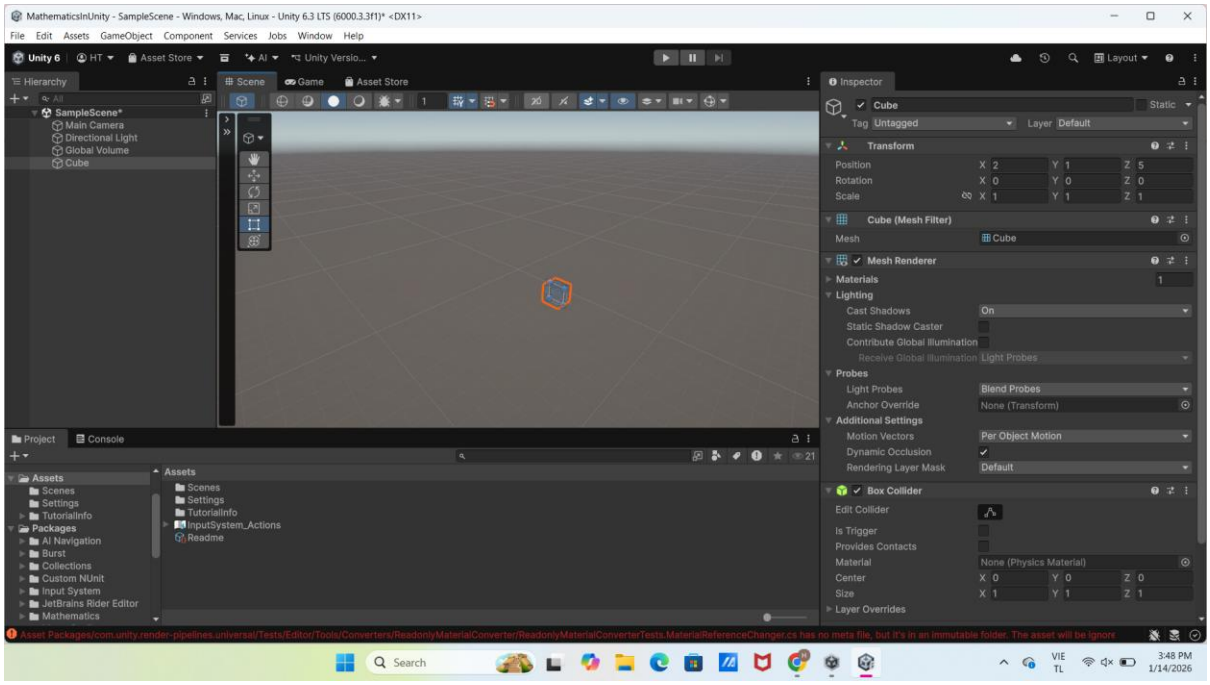
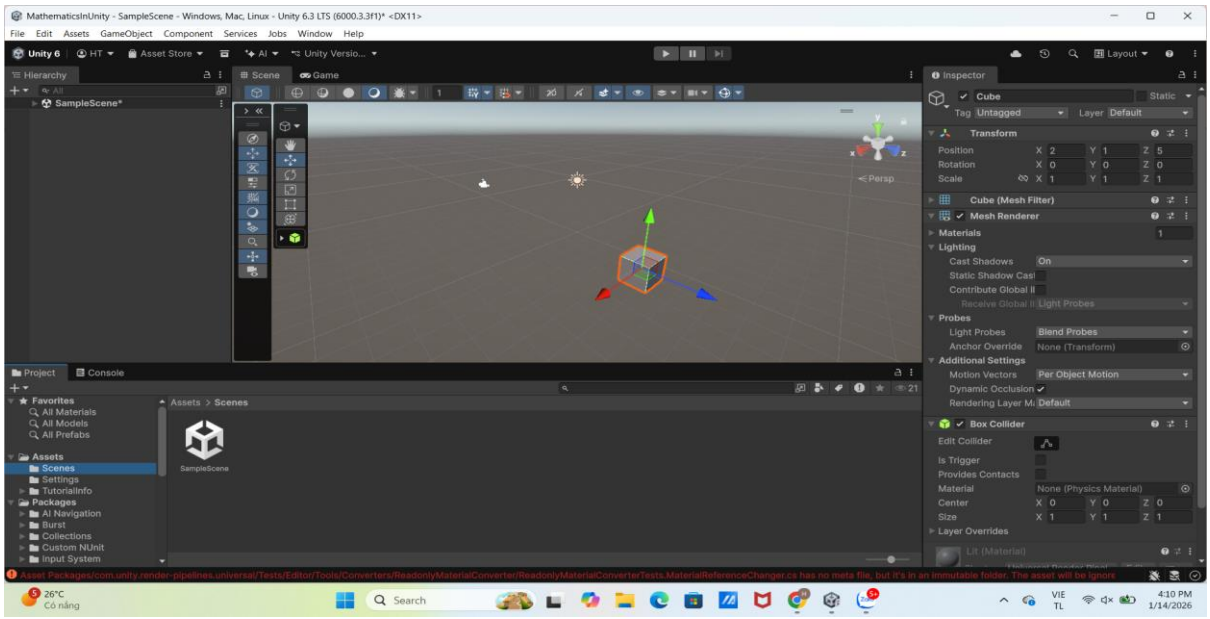
PHẦN A – COORDINATE SYSTEM & WORLD SPACE (20%)

