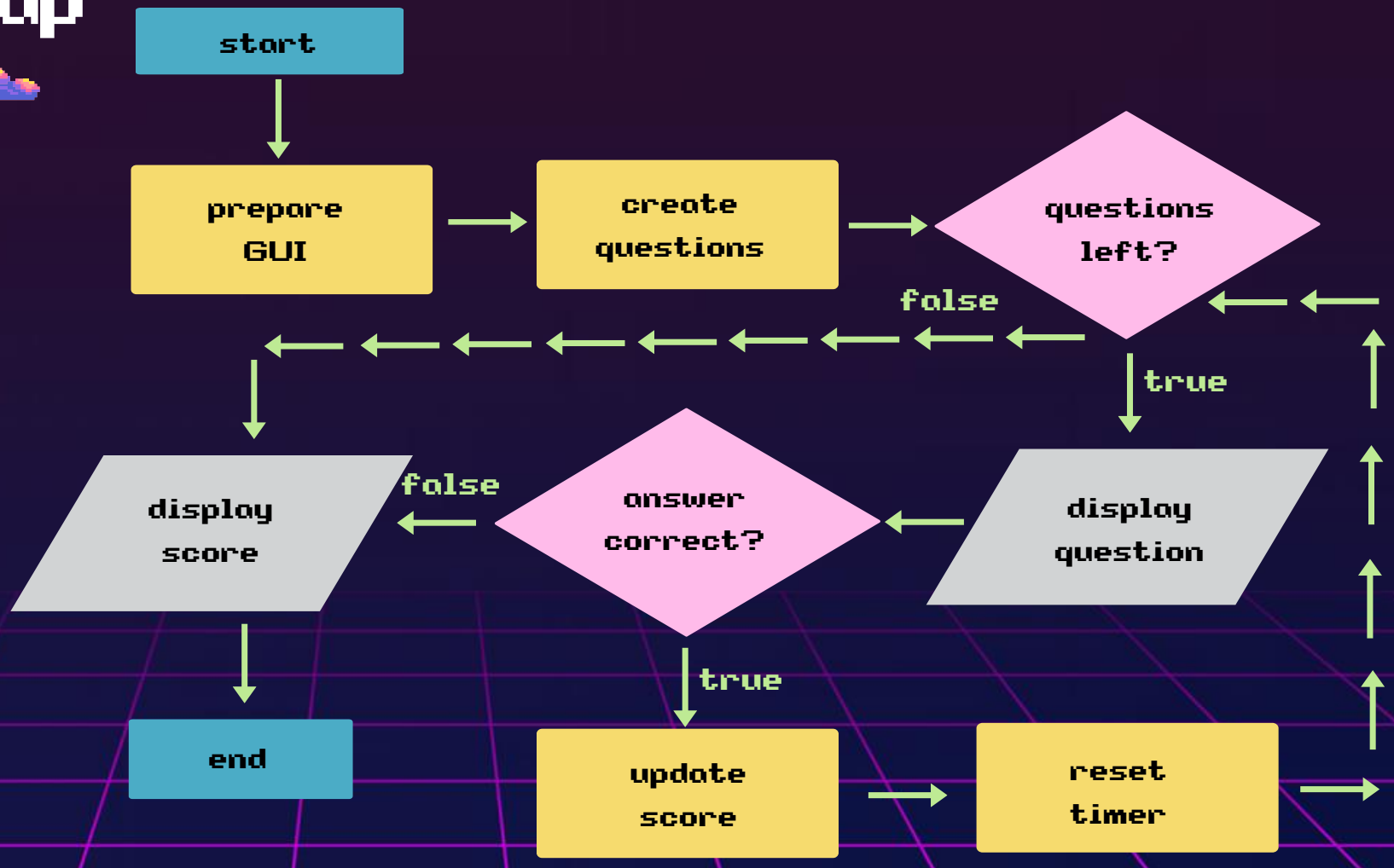
It does not matter how well you did in Part 1. You all start from the same place!

