A python project on

Car Logic Simulator

Submitted by: Mohit Topiya

Table of Contents

1. Introduction

This project involves the development of a Python-based Driving Simulator that mimics real-world driving scenarios. The simulator allows users to interact through commands to perform actions such as pressing the clutch, shifting gears, accelerating, braking, applying the handbrake, and turning the ignition on or off. It includes safety checks and realistic behaviour, ensuring that the user follows proper driving procedures. The simulator also incorporates features to monitor speed, gear limits, and prevents unsafe operations, providing an educational and engaging experience for users to learn driving dynamics.

.

2. Objective

The primary objectives of this project are:

1. To simulate realistic driving controls in a Python environment.
2. To ensure user safety through logical checks and constraints.
3. To provide feedback for correct and incorrect operations.
4. To simulate speed limits based on gear selection.
5. To educate users about gear-shifting dynamics and driving safety.

3. Features

1. Ignition Control:
    - Turn the ignition ON and OFF while ensuring proper prerequisites are met (e.g., gear in neutral).
2. Handbrake Application and Release:
    - Prevent accidents by enforcing handbrake rules.
3. Gear System:
    - Shift gears up and down sequentially, with speed limits for each gear.
    - Includes reverse gear functionality.
4. Speed Control:
    - Accelerate and decelerate within gear speed limits.
5. Safety Mechanisms:
    - Prevent actions such as shifting to reverse while moving forward.
    - Trigger warnings for improper actions.
6. Interactive Commands:
    - Users can control the car using simple commands (e.g., 'i' for ignition, 'a' for acceleration).
7. Maximum Speed Indicator:
    - Notify the user when the top speed in the highest gear is achieved.
8. Exit Protocol:
    - Ensure the car is stopped, the handbrake is applied, and ignition is OFF before quitting.

4. Code Structure

The program utilizes a while loop to accept continuous user input for commands. Key variables track the car's state, including:

1. Ignition on: Tracks whether the ignition is ON or OFF.
2. Handbrake released: Tracks the handbrake status.
3. Clutch pressed: Tracks if the clutch is pressed.
4. gear: Tracks the current gear (0 for neutral, -1 for reverse).
5. speed: Tracks the car's speed.

Each command triggers specific checks and actions, ensuring realistic and safe interactions. Speed limits for gears are predefined in a dictionary for scalability.

5. Challenges Faced and Solutions

1. Handling Invalid Sequences:
    - Challenge: Users attempted to accelerate without proper prerequisites.
    - Solution: Added detailed checks for ignition, handbrake, and gear status.
2. Gear Restrictions:
    - Challenge: Users shifted multiple gears without adhering to sequence.
    - Solution: Restricted gear changes to one step up or down at a time.
3. Exit Conditions:
    - Challenge: Users exited the simulator abruptly without following a proper shutdown sequence.
    - Solution: Mandated car stop, handbrake application, and ignition OFF before quitting.

6. Final code

```
def driving_simulator():
    print("Welcome to the Driving Simulator!")
    print("Controls:")
    print(" - Press 'c' for clutch")
    print(" - Press 'i' to turn on the ignition")
    print(" - Press 'I' to turn off the ignition")
    print(" - Press 'h' to release the handbrake")
    print(" - Press 'H' to apply the handbrake (Warning: Pressing while
driving will cause an accident)")
    print(" - Press gear numbers (1-5) with clutch to gear up or down")
    print(" - Press '0' with clutch for neutral")
    print(" - Press 'a' to accelerate")
    print(" - Press 'b' to brake")
    print(" - Press 'r' for reverse gear")
    print(" - Press 'q' to quit (Only after turning off ignition and applying
handbrake)")

    ignition_on = False
    handbrake_released = False
    clutch_pressed = False
    gear = 0  # Neutral
    speed = 0
    reverse = False

    # Speed limits for each gear
    gear_speed_limits = {1: 20, 2: 40, 3: 60, 4: 80, 5: 120}

    while True:
        command = input("Enter command: ")

        # Ignition ON
        if command == 'i':
            if ignition_on:
                print("Ignition is already ON.")
            elif gear != 0:
                print("Please shift to neutral before turning on the
ignition.")
            elif not handbrake_released:
                print("Release the handbrake before turning on the
ignition!")
            else:
                ignition_on = True
                print("Ignition turned ON. Engine has started and is ready to
drive!")

        # Ignition OFF
        elif command == 'I':
            if not ignition_on:
                print("Ignition is already OFF.")
            elif speed > 0:
```