A1. Tạo một Cube tại vị trí:
X = 2, Y = 1, Z = 5



A2. Bật Gizmos trong Scene View và chụp ảnh thể hiện:
- Trục X
- Trục Y
- Trục Z

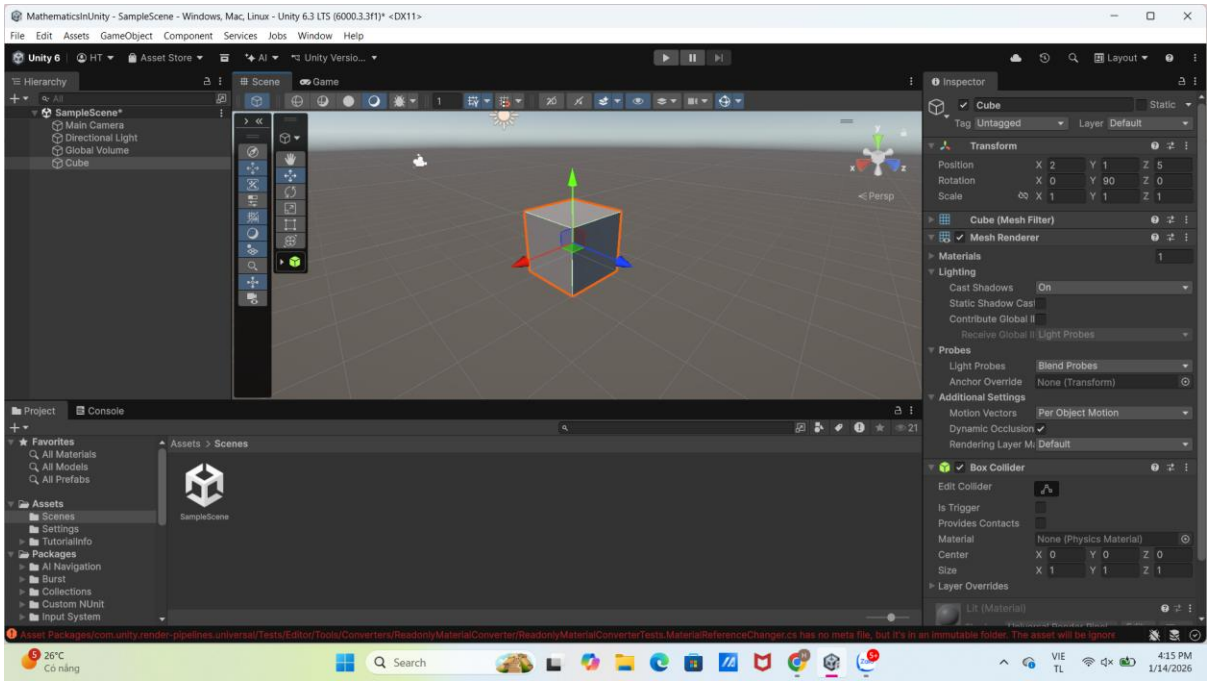


A3. Trả lời các câu hỏi:
- Trục nào hướng lên trên trong Unity? Trục Y hướng lên trên

- Trục nào hướng về phía Camera? Trục Z dương (+Z) hướng về phía Camera

PHẦN B – LEFT-HANDED COORDINATE SYSTEM (15%)

