**LAB 6**

**Concurrent Server - High-level Network programming**

| Họ tên và MSSV:  Nhóm học phần: |
| --- |

* *Các sinh viên bị phát hiện sao chép bài của nhau sẽ nhận 0đ cho tất cả bài thực hành của môn này.*
* *Bài nộp phải ở dạng PDF, hình minh họa phải rõ ràng chi tiết.* Hình minh hoạ chỉ cần chụp ở nội dung thực hiện, không chụp toàn màn hình.

1. **Bài 1**

Cập nhật phần Server của Bài 2-Lab05 và Bài 4-Lab05 để có thể phục vụ song song nhiều Client cùng một lúc (sử dụng kỹ thuật Multi-Threading).

(Chụp hình minh họa code bài làm và kết quả thực thi; đính kèm tập tin code khi nộp bài.)

1. **Bài 2**

Tham khảo ví dụ MessageServer.py và MessageClient.py, viết chương trình Chat đơn giản sử dụng UDP socket cho phép hai người trên hai máy tính trò chuyện với nhau. Lưu ý: tạo 2 thread (1 dùng để gửi, 1 để nhận thông điệp) để chương trình cho phép người dùng nhận và gửi thông điệp song song.

(Chụp hình minh họa code bài làm và kết quả thực thi; đính kèm tập tin code khi nộp bài.)

1. **Bài 3**

Sử dụng các thư viện hỗ trợ lập trình mạng ở mức độ High-level của Python để viết một chương trình EmailCrawler cho phép tìm các địa chỉ email trên một website.

Gợi ý:

* (1) Sử dụng hàm urllib.request.urlopen để load trang chủ của 1 website.
* (2) Tìm trong nội dung trang web tải xuống các email và URL. Đưa email và URL vào danh sách.
* (3) Tiếp tục tải các trang web có trong website sử dụng các URL tìm được trong danh sách. Lập lại đến khi tất cả hết các URL.
* Có thể sử hàm re.findall để tìm các email và URL các trang web. Ví dụ:
* emails = re.findall(r'[\w.+-]+@[\w-]+\.[\w.-]+', text)
* urls = re.findall\

('http[s]?://(?:[a-zA-Z]|[0-9]|[$-\_@.&+]|[!\*\(\),]|\

(?:%[0-9a-fA-F][0-9a-fA-F]))+', text)

(Chụp hình minh họa code bài làm và kết quả thực thi; đính kèm tập tin code khi nộp bài.)

--- Hết ---