- **Revise the flowchart:** We will do a quick recap of the flowchart.
- **Follow the recipe:** We will provide you with a clear outline to complete the project.
- **Support is available:** Refer to your notes, previous class code, or search online.
- **Remember:** Your teacher is here to assist you.
- **Sample code:** You will **not** start from scratch this time. You will start from a code sample. You should find it in your project.



Flowchart Recap

	shuffle questions
	final level?



6.2 Big Quiz Project

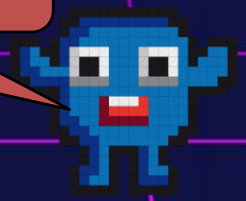
There is a catch!

- When you run the sample code you will notice that it does not work!
- This can feel scary, but do not worry, it is all part of your assessment.
- Dealing with **simple errors** is one of the skills we will assess.

```
/Microsoft/WindowsApps/python3.11.exe c:/Users/erikacamilleri/Document
big_quiz_project/big_quiz_p2.py
pygame 2.5.2 (SDL 2.28.3, Python 3.11.9)
Hello from the pygame community. https://www.pygame.org/contribute.htm
Traceback (most recent call last):
  File "c:\Users\erikacamilleri\Documents\GitHub\code-games-in-python\
y", line 141, in <module>
    pgzrun.go()
  File "C:\Users\erikacamilleri\AppData\Local\Packages\PythonSoftwareF
\LocalC                                    311\site-packages\pgzrun.py", line 31
    run
  File "C:\Users\erikacamilleri\AppData\Local\Packages\PythonSoftwareF
\LocalCache\local-packages\Python311\site-packages\pgzero\runner.py",
    PGZeroGame(mod).run()
  File "C:\Users\erikacamilleri\AppData\Local\Packages\PythonSoftwareF
\LocalCache\local-packages\Python311\site-packages\pgzero\game.py", li
    self.mainloop()
  File "C:\Users\erikacamilleri\AppData\Local\Packages\PythonSoftwareF
\LocalCache\local-packages\Python311\site-packages\pgzero\game.py", li
    draw()
```

Open file in editor (ctrl + click)

Oh no! Will we ever reach safety again?



What do we do now?

There are two kinds of errors: _____ and _____.

Logical errors are small _____ that make our program behave unreliably.

Syntax errors disrupt code translation and so the program cannot _____ .

bugs

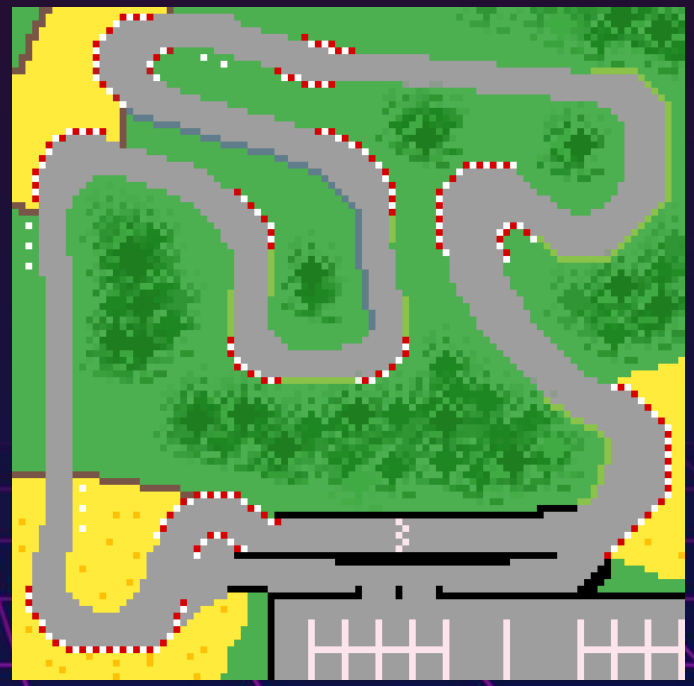
logical

run

syntax

Plan of action

- First, you must get to a safe point where the code can reliably run.
- Then, you can complete the screen which displays a question and four answer options.
- You will complete the code so that the screen updates at one second intervals to show the time decreasing.
- When the time runs out the screen will display a game over message along with the user score.

