1. Capture and paste two program code segments you developed that contain a list (or other collection type) being used to manage complexity. The first segment must show how data has been stored in the list. The second must show the data in the same list being used, such as creating new data from the existing data or accessing multiple elements in the list, as part of fulfilling the program's purpose.

```
player = 0
SPACE = -1
board = [SPACE for _ in range(9)]
def valid_move(index: int) -> bool:
   if board[index] == SPACE:
      return True
   else:
      return False
```

The code uses the board list to represent the state of the tic-tac-toe board. The valid_move function utilizes this list to check the validity of a move by checking the board represented by the given index.

2. Identify the name of the list being used in this response.
The name of the list being used is "board".

3. Describe what the data contained in the list represents in your program.
The "board" list represents the space that isn't occupied on the tic-tac-toe board by either player.

4. Explain how the selected list manages complexity in your program by explaining why your program could not be written, or how it would be written differently, if you did not use the list.

The list could be written differently to say

```
for i in range(9)
        board = SPACE
```

Instead of
[SPACE for _ in range(9)]