B1. Xoay Cube với Rotation:
Y = 90

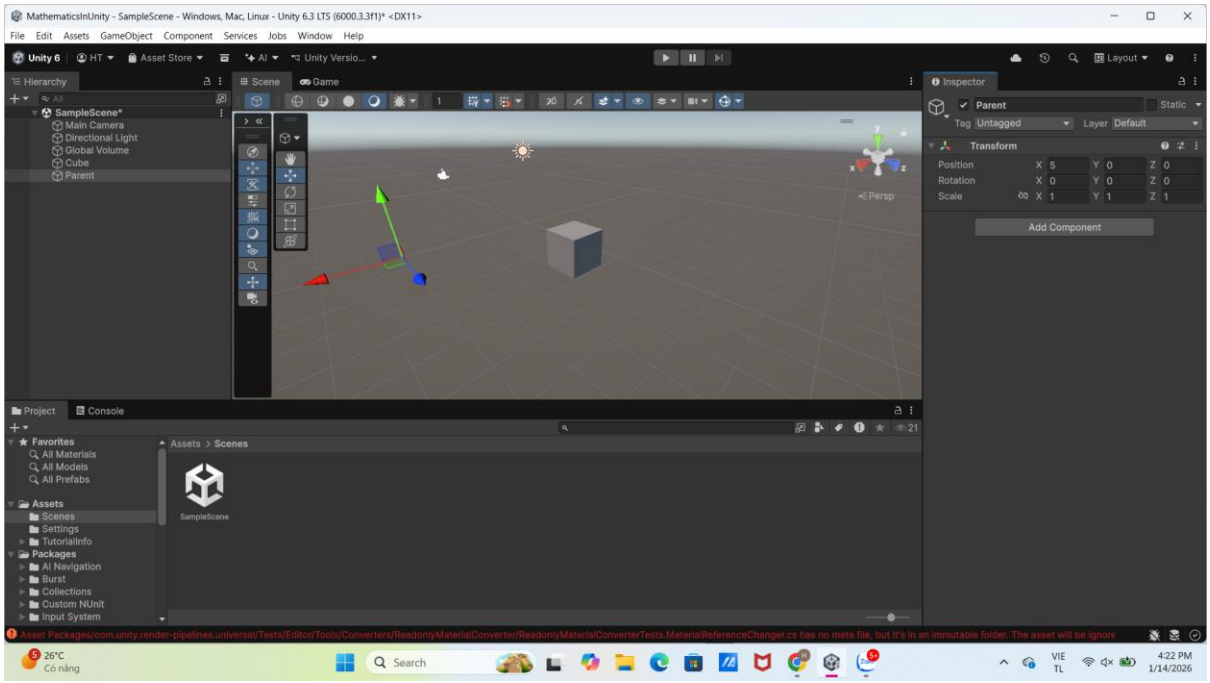


B2. Quan sát hướng quay của Cube và trả lời:
- Cube quay theo chiều nào? Cube quay sang bên phải (theo chiều kim đồng hồ khi nhìn từ trên xuống)

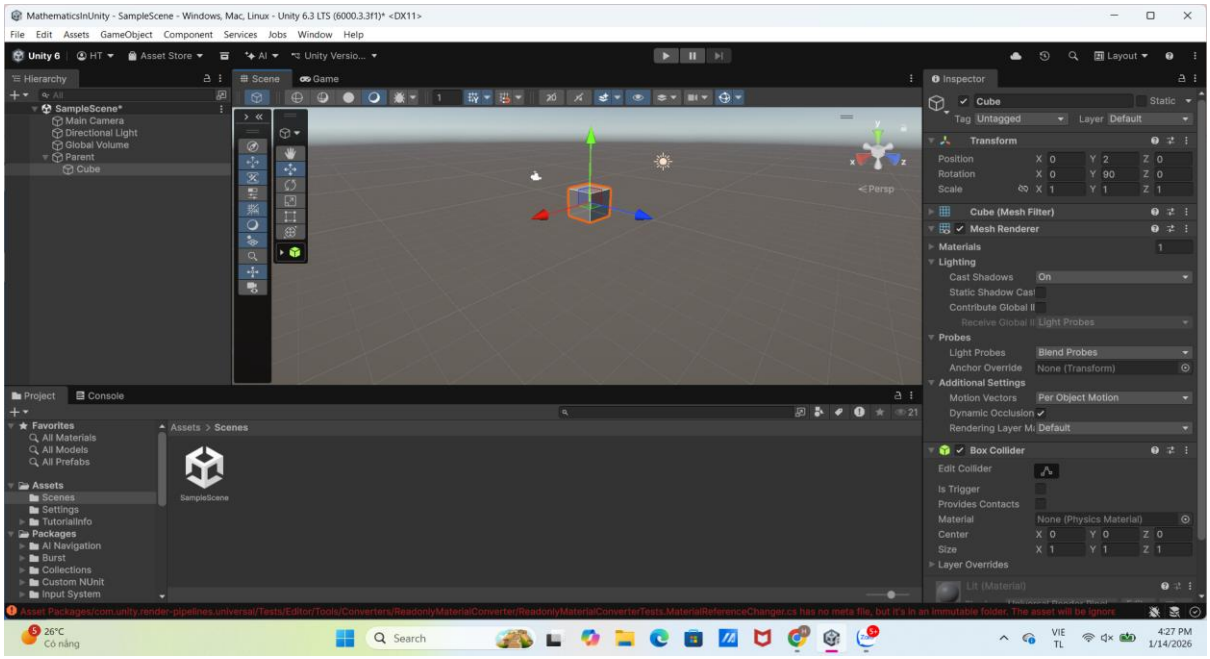
- Điều này thể hiện Unity sử dụng Left-Handed Coordinate System như thế nào? Trong hệ tọa độ Left-Handed, khi xoay dương quanh trục Y, object quay theo chiều kim đồng hồ → đúng với Unity

PHẦN C – LOCAL SPACE VÀ WORLD SPACE (25%)

C1. Tạo một Empty GameObject tên là “Parent” tại vị trí:
(5, 0, 0)



C2. Đặt Cube làm con của Parent và thiết lập:
Local Position của Cube = (0, 2, 0)

