Python Mini Project 1 – CLI Robot | By Cameron Rosenthal

Design and write a simple program that takes user input, and moves a virtual robot across a 2D array.

Create a Python main function:

```
def main():
   print("Hello World!")

if __name__ == "__main__":
   main()
```

Create a constant**, BOARD_SIZE**, and initialize it with a tuple **(5, 5)**.

Create a constant, **BOARD_ICON**, and initialize it to a character/string of your choice. Ex: ‘ - ’

Create a constant, **ROBOT_ICON**, and initialize it to a character/string of your choice. Ex: ‘ :[]: ’

Create a constant, **ROBOT_START_LOCATION**, and initialize it to a tuple containing a valid index in regards to the **BOARD_SIZE**. Ex: (0,0)

Note: Another choice is to use a dictionary to store these values instead of using constants. Ex:

```
BOARD_INFO = {
   “board_size” : (5, 5),
   “board_icon” : ‘ - ’,
   “robot_icon” : ‘ :[]: ’,
   “robot_start_location” : (0, 0)
}
```

Create a local variable within main(), **current_robot_location**, and initialize it to **ROBOT_START_LOCATION**.

Create a 2D array named **board**, define a function, **init_board()**, that initializes the 2D array to size **BOARD_SIZE**, and assigns each index to **BOARD_ICON**. Additionally, this function will put the **ROBOT_ICON** on the board at **ROBOT_START_LOCATION**.

Define a **show_board()** function to print out the 2D array as an n x n board.

Define a **show_menu()** function to print a menu:

1 – Forward.

2 – Backwards.

3 – Left.

4 – Right.

0 – Exit.

Write a **while loop** that contains the necessary code to show the menu and get user input.

Write the logic to take user input, and move the robot icon one step on the board in relation to what the user chooses in the menu.

Error check user input: You can use int(input("Example Prompt")) inside a **try-catch** statement to check for correct type, and an **if-statement** to check for menu bounds.

Use a **switch statement** or **if-else** to handle menu logic.

Use a **generalized function** for robot movement. You should not need to write a function for each movement direction.

Error check robot movement requests, so as to not move outside the 2D array.

Note: Make sure to update **current_robot_location** each time the robot is moved.