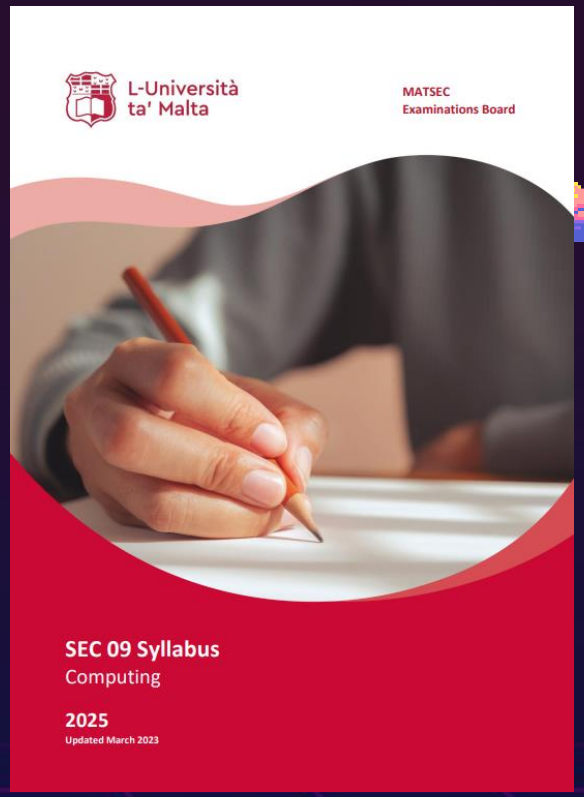


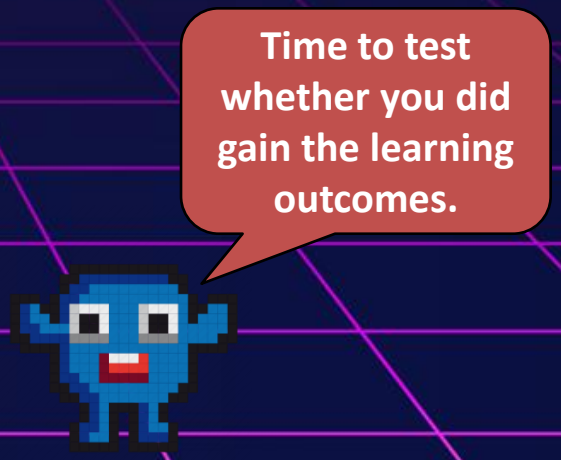
Over to you from here

- **Start from a sample:** You'll begin by going through the sample code and be familiar with the exercises marked as **TODO**.
- **Easy tasks:** Identify the parts that you think you can do easily.
- **Just write code:** It is alright if your code does not work as you wish. You can still get a lot of marks.

```python
next_question = questions.pop(0)

# TODO: set variables to store the player's score and time_left
# ... your code ...

# ---- define functions --------------------
def draw():
    screen.fill("dim gray")
    screen.draw.filled_rect(question_box, "sky blue")
    screen.draw.filled_rect(timer_box, "sky blue")
    for box in answer_boxes:
        screen.draw.filled_rect(box, "orange")

    # draw text for the time_left
    # TODO: time_left is a number but it is not supported by draw
    #       it needs to be converted to a string
    screen.draw.textbox(time_left, timer_box, color=("black"))

    # draw text for the next_question["q"] which is the question itself
    screen.draw.textbox(next_question["q"], question_box, color=("black"))

    # TODO: loop through each answer box and draw the respective answer option
    # ... your code ...

def handle_game_over():
    global next_question
```

