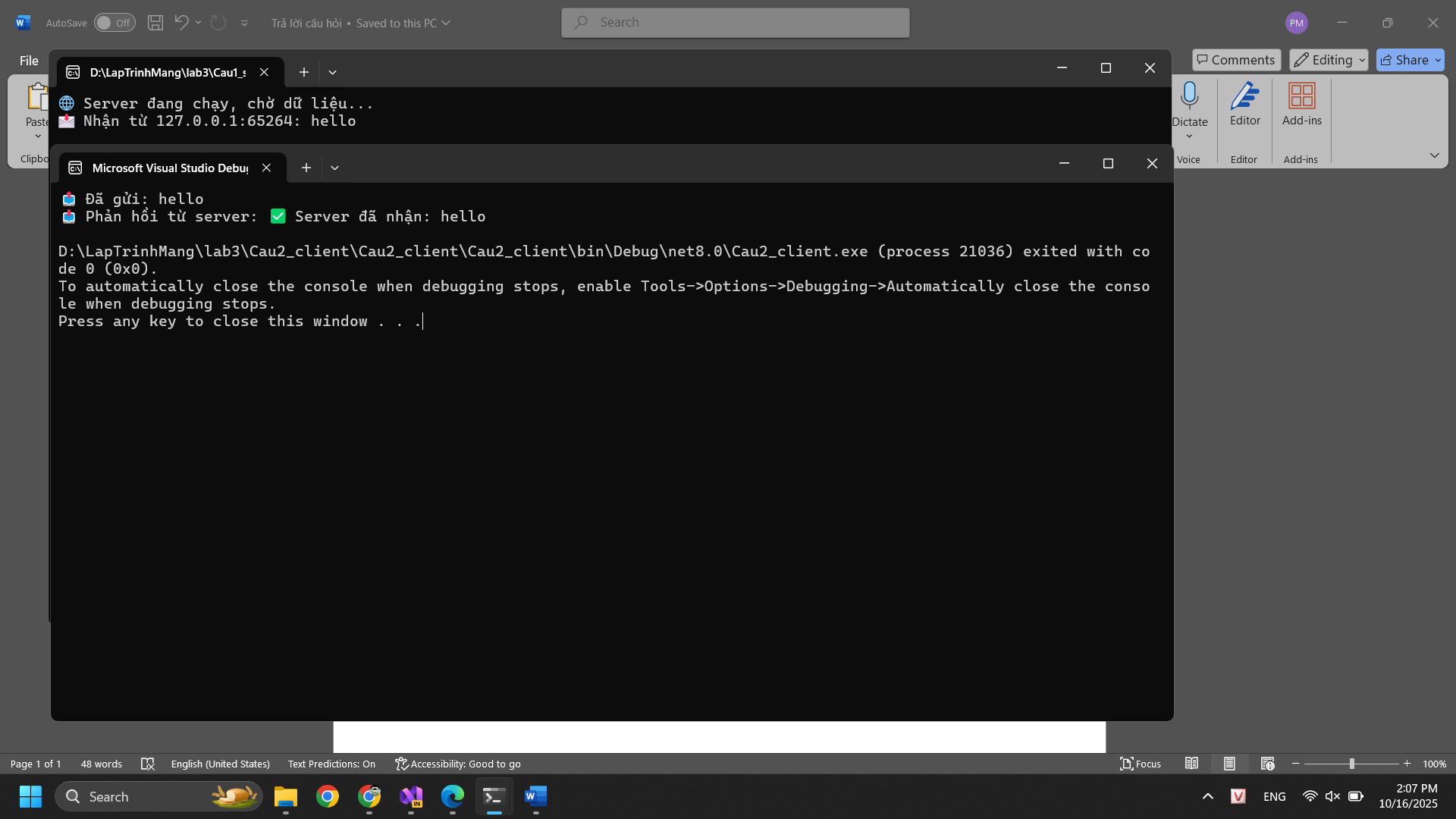
**Câu 1:**  


**Câu 1.4:**  
**1) Như hình trên port 2747 ở đâu ra**

-Hệ điều hành chọn một port ngẫu nhiên (vd: 2747) trong dải port tạm thời (thường từ 49152 – 65535 trên Windows). Vì vậy, 2747 là port tạm thời được hệ điều hành tự cấp cho client.