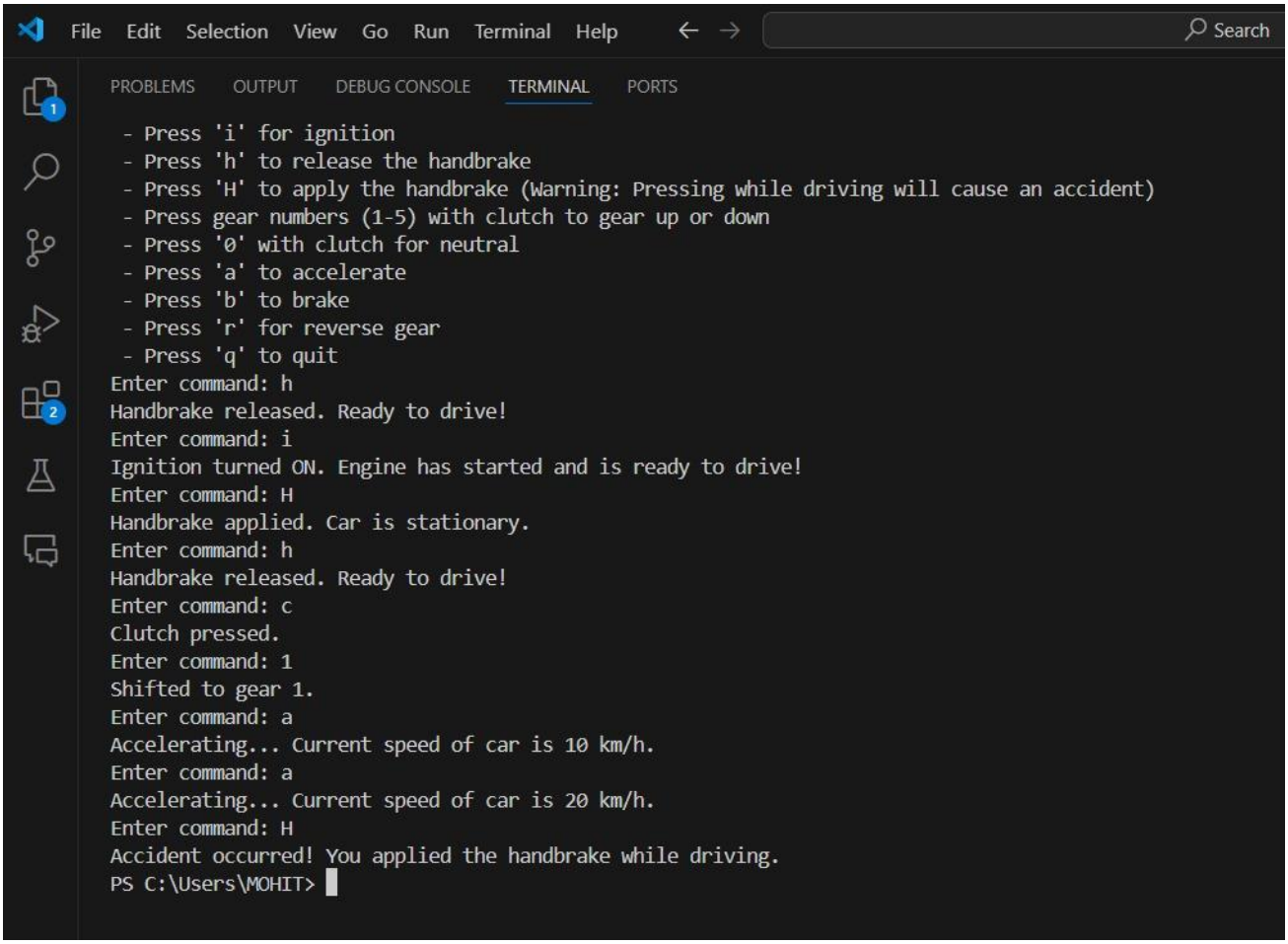
```
                print("Please stop the car completely before turning off the
ignition.")
            elif handbrake_released:
                print("Apply the handbrake before turning off the ignition.")
            else:
                ignition_on = False
                print("Ignition turned OFF. You can now quit safely.")

        # Handbrake Release
        elif command == 'h':
            if speed > 0:
                print("Accident occurred! You pressed the handbrake while
driving.")
                break
            handbrake_released = True
            print("Handbrake released. Ready to drive!")

        # Handbrake Apply
        elif command == 'H':
            if speed > 0:
                print("Accident occurred! You applied the handbrake while
driving.")
                break
            handbrake_released = False
            print("Handbrake applied. Car is stationary.")

        # Clutch
        elif command == 'c':
            clutch_pressed = True
            print("Clutch pressed.")

        # Gear Change
        elif command in ['1', '2', '3', '4', '5'] and clutch_pressed:
            new_gear = int(command)
            if reverse:
                print("Cannot shift to forward gear while in reverse.")
            elif abs(new_gear - gear) > 1:
                print("You can only shift up or down by one gear at a time.")
            elif gear == 0 or new_gear > gear:  # Neutral or gear up
                if speed <= gear_speed_limits[new_gear]:
                    gear = new_gear
                    reverse = False
                    clutch_pressed = False
                    print(f"Shifted to gear {gear}.")
                else:
                    print(f"Reduce speed to below
{gear_speed_limits[new_gear]} km/h before shifting to gear {new_gear}.")
            elif new_gear < gear:  # Gear down
                gear = new_gear
                clutch_pressed = False
                print(f"Shifted to gear {gear}.")

        # Reverse Gear
        elif command == 'r' and clutch_pressed:
            if speed == 0:
                reverse = True
```

```
                gear = -1  # Reverse gear
                clutch_pressed = False
                print("Shifted to reverse gear.")
            else:
                print("Cannot shift to reverse while moving.")

        # Accelerate
        elif command == 'a':
            if ignition_on and handbrake_released and (gear > 0 or reverse):
                if reverse:
                    if speed > -20:
                        speed -= 5
                        print(f"Accelerating in reverse... Current speed:
{abs(speed)} km/h.")
                    else:
                        print("Cannot accelerate further in reverse. Max
speed is 20 km/h.")
                else:
                    max_speed = gear_speed_limits.get(gear, 0)
                    if speed < max_speed:
                        speed += 10
                        print(f"Accelerating... Current speed of car is
{speed} km/h.")
                        if gear == 5 and speed == gear_speed_limits[5]:
                            print("Max speed achieved! You are at the top
speed of the car.")
                    else:
                        print(f"Cannot accelerate further. Max speed for gear
{gear} is {max_speed} km/h.")
                        print("Shift to a higher gear to increase speed.")
            else:
                print("Cannot accelerate. Ensure ignition is on, handbrake is
released, and correct gear is engaged.")

        # Brake
        elif command == 'b':
            if speed > 0:
                speed -= 10
                print(f"Braking... Current speed: {speed} km/h.")
                for g, limit in gear_speed_limits.items():
                    if speed <= limit:
                        if gear > g:
                            print(f"Speed has reduced to {speed} km/h. Shift
down to gear {g}.")
                        break
                if speed == 0:
                    print("Car has stopped. Shift to the correct gear before
accelerating.")
            elif speed < 0:
                speed += 10
                print(f"Braking in reverse... Current speed: {abs(speed)}
km/h in reverse.")
            else:
                print("The car is already stopped.")

        # Quit
```