Assessment Criteria	
1. Application of Python Skills in a Scenario	
Use of output statements with string concatenation	3
Appropriate use of variables including type conversion	3
Able to access data items from a list	2
Able to access data items from a dictionary	3
Appropriate use of user-defined functions	3
Appropriate use of nested decision/iteration statements	6
Appropriate use of the Pygame Zero module	2
2. Game Functionality	
Timer automatically decreases at 1 second intervals	3
When the time runs out the final screen is displayed	3
3. Programming Practices	
Abide by Python programming style conventions	1
Descriptive commenting	1
Identify and fix errors in the program independently	3
Total	**33**





Follow the recipe

- We have given you instructions on how to complete the game.
- You have **6 small tasks** to complete in this part.
- Read the instructions carefully and implement them as best as you can.
- You may always refer to code we wrote in class and search a little bit on the Internet.

6.1a Set score and timer

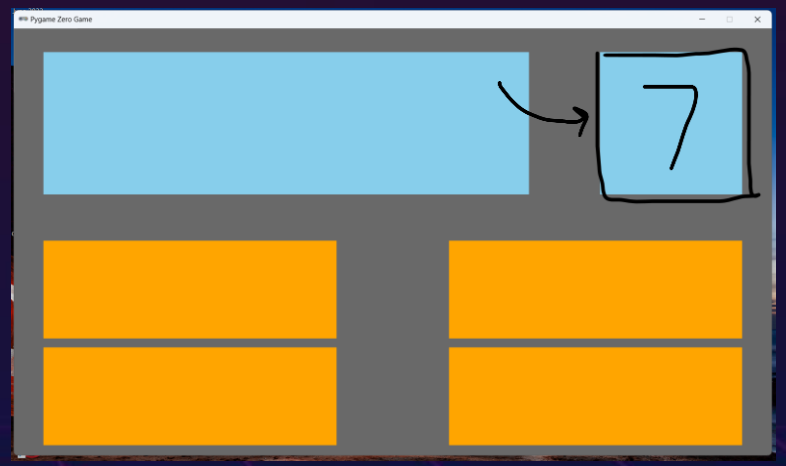
- You need to start thinking about how the game will work.
- Create a variable to hold the **score** and set it to zero.
- You also need to create a variable for the **timer** that will store the number of seconds the player has left to answer each question.
- You can give them ten seconds.

This is a basic skill which you should complete easily. Have a look at an example here.

```
next_question = questions.pop(0)

# TODO: set variables to store the player's score and timer
# ... your code ...
```

6.1b Draw the time left

- Update the **draw()** function so that it displays the time left in the timer box.
- We have already included the line of code that is supposed to do this.
- But time left is a **number** and the function to draw inside the timer box works with a **string**.