**2) Có phải lúc nào client cũng mở port 2747 để kết nối với Server không ?**

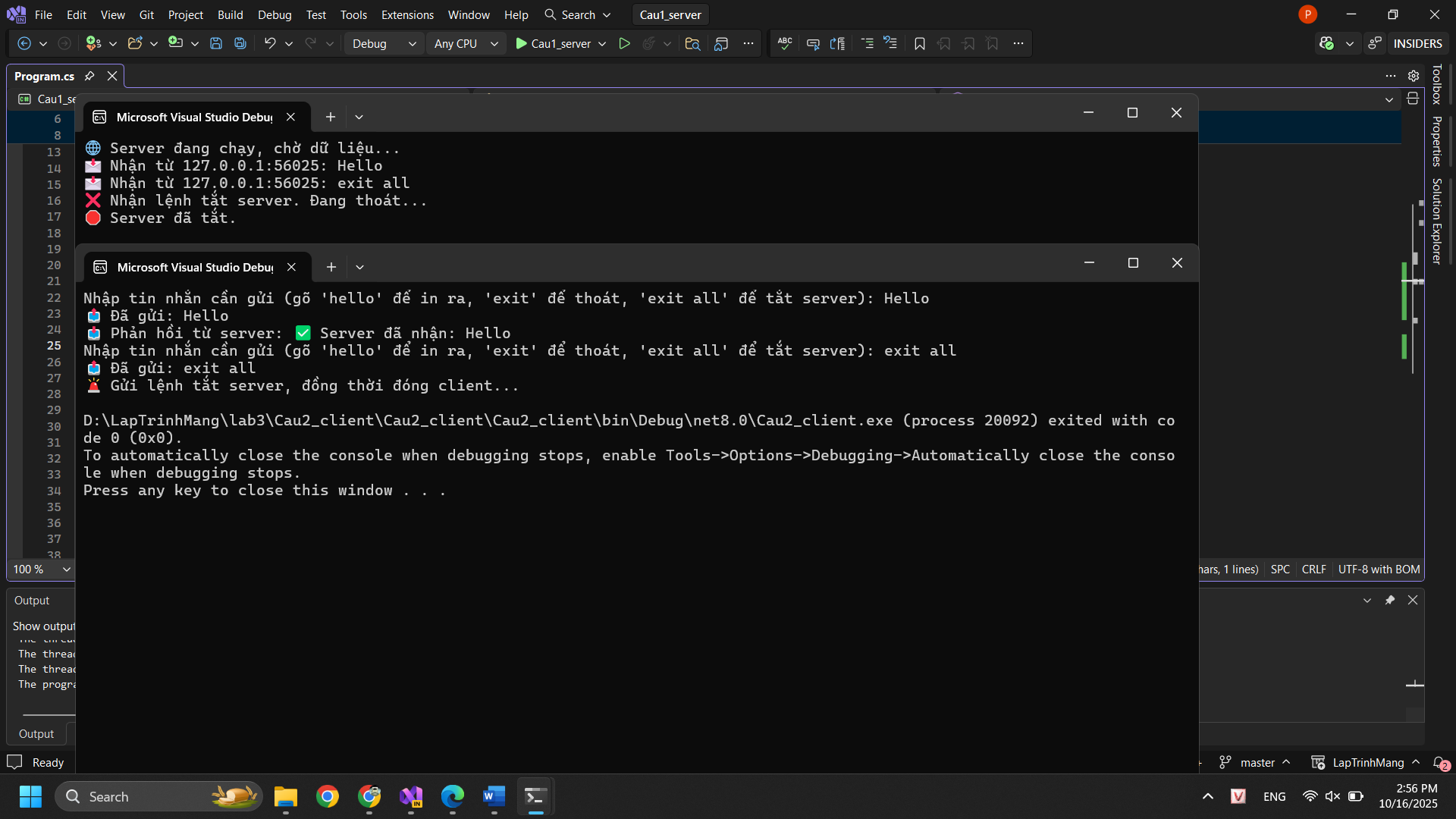
-Không. Mỗi lần chạy chương trình client, hệ điều hành có thể gán một port khác nhau. Miễn là port đó chưa bị chiếm, Windows/Linux sẽ chọn ngẫu nhiên trong dải port tạm.