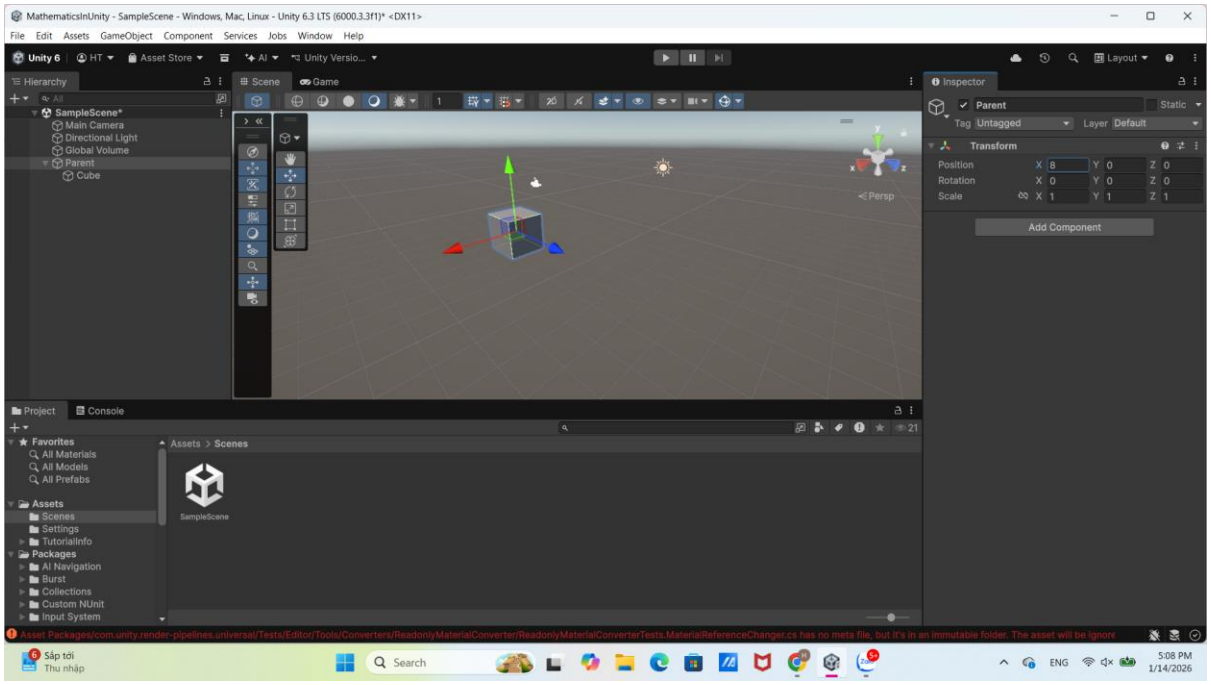


C3. Ghi lại:
- Local Position của Cube
- World Position của Cube

Cube có Local Position là (0,2,0) so với Parent.
Vì Parent có World Position (5,0,0), nên World Position của Cube là (5,2,0).

C4. Di chuyển Parent sang vị trí:
(8, 0, 0)

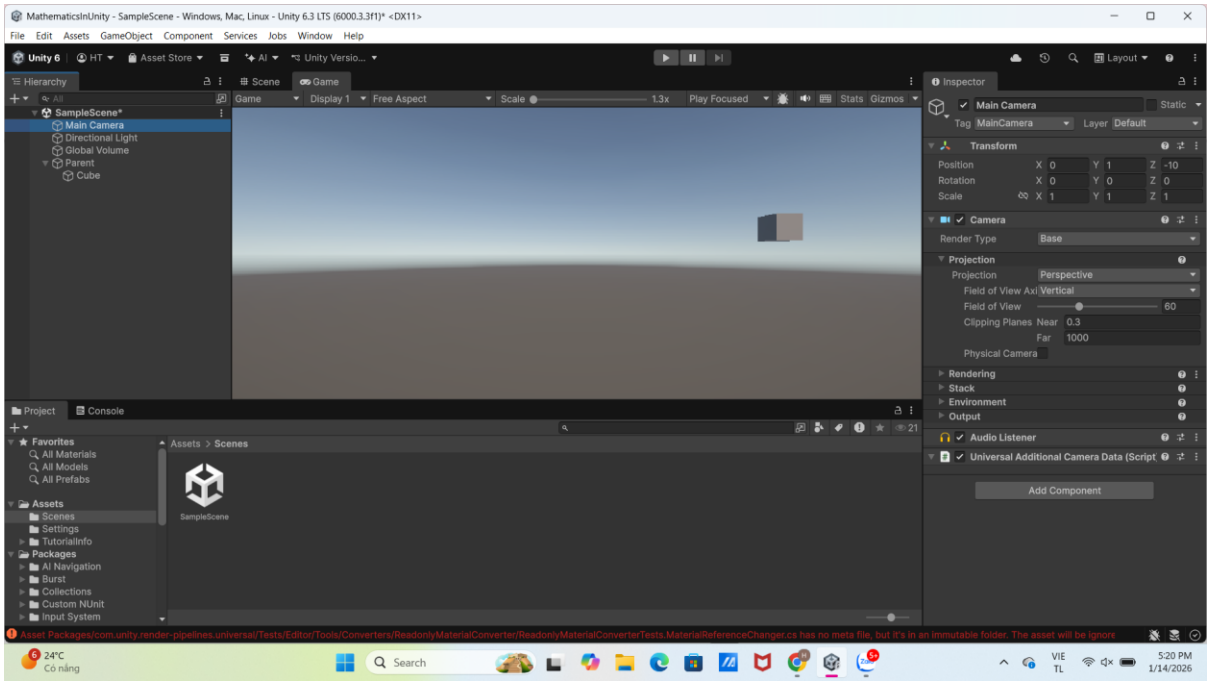
Trả lời:
- Local Position của Cube có thay đổi không? Không đổi (0, 2, 0)
- World Position của Cube thay đổi như thế nào? Thành (8, 2, 0)