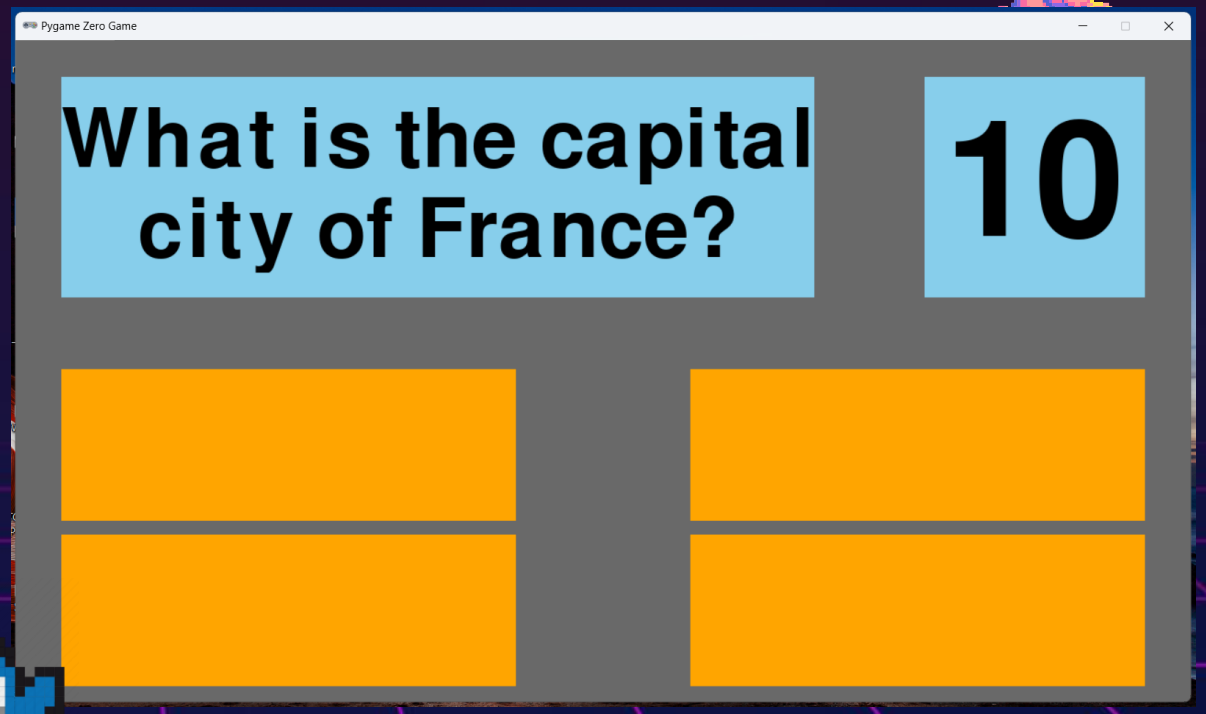


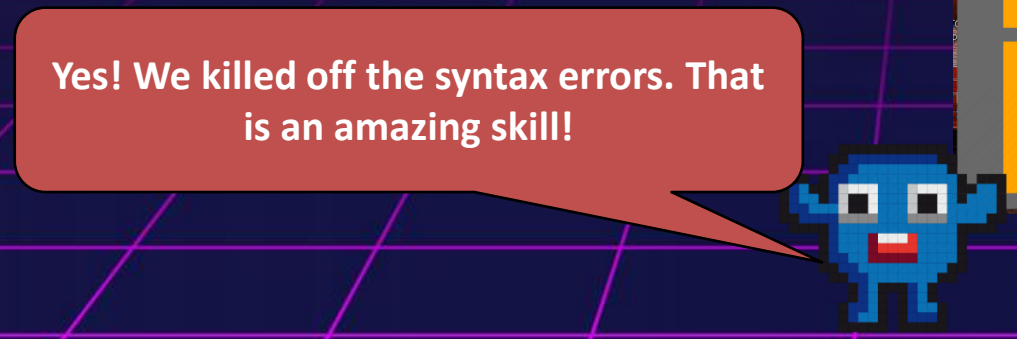

```
# draw text for the time_left
# TODO: time_left is a number but it is not supported by draw
#       it needs to be converted to a string
screen.draw.textbox(time_left, timer_box, color=("black"))
```

This is a basic skill which you should complete easily. Have a look at an example here.

6.1c Do you have syntax errors?

- If you completed the previous exercises well then you should have reached safety. Phew!
- Run your code and test that some text is being displayed on screen.