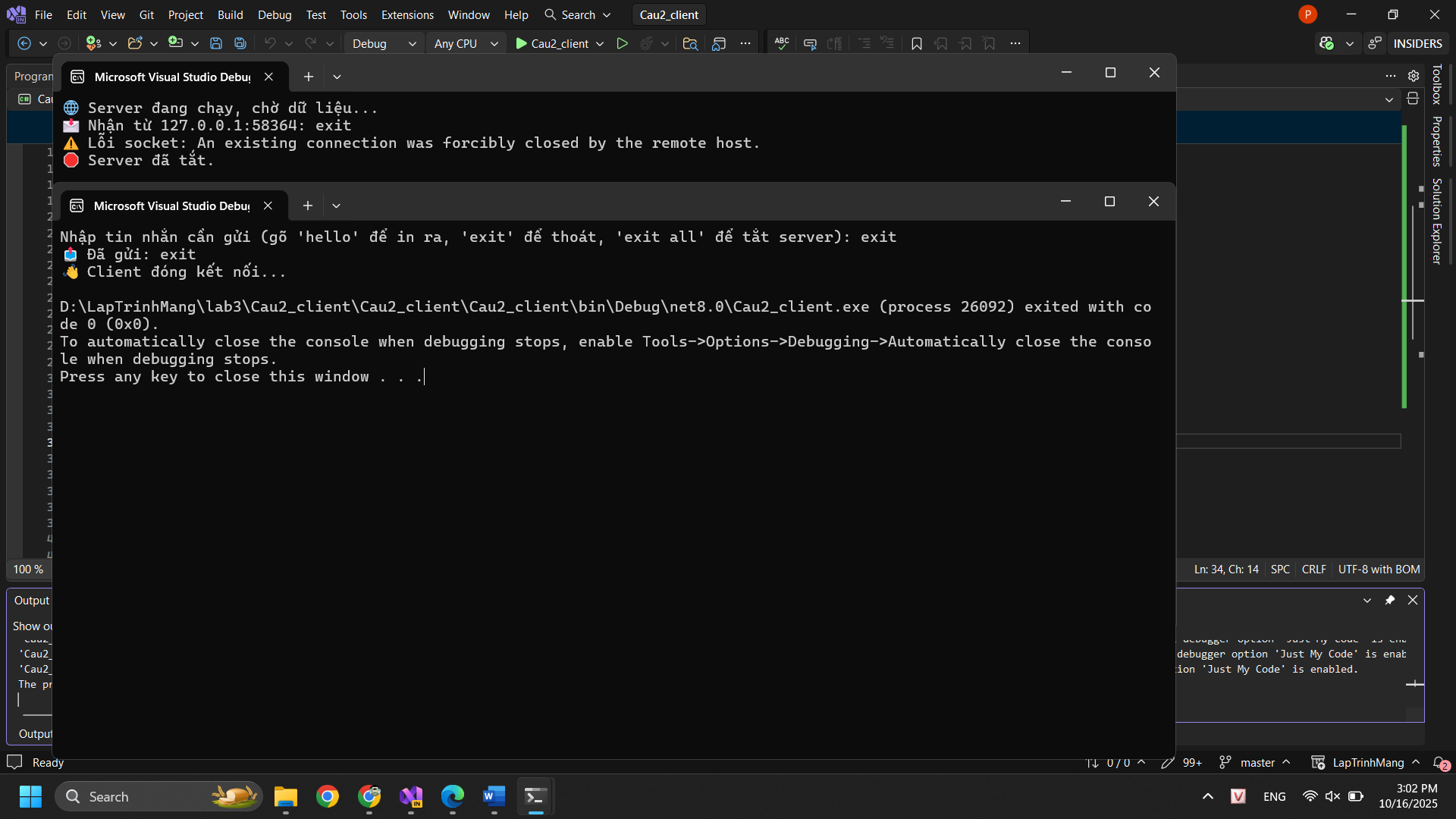
**3) Tại sao khi lập trình mạng dùng giao thức UDP thì client phải gởi câu chào lên server trước?**

- Vì UDP không có kết nối sẵn, server chỉ biết client sau khi nhận được gói đầu tiên.

Câu 2:

-Cải tiến chương trình để ở client gõ “exit” thì đóng client, khi client gõ “exit all” thì đóng cả client và server.





**Câu 3: Hướng dẫn lập trình UDP Server**

- Dùng vòng lặp vô hạn lấy kết quả client gởi lên, chuyển nó thành chuỗi và hiển thị lên màn hình đồng thời gởi lại dữ liệu nhận được về lại cho client

A screenshot of a computer program

AI-generated content may be incorrect.