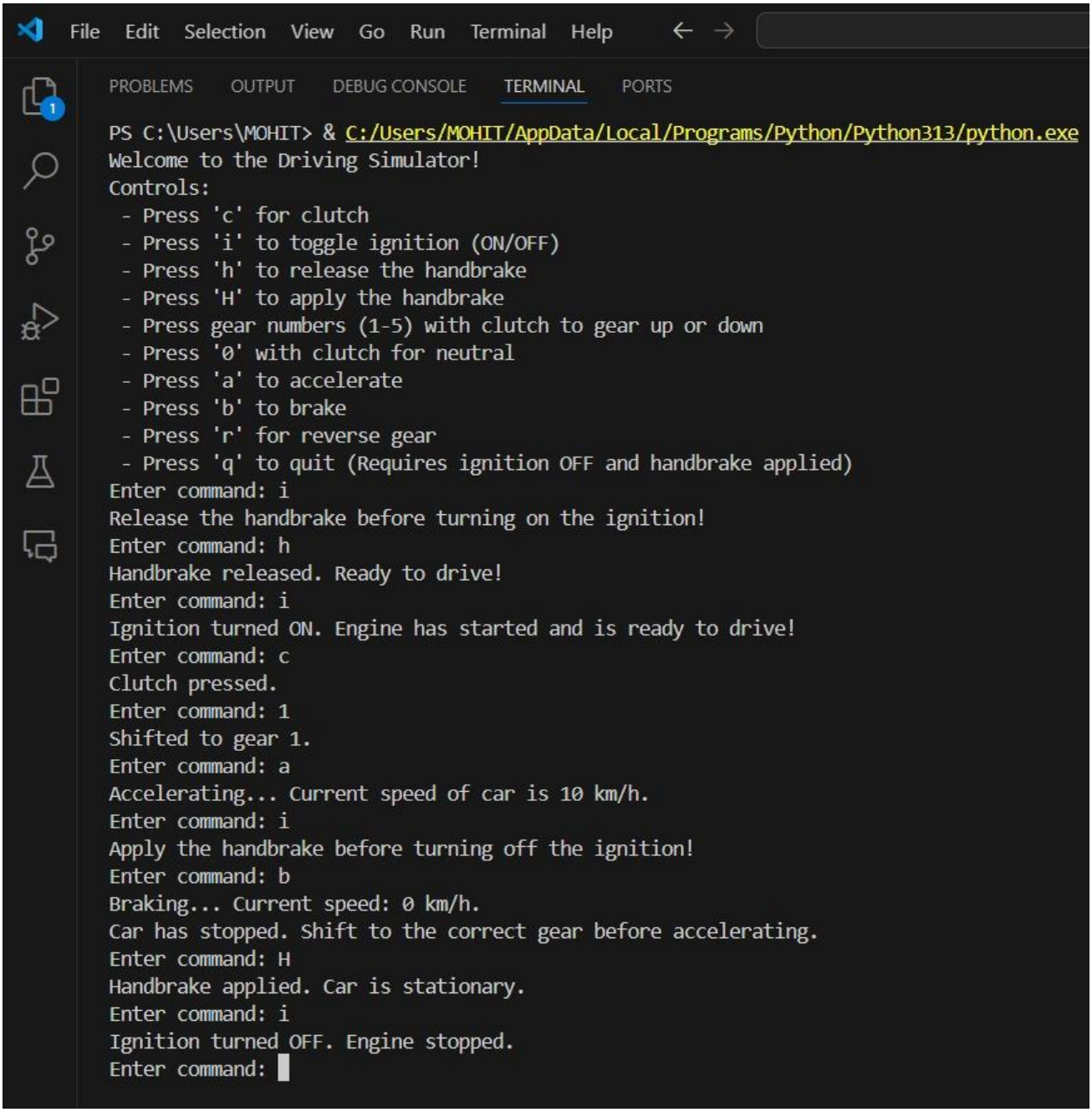
```
        elif command == 'q':
            if ignition_on:
                print("Turn off the ignition before quitting.")
            elif handbrake_released:
                print("Apply the handbrake before quitting.")
            else:
                print("Exiting simulator. Drive safe!")
                break

        # Invalid Command
        else:
            print("Invalid command or action. Check the instructions.")

# Start the simulator
driving_simulator()
```