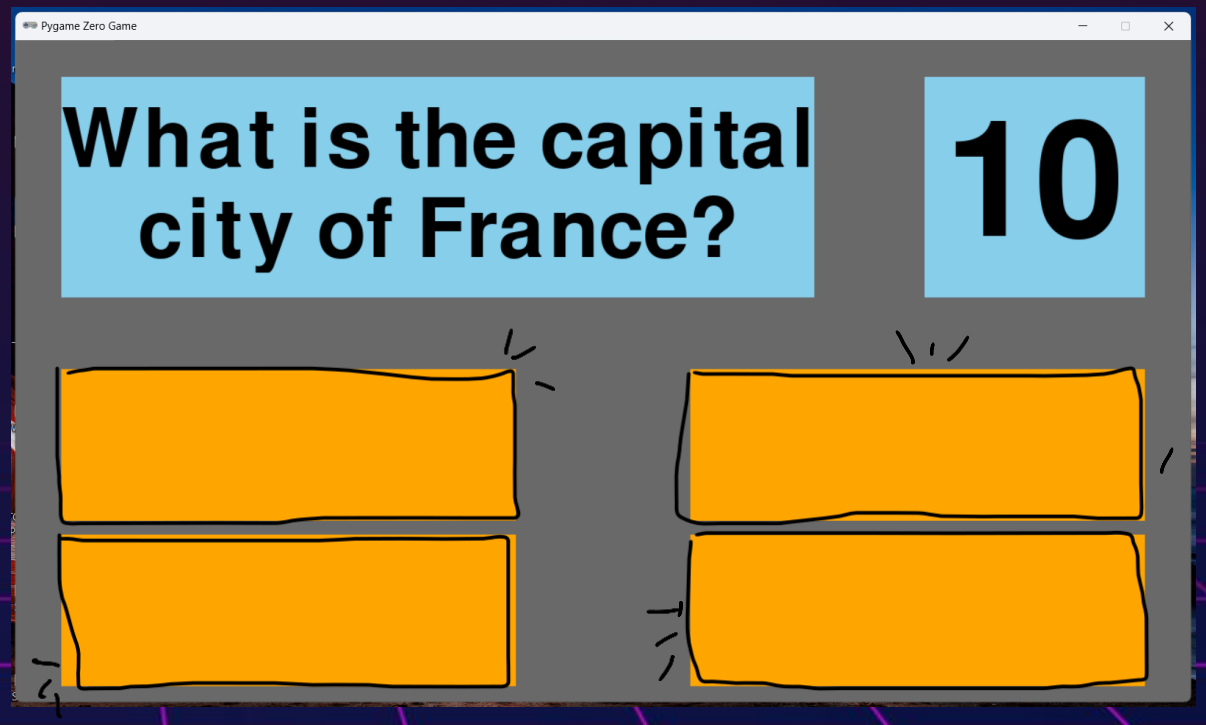


Are you stuck?

If you are finding it difficult to proceed, please use a help token and your teacher will give you more guidance.

6.1d Draw the answer options

- Update the **draw()** function again so that it displays the answers inside the answer boxes.
- You must extract the answers from the **next question** which is a **dictionary**.
- You must use **`screen.draw.textbox`** by Pygame to draw text in an **answer box**.



6.1d Draw the answer options

- It is alright if you need to get warmed up before tackling a challenging piece of code.
- Why don't you practice working with a question dictionary here?
- When you feel ready find the TODO in the sample code.