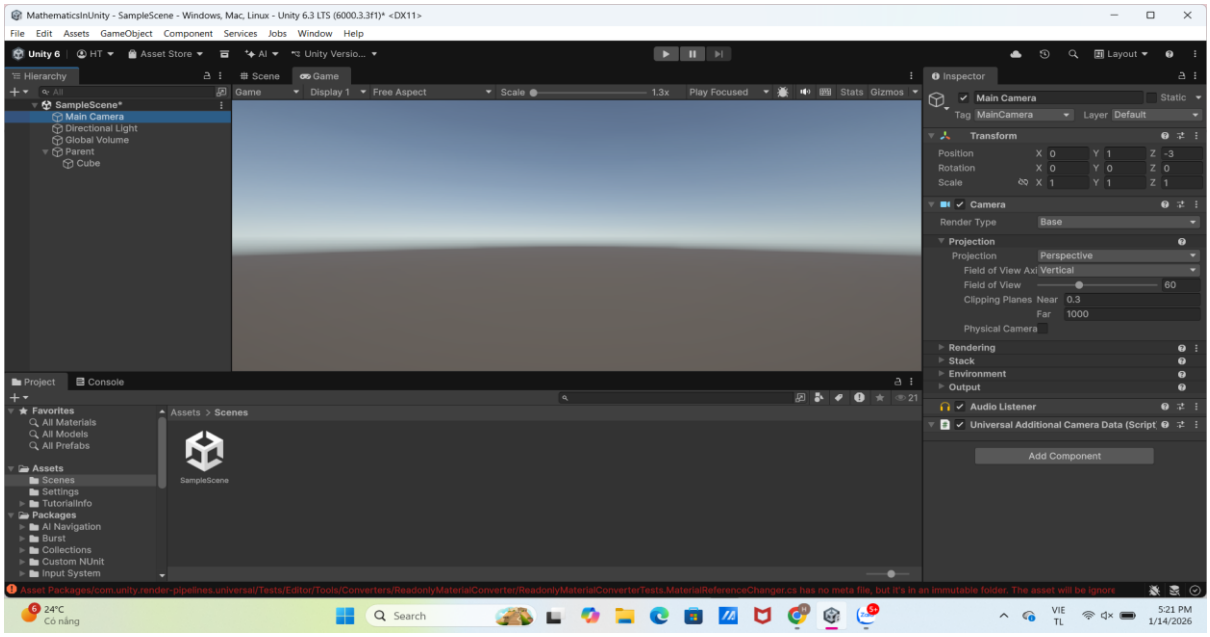
PHẦN D – GRAPHICS PIPELINE (20%)