**Câu 3.4:**

**1) Khi chạy chương trình với đoạn code gởi nhận dữ liệu như trên, lúc chưa nhập dữ liệu cho client để gởi lên thì không xảy ra lỗi nhưng khi nhập dữ liệu để gởi lên server sẽ xảy ra lỗi, vì sao lại xảy ra lỗi này ?**

- Lỗi này xảy ra vì client gửi dữ liệu đi nhưng không nhận được phản hồi hợp lệ từ server (do server chưa sẵn sàng hoặc đã tắt), mà UDP không đảm bảo kết nối, nên khi gọi Receive() sẽ lỗi.

**2) Khi server chưa bật thì chương trình trên có bị lỗi không? Tạo sao?**

- UDP client vẫn gửi được khi server chưa bật, nhưng sẽ không nhận được phản hồi (vì không ai trả lời) → có thể treo ở lệnh Receive() chờ vô hạn.

**3) Khi đang chạy chương trình tắt client thì chương trình trên có bị lỗi không? Tại sao**

- Không bị lỗi. UDP server không duy trì trạng thái kết nối, nó chỉ lắng nghe gói tin trên cổng.

**III.4. Sử dụng phương thức Connect ở client để thiết lập kết nối trước với server**

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a black screen

AI-generated content may be incorrect.

**III.4.3. Trả lời câu hỏi:**

1. **Khi chạy chương trình mà server chưa được bật thì có hiện tượng gì xảy ra? Tại sao lại có hiện tượng này?**

* Khi server chưa bật, client vẫn gửi được dữ liệu bình thường (không báo lỗi). Tuy nhiên, server không nhận được gì, và client cũng không nhận được phản hồi.
* Vì UDP là giao thức không kết nối (connectionless), nên khi gọi Connect() chỉ là thiết lập địa chỉ mặc định để gửi dữ liệu, không kiểm tra xem server có tồn tại hay không. Do đó, gói tin vẫn được gửi đi, nhưng nếu server chưa chạy, nó sẽ bị “mất” (không ai nhận).

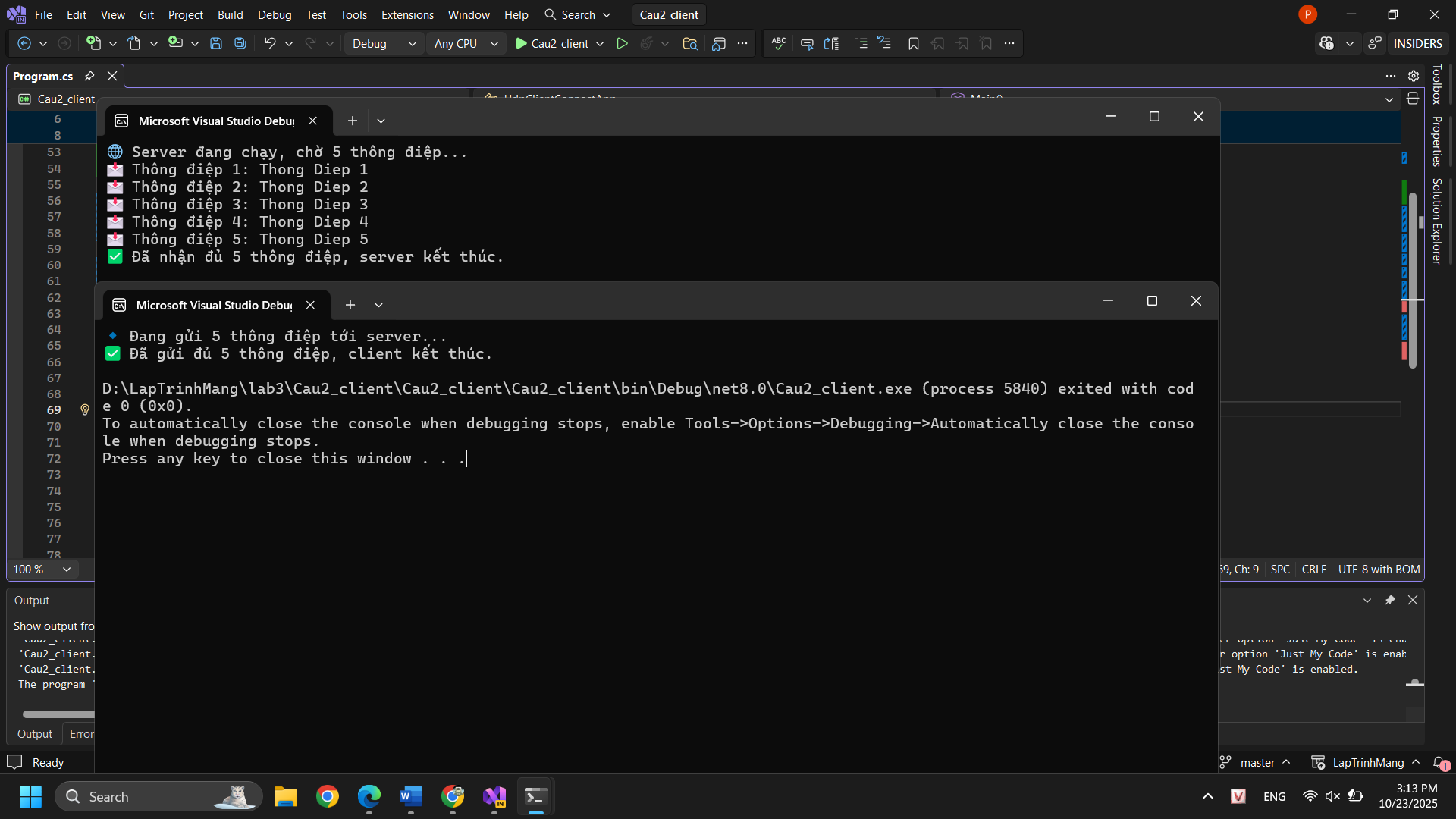
1. **Khi đang chạy tắt server thì chương trình trên có bị lỗi không? Tại sao?**

