**LẬP TRÌNH MẠNG CƠ BẢN – CHUẨN ĐẦU RA VÀ   
HỆ THỐNG BÀI TẬP**

* Xác định file/folder có tồn tại hay không?
* Xác định các thuộc tính hệ thống (quyền đọc, ghi,..)của file/folder
* Tạo/xóa file/folder bất kỳ
* Tìm kiếm file/folder từ 1 thư mục bất kỳ theo tên hay phần mở rộng
* Hiển thị cấu trúc cây của thư mục
* Hiển thị 1 loại file cho trước (fileter)
* Phân biệt kiểu dữ liệu nhị phân và văn bản
* Các Stream và cách thức thao tác trên dữ liệu nhị phận
* Các Stream và cách thức thao tác trên dữ liệu nhị phận
* Phân biệt chuỗi (string) và dòng văn bản (line)
* Viết ứng dụng Copy/split/Join file nhị phân
* Lưu danh sách sinh viên xuống file và load được dữ liệu từ file này
* Import / export dữ liệu dưới dạng text
* Hiểu về định dạng dữ liệu -> viết ứng dụng Image Info
* Định nghĩa định dạng file nhị phân đơn giản để Lưu danh sách sinh viên xuống file và load được dữ liệu từ file này
* Khái niệm Object serialization và các vấn đề liện quan đến VersionID
* Hiểu rõ giao thức truyền dữ liệu
* TCP: kết nối, lựa chọn I/O phù hợp, gửi nhận dữ liệu
* Phục vụ nhiều Client đồng thời
* Xây dựng giao thức (truyền dữ liệu Text + Nhị phân)

Bài tập:

**Phần File Class:**

1. **Bài 1**: (1)Hiện thực hàm **boolean delete(String path)** xóa tất cả những gì có thể trong thư mục được chỉ định bởi path; trả về **true** nếu xóa thành công, **false** nếu xóa không thành công  
   Mở rộng: Chỉ xóa files giữa nguyên cấu trúc thư mục
2. Bài 2: Hiện thực **hàm boolean findFirst(String path, String pattern)** tìm và hiển thị đường dẫn đầy đủ file/folder chỉ định bởi path có chứa chuỗi quy định bởi pattern; trả về true nếu tìm thấy, false nếu không tìm thấy
3. Bài 3 (**Advanced**): Viết hàm hiển thị cấu trúc cây của thư mục void dirTree(String path); dùng các ký tự + - | để vẽ cấu trúc cây. Cần hiển thị được cấp con hay ngang cấp,… (tương tự lệnh tree trong DOS)
4. Bài 4 (Advanced): Viết hàm tính và hiển thị dung lượng theo cấu trúc cây thư mục **void dirStat(String path)**
5. Bài XX: Hiện thực **hàm void findAll(String path, String ext1, String ext2, …)** tìm và hiển thị đường dẫn đầy đủ file/folder chỉ định bởi path có phần mở rộng quy định bởi ext1, ext2,…, extn;
6. Bài XX: Hiện thực **hàm void deleteAll(String path, String ext1, String ext2, …)** tìm và xóa file/folder chỉ định bởi path có phần mở rộng quy định bởi ext1, ext2,…, extn;
7. Bài XX: Hiện thực **hàm void copyAll(String sDir, String dDir String ext1, String ext2, …)** copy từ thư mục nguồn sDir vào thư mục đích dDir tất cả các file có phần mở rộng quy định bởi ext1, ext2,…, extn;