7. Testing

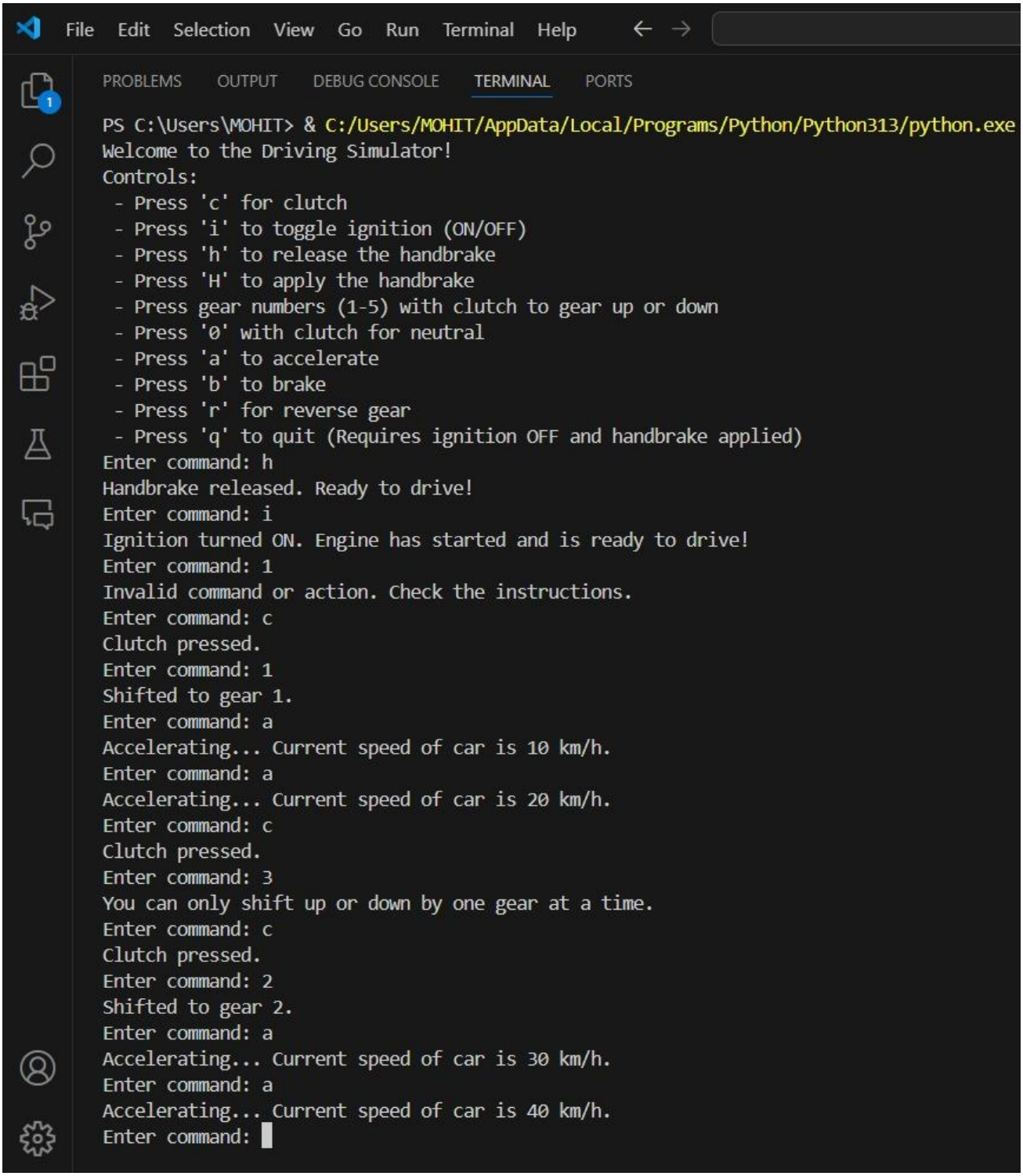
Case 1: Handbrake



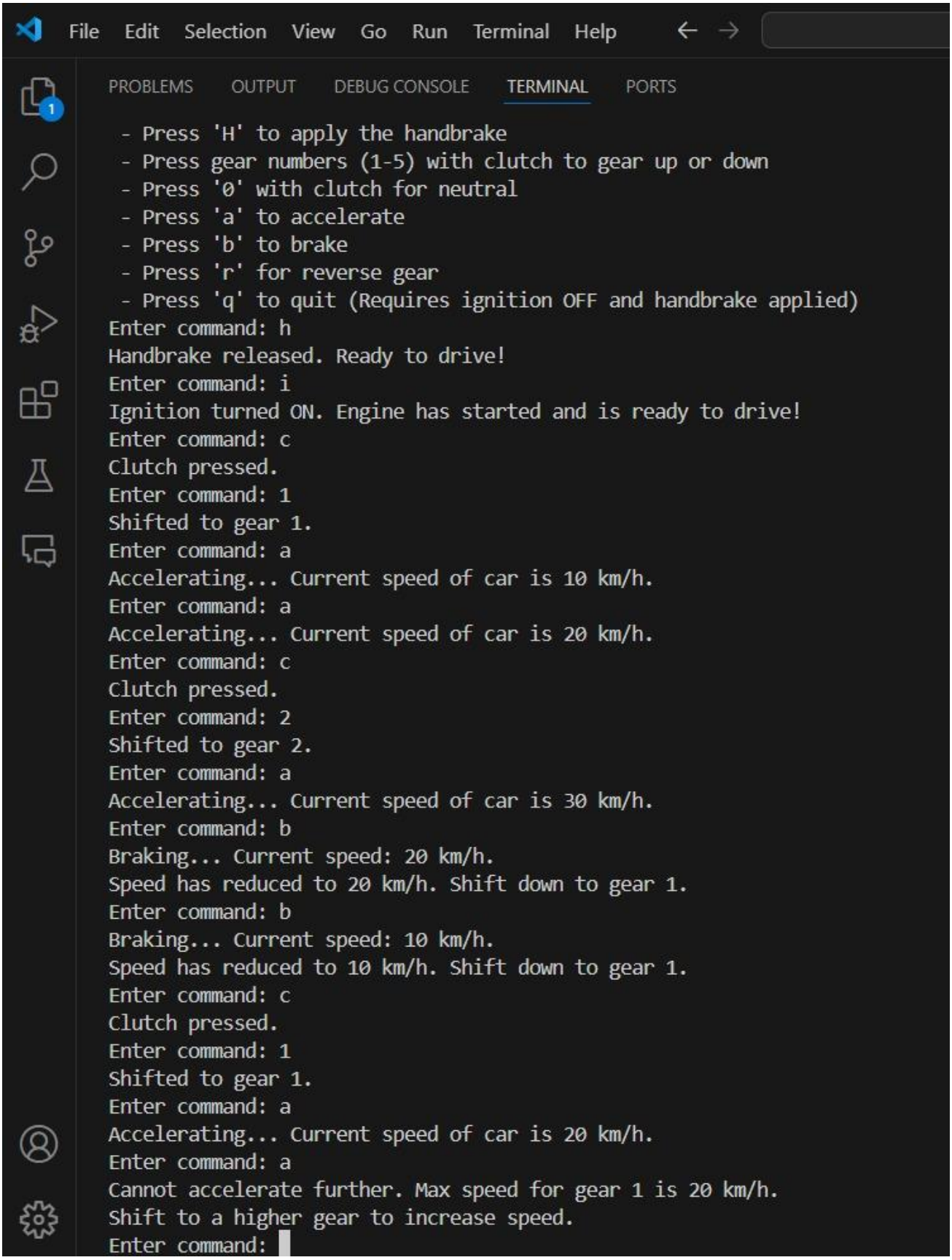