* Khi đang chạy mà server bị tắt, client không bị lỗi ngay — vì UDP không duy trì kết nối. Nhưng nếu client gửi dữ liệu sau khi server đã tắt, nó vẫn gửi được, chỉ là không nhận được phản hồi (nếu có lệnh Receive() thì có thể bị treo chờ phản hồi).
* UDP không kiểm tra trạng thái kết nối, nên không báo lỗi mất kết nối như TCP. Chỉ khi client chờ phản hồi (Receive) mà server đã tắt, thì hàm Receive() sẽ chờ vô hạn (treo chương trình).

1. **Khi đang chạy chương trình tắt client thì chương trình trên có bị lỗi không? Tại sao**

* Khi client tắt, server không bị lỗi. Server dừng nhận dữ liệu từ địa chỉ đó, và tiếp tục lắng nghe các client khác.
* UDP không giữ “trạng thái kết nối” giữa client–server, nên việc client tắt đi không ảnh hưởng gì đến server. Server vẫn hoạt động bình thường, vì mỗi gói tin UDP độc lập, không cần kết nối duy trì.

**III.5. Kiểm tra khả năng phân biệt biên thông điệp của giao thức UDP**



**III.5.4. Trả lời câu hỏi:  
1) Tại sao khi lập trình bằng giao thức UDP các thông điệp được phân biệt với nhau ?**

-Vì UDP là giao thức hướng thông điệp. Mỗi lần gửi sẽ tạo ra một gói tin riêng biệt, và mỗi lần nhận sẽ nhận đúng một gói tin, không bị ghép hay cắt như trong giao thức TCP.

**III.6. Ngăn cản mất dữ liệu khi lập trình mạng sử dụng giao thức UDP**

A computer screen shot of a program

AI-generated content may be incorrect.

1. **Khi gởi dữ liệu với kích thước lớn hơn 10 byte thì có lỗi xảy ra không? Tại sao?**

- Nếu dùng Socket.ReceiveFrom với byte[10] mà server gửi datagram >10 byte thì sẽ nảy SocketException trên Windows vì buffer nhỏ hơn datagram.

**III.6.2. Hướng dẫn lập trình để ngăn cản mất dữ liệu**

A screenshot of a computer

AI-generated content may be incorrect.

**III.6.4. Trả lời câu hỏi:**

1. **Khi dùng phương pháp này các dữ liệu bị mất do kích thước bộ đệm ban đầu nhỏ có lấy lại được không?**

-Không, Vì UDP không có cơ chế xác nhận, truyền lại hoặc đảm bảo toàn vẹn dữ liệu

**III.7. Ngăn cản mất gói tin khi lập trình mạng sử dụng giao thức UDP**

A screenshot of a computer

AI-generated content may be incorrect.

**III.7.3. Trả lời câu hỏi:**

1. **Sử dụng phương pháp này có ngăn cản triệt để mất dữ liệu không? Nếu không hãy đề xuất phương pháp khác**

-Phương pháp này không ngăn cản triệt để mất dữ liệu. Phương pháp khác đó là sử dụng TCP thay cho UDP vì TCP thích hợp khi yêu cầu truyền dữ liệu tin cậy.