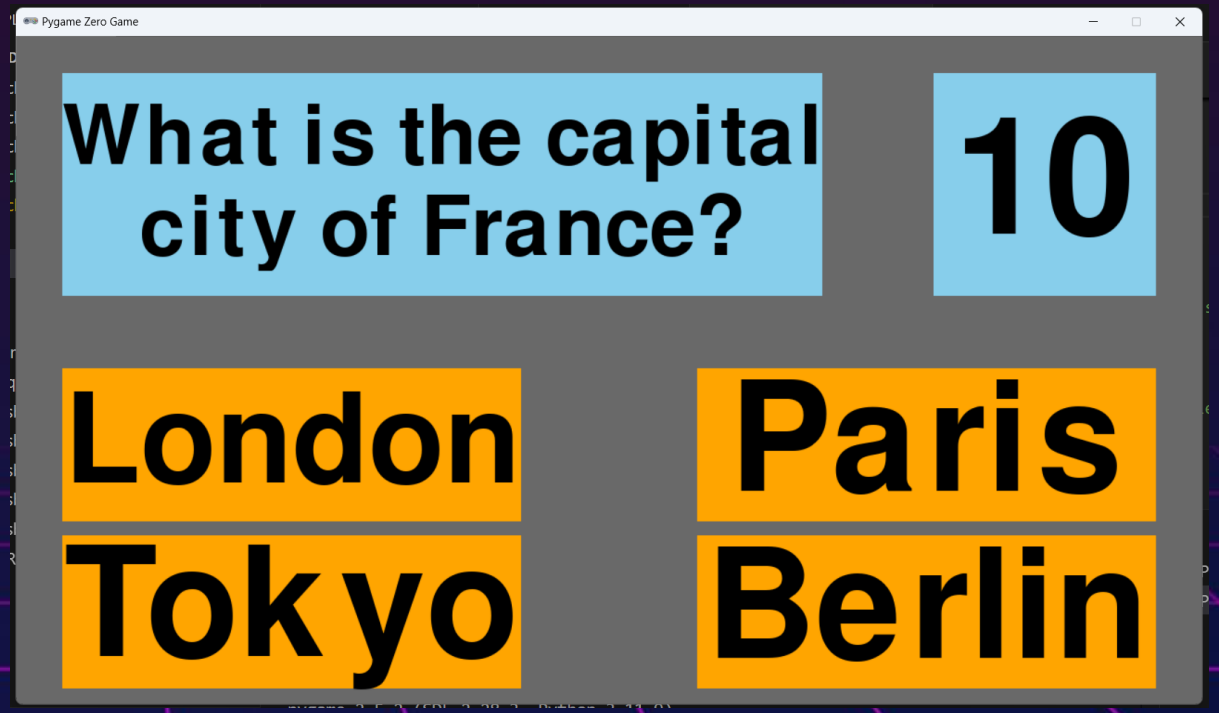
```
q1 = {
    "q": "What is the capital city of France?",
    "o": ["London", "Paris", "Tokyo", "Berlin"],
    "a": 1
}
// TODO: Get the options list from q1

// TODO: Loop through the four options
```

```
# draw text for the next_question["q"] which is the question itself
screen.draw.textbox(next_question["q"], question_box, color=("black"))

# TODO: loop through each answer box and draw the respective answer option
# ... your code ...
```

When you do the real task, remember that you can always copy existing code and adapt it to suit your needs.

6.1d Draw the answer options

- You can test your code at this stage by running it.
- You should see a screen complete with text.