Case 2: Ignition



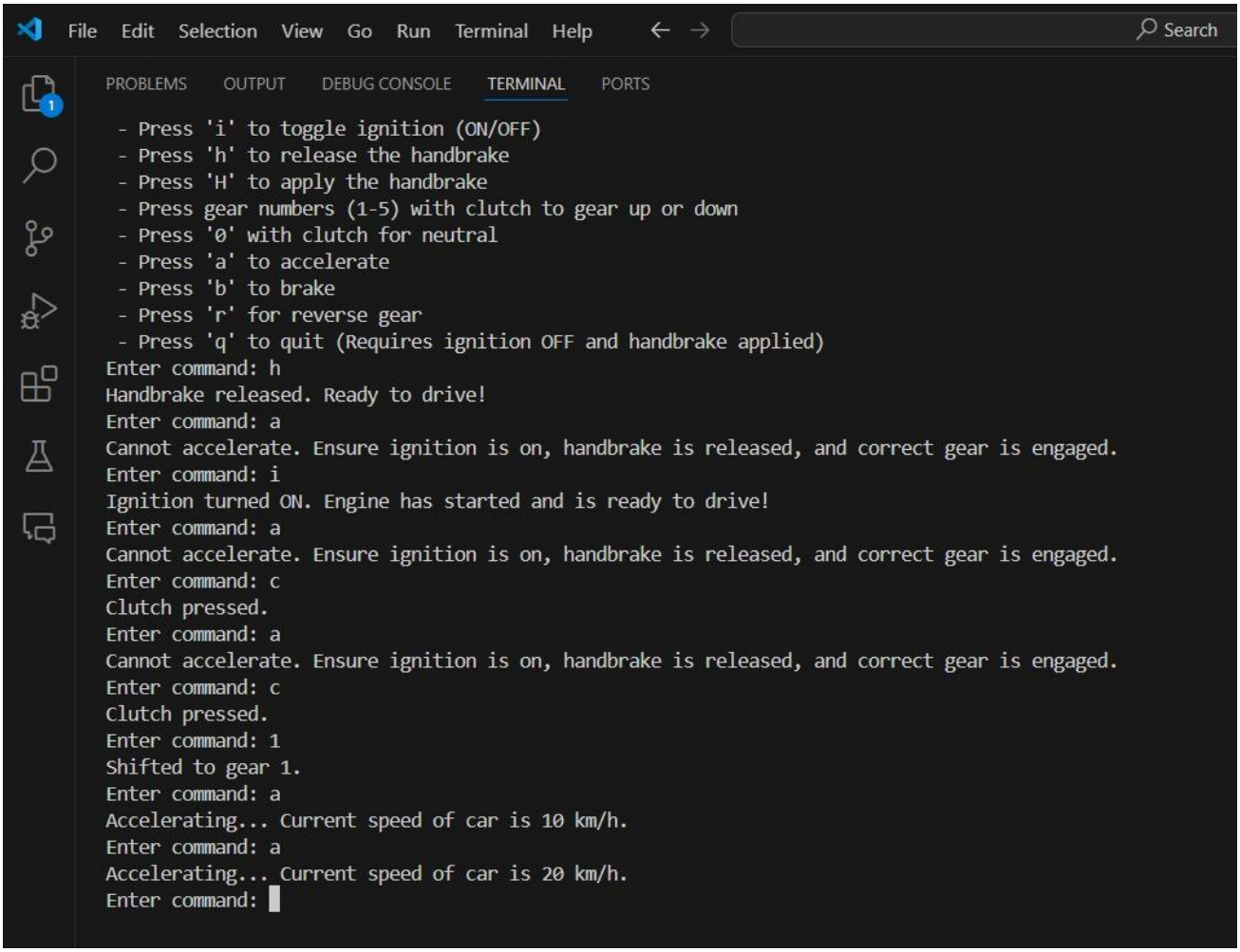
Case 3: Gear shifting without clutch and skipping gears in between