D1. Di chuyển Camera dọc trục Z từ -10 đến -3

Z= -10

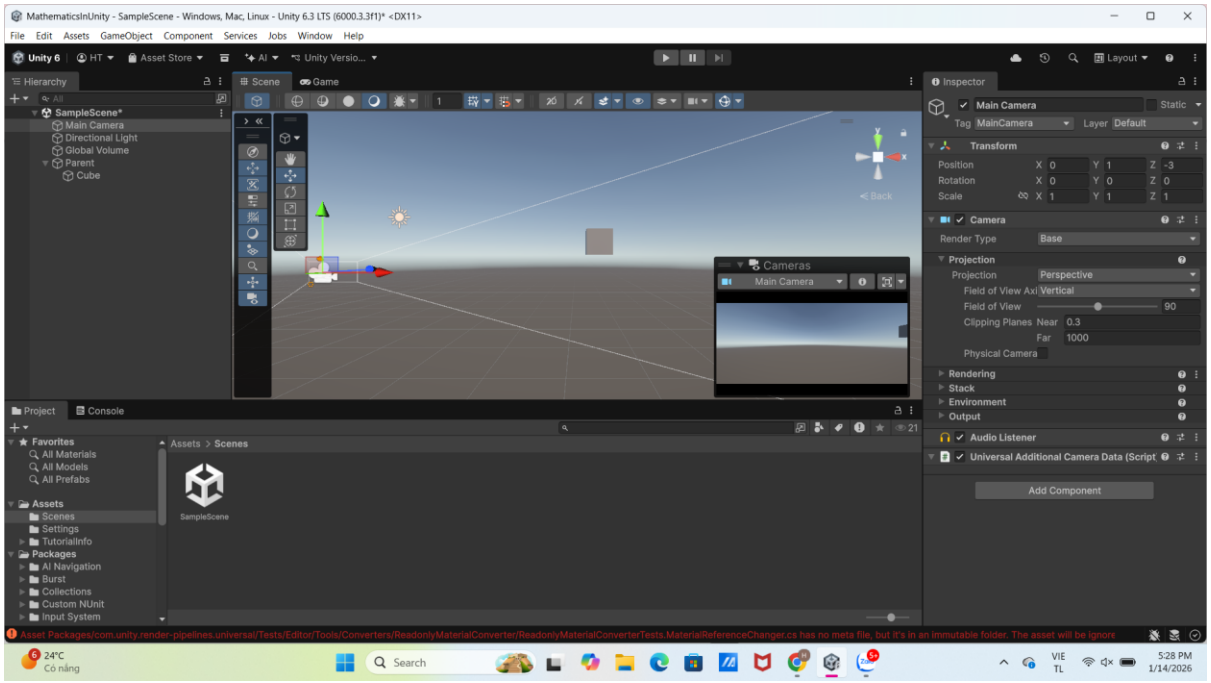


Z= -3

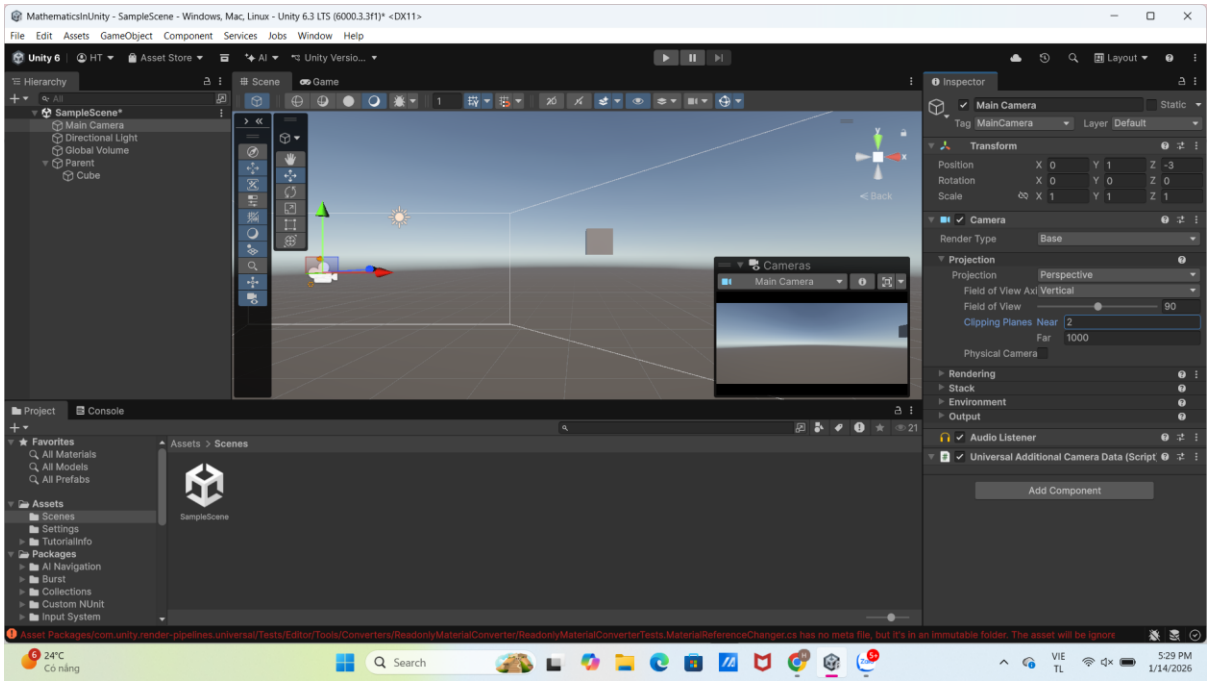


D2. Thay đổi các thông số của Camera:

- Field of View



- Near Clip Plane



Trả lời:

- Vì sao object trông to hoặc nhỏ hơn dù không đổi vị trí? Do Field of View thay đổi góc nhìn của Camera

- Vì sao object có thể biến mất khỏi màn hình? Vì object nằm ngoài Near/Far Clip Plane

PHẦN E – SCREEN SPACE (20%)

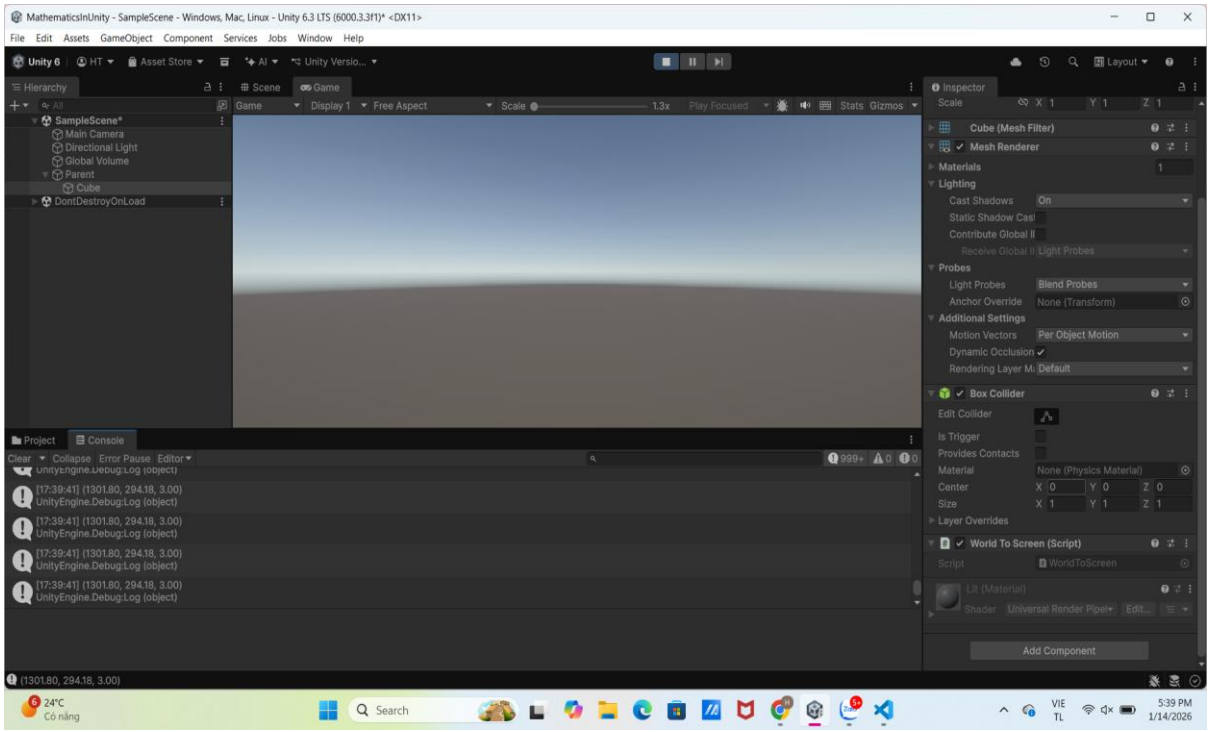
E1. Tạo script WorldToScreen.cs với nội dung:

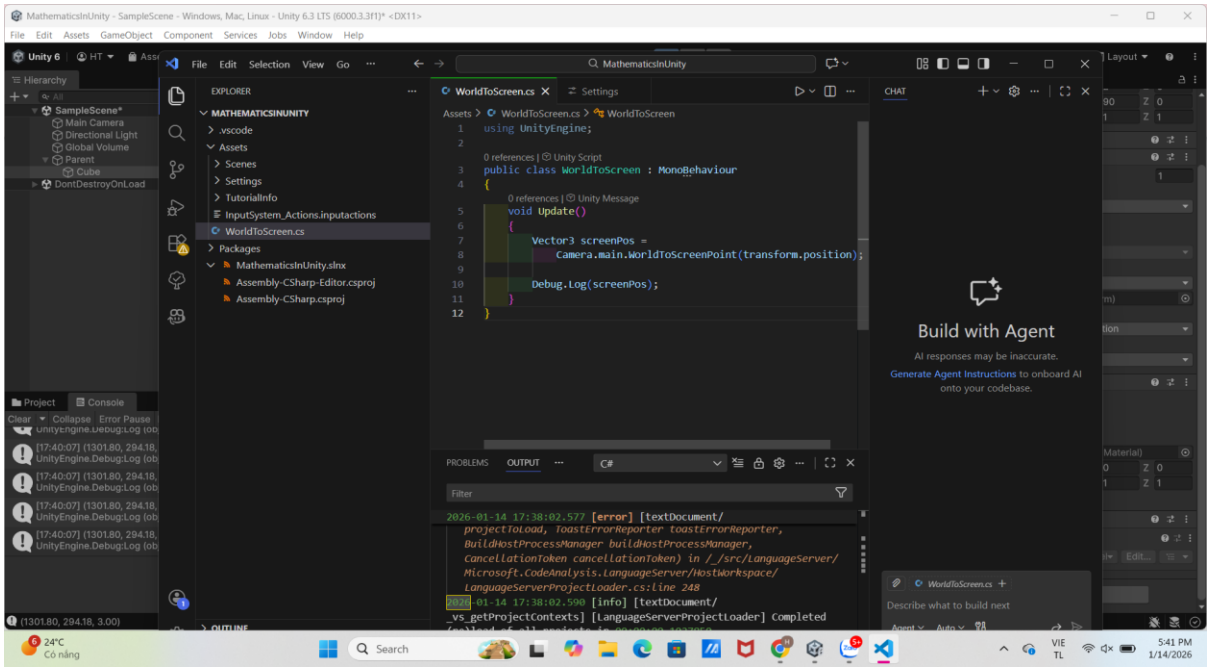
```
using UnityEngine;

public class WorldToScreen : MonoBehaviour
{
    void Update()
    {
        Vector3 screenPos =
            Camera.main.WorldToScreenPoint(transform.position);

        Debug.Log(screenPos);
    }
}
```

