BÀI TẬP TUẦN SỐ 4 – IT4062

Yêu cầu nộp bài:

- Đặt mã nguồn của mỗi chương trình vào thư mục riêng rẽ có tên như gợi ý ở dưới.
- Tạo Makefile để biên dịch đồng thời các chương trình với tên file chạy lần lượt là **server** và **client**
- Đóng gói các thư mục này vào file nén có tên theo dịnh dạng HotenSV_MSSV_HW04.zip. Ví dụ với bài tập tuần này, cấu trúc file nén nộp như sau:

```
HotenSV_MSSV_HW04.zip
      |-- TCP_Client
           |-- Các file mã nguồn
      |-- TCP_Server
           |-- Các file mã nguồn
      |-- Makefile
```

Mô tả bài tập

Viết ứng dụng sử dụng TCP Socket để truyền file từ client lên server. File có định dạng bất kỳ và kích thước file giới hạn tới 2^{32} byte.

- **Server:**
  - Khởi động với số hiệu cổng ứng dụng và thư mục lưu trữ là giá trị truyền qua tham số dòng lệnh:

    `$./server Port_Number Directory_name`

    Trong đó Directory_name là tên thư mục để lưu trữ file của client gửi lên

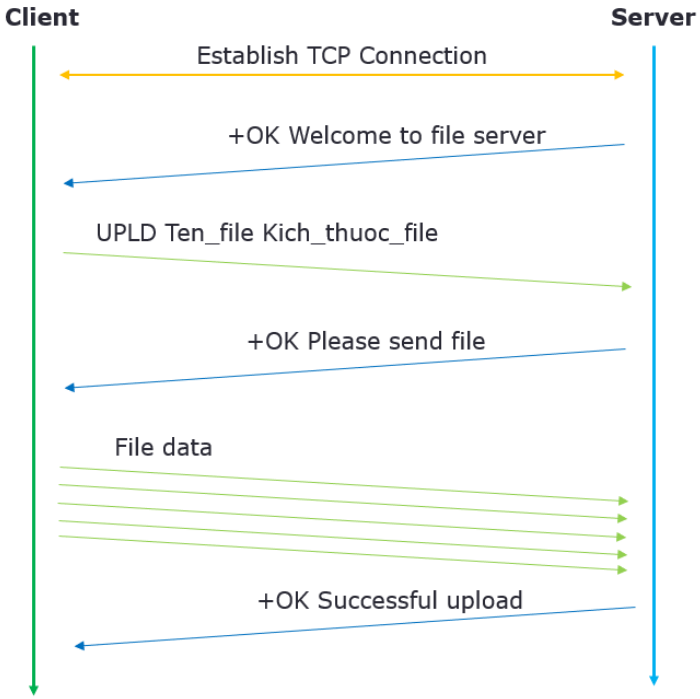
    `(Ví dụ: $./server 5500 storage)`
  - Sử dụng một thư mục có tên từ tham số dòng lệnh để lưu file của client gửi lên
  - Khi có client mới kết nối tới, gửi thông điệp chào mừng có nội dung **+OK Welcome to file server**
  - Nhận thông điệp báo tên file và kích thước(tính theo đơn vị byte) do client gửi lên. Gửi lại thông điệp **+OK Please send file**
  - Nhận dữ liệu của file mà client gửi lên và ghi vào file có tên mà client đã yêu cầu. Thực hiện chế độ ghi đè.
  - Nếu quả trình truyền file kết thúc thành công (nhận đủ số byte), gửi phản hồi **+OK Successful upload.**
  - Yêu cầu bắt buộc: **Chương trình server không được tự động kết thúc trong mọi tình huống xử lý.**
- **Client**:
  - Khởi động với địa chỉ server là các giá trị truyền qua tham số dòng lệnh:

    `$./client IP_Addr Port_Number (Ví dụ: $./client 10.0.0.1 5500)`
  - Nhận và hiển thị thông điệp chào mừng

- Nhận đường dẫn file do người dùng nhập. Gửi thông điệp tới server theo cú pháp **`UPLD Ten_file Kich_thuoc_file`**

  Ví dụ: **`UPLD test.txt 1024`**
- Gửi file lên cho server
- Hiển thị kết quả truyền file
- Chức năng lặp lại cho tới khi người dùng nhập vào đường dẫn file là xâu rỗng

Hoạt động của hai bên được mô tả trong hình vẽ sau:



Server cần ghi lại nhật ký hoạt động vào file có tên log_MSSV.txt., ví dụ log_20201234.txt
Mỗi dòng có cấu trúc như sau:
[dd/mm/yyyy hh:mm:ss]$Địa chỉ client$Yêu cầu nhận được$Kết quả truy vấn
Trong đó:

dd/mm/yyyy: Định dạng ngày nhận yêu cầu

hh:mm:ss: Định dạng thời điểm nhận yêu cầu

Địa chỉ client có định dạng *IPAddress:PortNumber*

Ví dụ:

[31/03/2023 14:42:24]$127.0.0.1:40000$+OK Welcome to file server

[31/03/2023 14:42:28]$ 127.0.0.1:40000$UPLD test.txt 1024\r\n$+OK Successful upload

Thang điểm:

- Điểm chức năng(FS):

  [1] Kết nối thành công và nhận thông điệp chào mừng: 1 điểm

  [2] Xử lý thành công thông điệp chứa tên file và kích thước: 1 điểm

  [3] Truyền file kích thước nhỏ(< 10KB) thành công: 1 điểm

  [4] Truyền file kích thước lớn tối đa 2^{32} byte thành công: 2 điểm

[5] Xử lý truyền dòng thành công: 2 điểm

[4] Một số lỗi sau đây bị trừ điểm:

- Server không phục vụ được liên tục cho nhiều client: -3 điểm
- Lỗi runtime error khiến client kết thúc: -1 điểm
- Lỗi runtime error khiến server kết thúc nhưng vẫn kiểm thử được các chức năng: -3 điểm
- Lỗi runtime error khiến không thể kiểm thử được tất cả chức năng: -100%
- Lỗi biên dịch: -100%
- Các lỗi khác: trừ điểm tùy theo mức độ nghiêm trọng của lỗi

- Điểm Tổ chức và trình bày mã nguồn(SS):

  Nếu $0 \leq FS < 1$: SS = 0

  Nếu $1 \leq FS < 3$: SS tối đa là 1 điểm

  Nếu $3 \leq FS \leq 5$: SS tối đa là 2 điểm

  Nếu $5 < FS \leq 7$: SS tối đa là 3 điểm

Yêu cầu môi trường:

- Hệ điều hành: Ubuntu 20.04
- Trình biên dịch: GCC

Gợi ý:

- Khi truyền file, đọc file theo từng khối dữ liệu để truyền đi. Kích thước khối nên đủ lớn (Ví dụ tối thiểu là 16KB) để hiệu năng truyền tốt.
- Một kịch bản kiểm thử có thể như sau:

Bước	Cửa sổ Client 1	Cửa sổ Client 2	Cửa sổ Client 3
1	Khởi động client 1		
2		Khởi động client 2	
3			Khởi động client 3
4			Gửi file nén
5		Gửi file ảnh	
6	Gửi file .txt		
7	Chờ hoàn tất Kết quả: Thành công. Có thể xem được file trên server		

8	Kết thúc chương trình		
9		Chờ hoàn tất Kết quả: Thành công. Có thể xem được ảnh trên server	
10		Kết thúc chương trình	
11			Chờ hoàn tất Kết quả: Thành công. Có thể giải nén được file trên server
12			Kết thúc chương trình