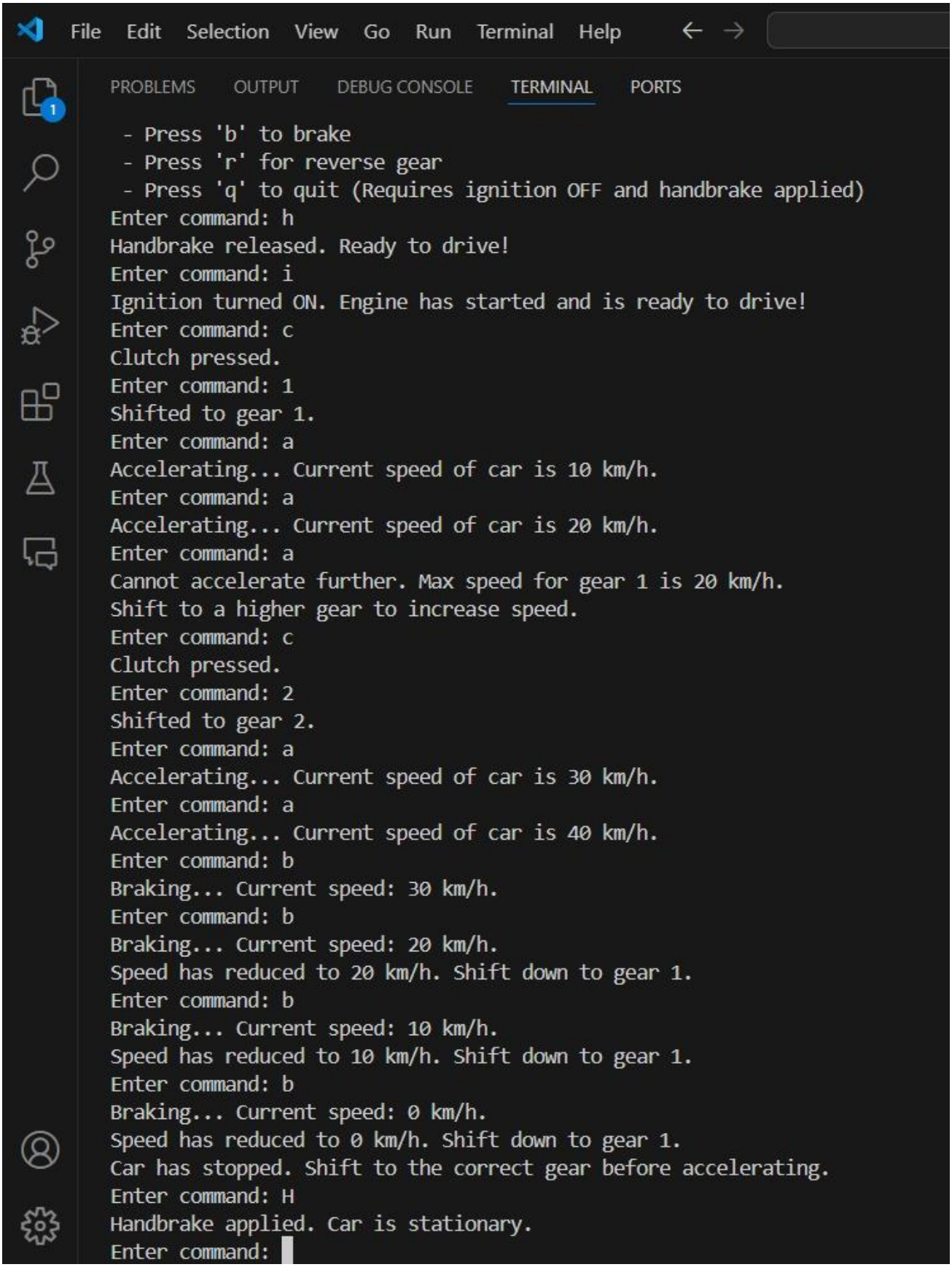
Case 4: Gear down shifting



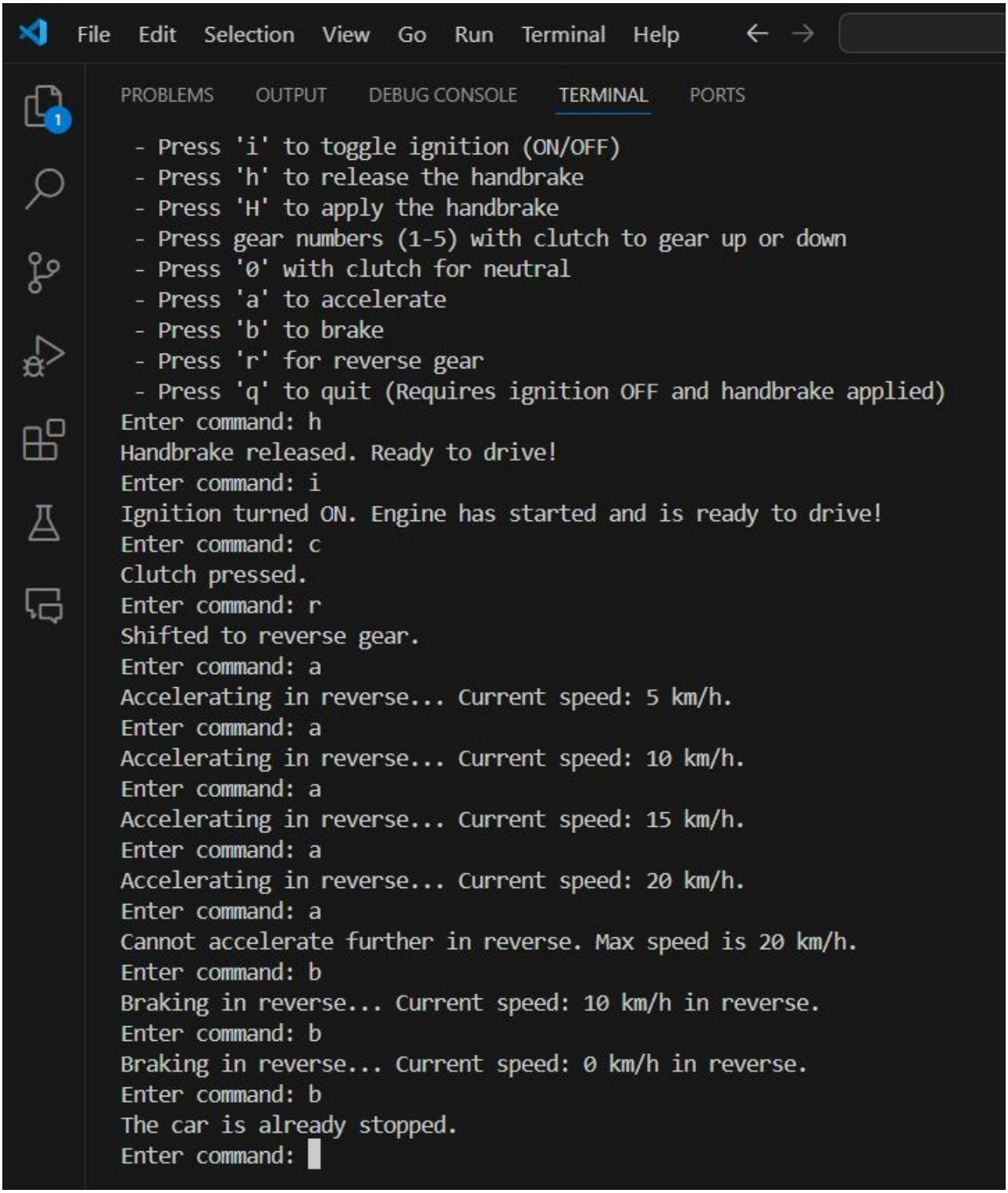
Case 5: Accelerating without gear applied