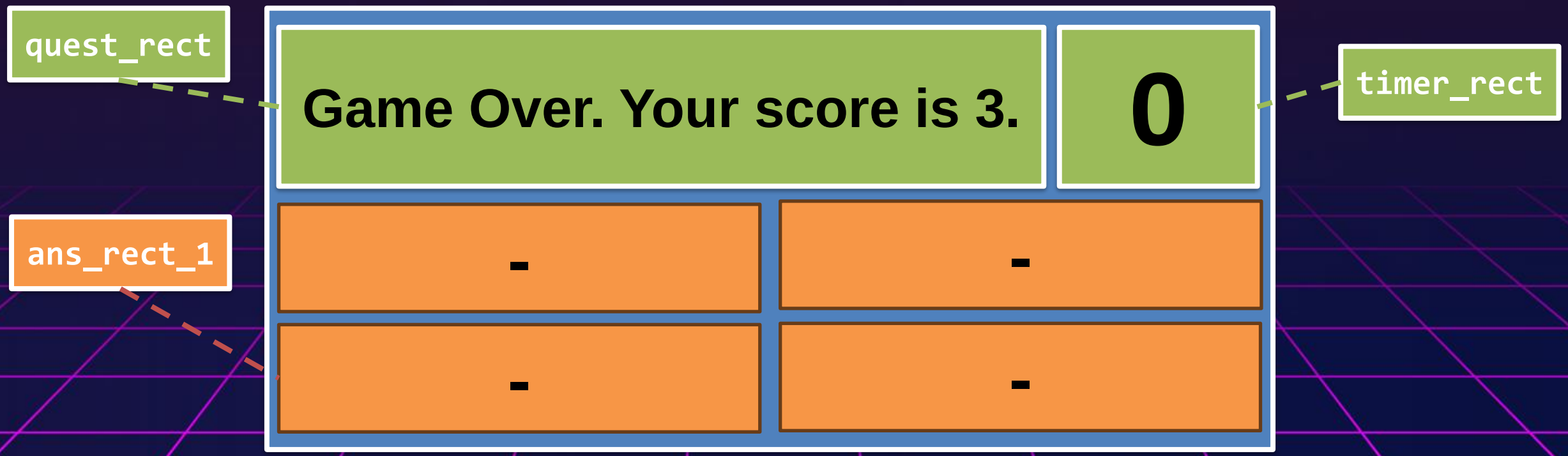


Are you stuck?

If you are finding it difficult to proceed, please use a help token and your teacher will give you more guidance.

6.1e Handle Game Over

How should the game look like when it is over?



6.1e Handle Game Over

- Complete the code that handles the end of the game.
- Create a string variable that will store the game over **message** along with the score.
- The **`next_question`** is a dictionary which will store the message instead of a question.
- At the end of the game, the **time left** is set to 0.

```python
def handle_game_over():
    global next_question
    # TODO: set a string variable that stores a message to the user along with th
    #       e.g., "Game over. Your score is 3."
    # ... your code ...

    # TODO: modify the following dictionary so that "q" maps to the message varia
    next_question = {
        "q": "Game over. Your score is 3.",
        "o": ["-", "-", "-", "-"],
        "a": -1
    }

    # TODO: update the timer and set it to 0
    # ... your code ...
```

This task has multiple steps, but they are all quite easy to complete. You can do this!

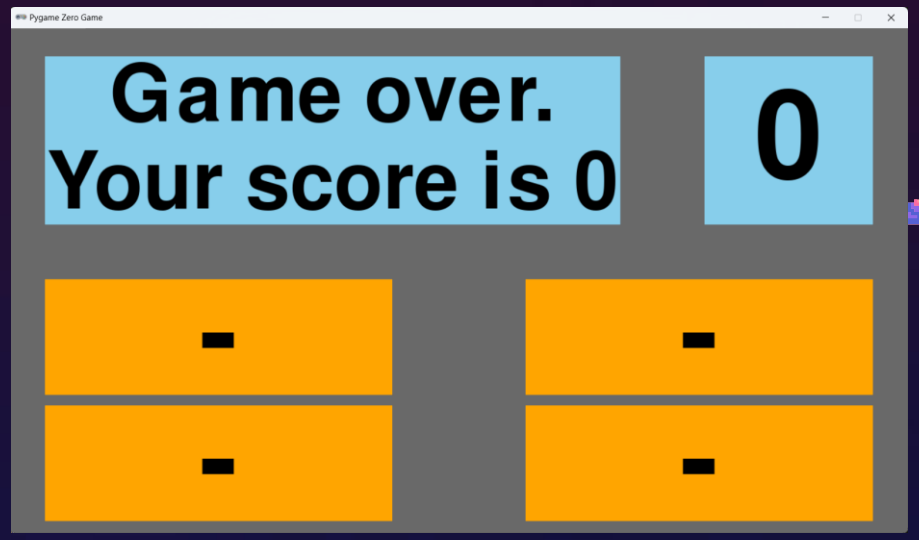
6.1f Update the timer

- We have already included a specific line of code that calls the **update_time_left()** in the background.
- Right now, the function does not do anything.
- You must complete the code such that if there is time left then the timer is decreased by **one**.
- When time runs out you need to call the **handle_game_over()** function.

```
def update_time_left():
    global time_left
    # TODO: if time_left is greater than 0
    #       then decrease time_left by 1
    #       else handle_game_over()
    # ... your code ...
```

Time to test!

- If you have completed all the exercises as instruct, then it is time to test your code.
- Run your code and ensure that there are no **syntax errors**.
- It is also a good idea to review the assessment criteria and make sure that your code is easy to correct by a teacher.



At this stage, you still cannot play the game. But you will see the screen update.

You have leveled up!

You are making great strides. Well done.