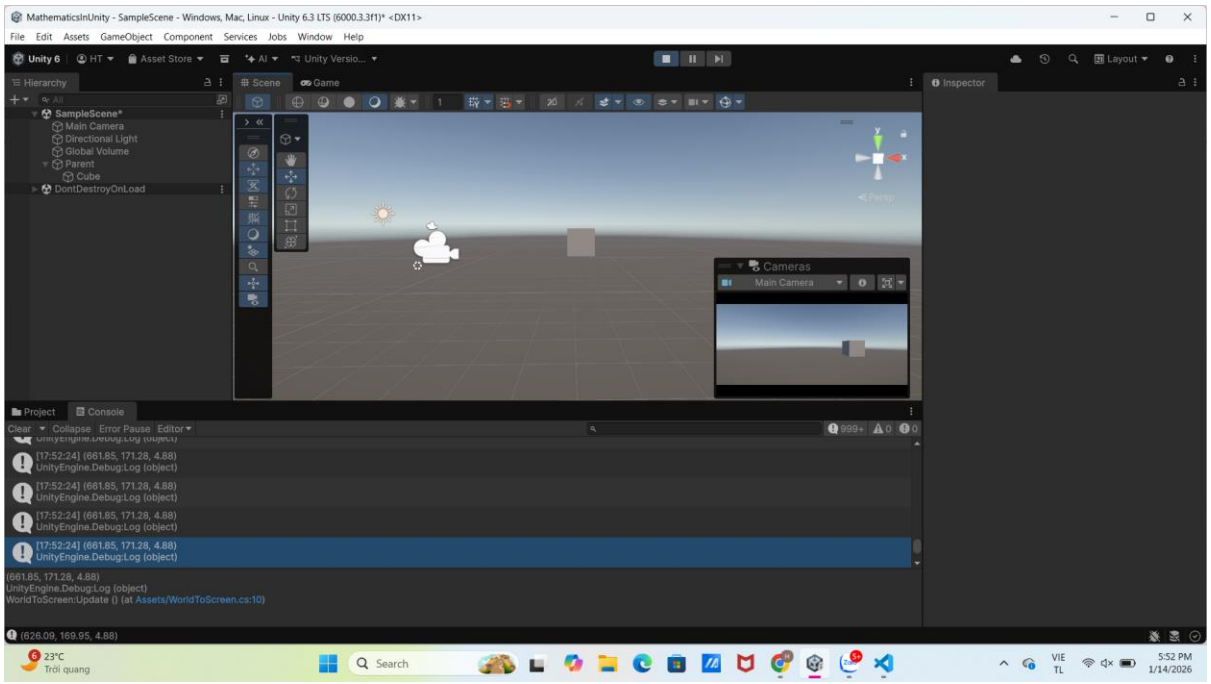
E2. Gắn script vào Cube và chạy game

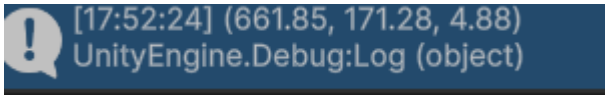




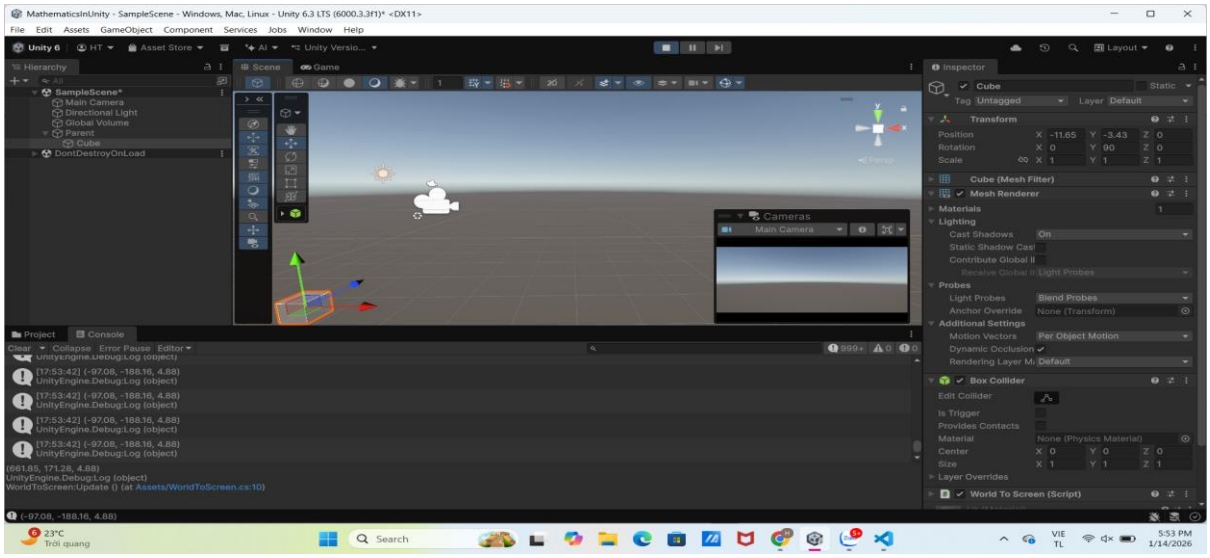
E3. Ghi lại:

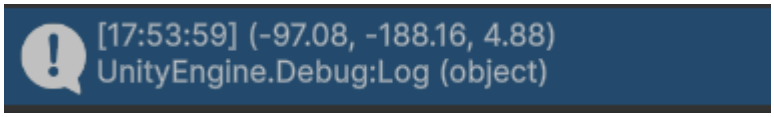
- Screen Position khi Cube ở giữa màn hình





- Screen Position khi Cube ở góc dưới bên trái





E4. Trả lời:

- Gốc tọa độ của Screen Space nằm ở đâu? Góc dưới bên trái màn hình (0,0)

- Screen Space khác World Space như thế nào? World Space dùng tọa độ 3D, Screen Space dùng pixel 2D

Nhận xét cá nhân

Qua bài tập *Mathematics in Unity*, em hiểu rõ hơn cách Unity sử dụng toán học để xây dựng và hiển thị không gian 3D. Trước đây em chỉ thao tác các đối tượng một cách trực quan mà chưa thực sự chú ý đến ý nghĩa của hệ tọa độ. Sau khi làm bài, em đã nắm được vai trò của các trục X, Y, Z, hiểu rằng trục Y hướng lên trên và trục Z hướng về phía Camera, đồng thời nhận ra Unity sử dụng hệ tọa độ Left-Handed thông qua cách xoay object. Bên cạnh đó, em phân biệt rõ hơn giữa Local Space và World Space khi làm việc với object cha – con, từ đó hiểu vì sao vị trí của object con thay đổi theo object cha. Phần làm việc với Camera giúp em thấy rõ việc thay đổi Field of View hay Clip Plane có thể khiến object trông to, nhỏ hoặc thậm chí biến mất dù không đổi vị trí. Cuối cùng, việc chuyển đổi từ World Space sang Screen Space bằng script giúp em hiểu rõ hơn cách Unity hiển thị không gian 3D lên màn hình 2D. Nhìn chung, bài tập này giúp em có cái nhìn thực tế và nền tảng hơn về toán học trong Unity, hỗ trợ cho việc học và phát triển game sau này.