**Phần 1 IO**

1. **Bài 5**: Viết CT copy/move file dùng byte array kết hợp với BIS, BOS:   
   **boolean fileCopy(String sFile, String destFile, boolean moved);**
2. Bài 6: Viết CT copy/move thư mục dùng byte array kết hợp với BIS, BOS:   
   **boolean folderCopy(String sFolder, String destFolder, boolean moved);**
3. **(2)Viết CT File Spliter** chia 1 file thành nhiều phần theo dung lượng hoặc số lượng. **Viết CT File Joiner** ghép các file thành phần thành file ban đầu.
4. Viết ứng dụng String fileType(String fname) xác định file bất kỳ có phải là: zip, rar, doc, … (Kiểu file)
5. **(3)Viết CT lưu/Đọc danh sách sinh viên xuống file nhị phân** (Lưu từng thuộc tính): Danh sách sinh viên bất kỳ; mỗi sinh viên có danh sách điểm bất kỳ
6. **(4)** Cho Danh sách sinh viên bất kỳ; mỗi sinh viên có danh sách điểm bất kỳ. Viết phương thức lưu danh sách sinh viên này xuống file sao cho có thể load sinh viên thứ n bất kỳ mà không phải load thông tin của n-1 sinh viên trước đó (dùng RAF)
7. **(5) Viết CT Pack/Unpack** lưu nguyên thư mục vào 1 file duy nhất, trích rút từng file cụ thể (tương tự zip nhưng không nén) (giai đoạn 1 giả sử thục mục không chức thu mục con và restore nguyên cả thư mục)
8. Dùng RAF Viết CT lưu/Đọc danh sách sinh viên xuống file nhị phân (Lưu từng thuộc tính); chương trình hỗ trợ đọc/thay đổi thông tin trực tiếp của phần tử thứ n bất kỳ mà không cần đọc các dữ liệu trước đó
9. **(6)Viết CT Import/Export** danh sách sinh viên from/to text file với bảng mã cho trước. Export/Import dữ liệu này vào excel để xử lý và ngược lại

**Student(id,name,grade)**

**(Lưu mỗi sv là 1 dòng văn bản, mỗi trường ngăn cách nhau bởi dấu tab.**

**Import vào file excel, mở dc trong file excel)**

**-extend: lưu ra 2 file, sv và điểm.**

**Phần 2 Socket**

1. Viết chương trình client/server copy file/folder qua mạng. Đích và nguồn chỉ định trực tiếp trong mã nguồn của client/server
2. (**7)Viết CT client/server upload/download file lên server**. Người dùng tương tác với clietn qua console với cú pháp lệnh: copy source\_file dest\_file
3. Viết CT client/server download file từ server. Người dùng tương tác với client qua console với cú pháp lệnh: get source\_file dest\_file
4. **(8)Viết server hỗ trợ các phép tính cơ bản (+-\*/)** client tương tác với server bằng giao thức TCP thông qua các dòng văn bản. Người dùng nhập lệnh trên client có dạng:

1 + 2 rồi nhấp enter, client gửi dòng dữ liệu tới server, server trả về kết quả có dạng:

1 + 2 = 3; xử lý các ngoại lệ các toán hạng không phải là số và phép chia cho 0.

1. Viết CT client/server ra cứu thông tin sinh viên qua mạng. Dữ liệu trên server lưu vào arraylist; Người dùng tương tác với client qua console với cú pháp lệnh:   
   findByName name  
   findByAge age  
   findByScore score
2. **(9)Viết CT client/server ra cứu thông tin sinh viên qua mạng** có hỗ trợ đăng nhập tương tự POP3. Dữ liệu trên server lưu vào arraylist; Người dùng tương tác với client qua console với cú pháp lệnh:   
   user user\_name  
   pass password

Sau khi đăng nhập thành công có thể thực hiện các thao tác:

* FindById id  
  - findByName name  
  - findByAge age  
  - findByScore score

1. Viết lại bài tập Echo sử dụng UDP
2. **(10) Sử dụng UDP làm lại bài tập (7) + (8)**

**Phần 3 CSDL**

1. **(11)Bài tập tra cứu thông tin sinh viên với dữ liệu trên server dùng CSDL** (access hoặc postgresql)

**Phần 4 RMI**

1. **(12) Bài tập copy, tra cứu thông tin dùng RMI**