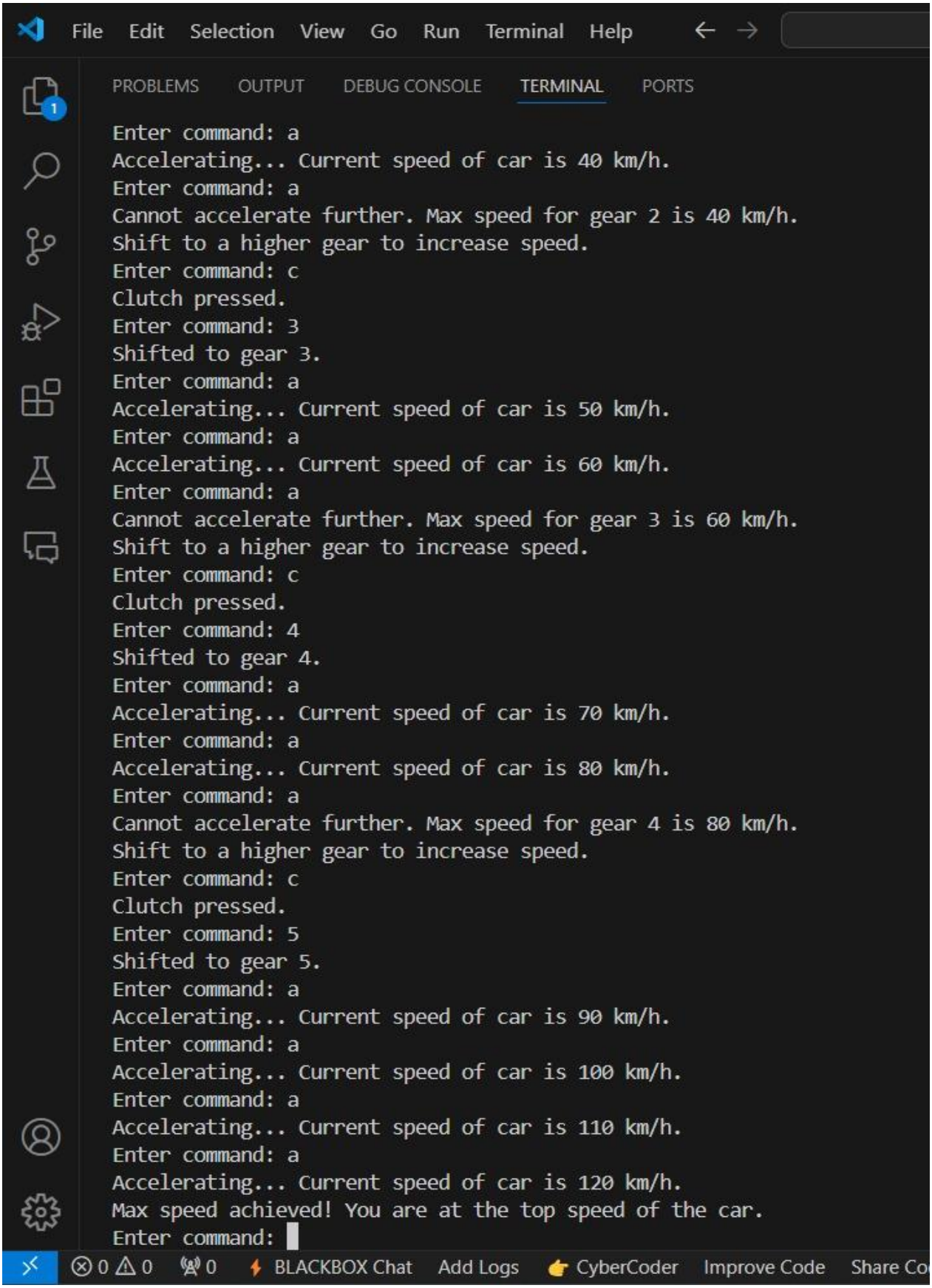
Case 6: Braking



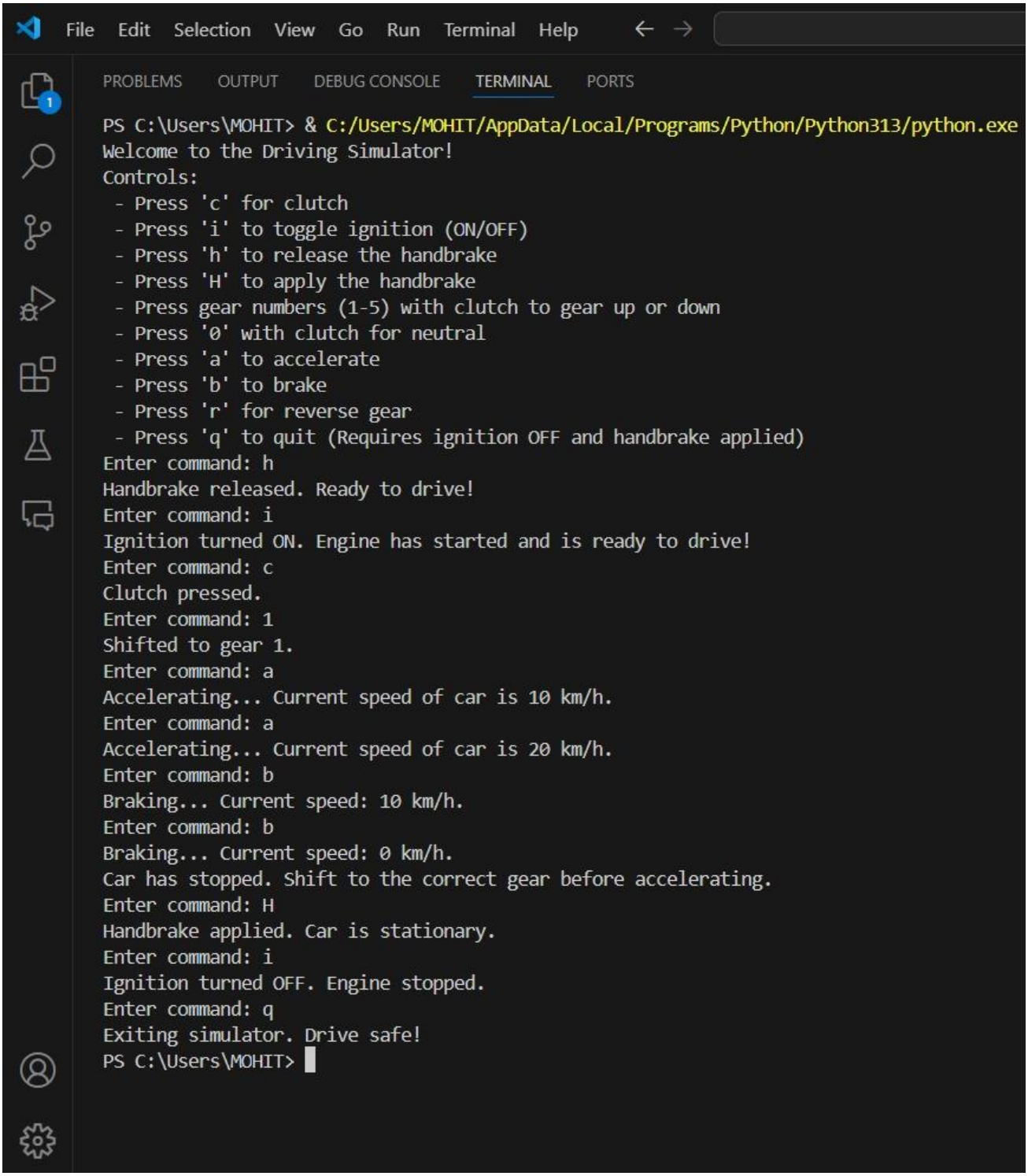
Case 7: Reverse



Case 8: Max speed



Case 9: Ideal driving



8. Conclusion

The Driving Simulator successfully replicates fundamental car-driving operations in a virtual environment. With its realistic rules and safety mechanisms, it provides users with an educational and interactive experience. Further enhancements could include graphical visualization and advanced driving scenarios for an even more immersive simulation.

9. References

1. Python Documentation
   - Python Software Foundation. *The Python Language Reference.*
     Available at: https://docs.python.org/3/
     Accessed for understanding the syntax and structure of Python commands used in this project.
2. PEP 8 – Python Style Guide
   - Python Software Foundation. *PEP 8: Style Guide for Python Code.*
     Available at: https://peps.python.org/pep-0008/
     Accessed to follow coding standards and improve code readability.
3. Microsoft Word Official Documentation
   - Microsoft Corporation. *Insert code or formatted text into Word documents.*
     Available at: https://support.microsoft.com
     Used to explain methods for embedding Python code into Word.
4. Code Syntax Highlighting Tools
   - Highlight.js. *Syntax Highlighting Library.*
     Available at: https://highlightjs.org/
     Utilized as a tool for syntax highlighting during presentation formatting.
5. Official Python Libraries
   - Python Software Foundation. *Python's Built-in Libraries Documentation.*
     Available at: https://docs.python.org/3/library/

Referenced for functions and methods utilized in the simulator.

6. TutorialsPoint Python Resources
   - TutorialsPoint. *Python Programming Basics.* Available at: https://www.tutorialspoint.com/python/index.htm Accessed for brushing up on Python concepts during development.
7. Driving Simulators Research
   - Sharma, S. & Gupta, P. *Driving Simulators in Learning: Applications and Case Studies.* Available at: https://example.com/simulator-research Referenced for understanding the real-world application of driving simulators.