

PHẦN CƠ BẢN.

Bài 1: Tính tổng hai số nguyên

Yêu cầu:

- **Client:** Nhập hai số nguyên từ bàn phím, gửi tới server.
- **Server:** Nhận 2 số, tính tổng và trả kết quả về client.
- **Client:** Hiển thị kết quả tổng từ server.

Kiến thức toán học: Cộng 2 số nguyên.

Mục tiêu socket: Kết nối, gửi và nhận dữ liệu dạng số nguyên.

Bài 2: Kiểm tra số nguyên tố

Yêu cầu:

- **Client:** Nhập 1 số nguyên, gửi tới server.
- **Server:** Kiểm tra xem số đó có phải là số nguyên tố không, trả kết quả dạng “Prime” hoặc “Not Prime”.

Kiến thức toán học: Kiểm tra số nguyên tố.

Mục tiêu socket: Xử lý logic ở server, trả phản hồi dạng chuỗi.

Bài 3: Giải phương trình bậc hai

Yêu cầu:

- **Client:** Nhập 3 hệ số a, b, c, gửi tới server.
- **Server:** Giải phương trình $ax^2+bx+c=0$, trả nghiệm (nếu có) về client.

Kiến thức toán học: Phân biệt Δ , tính nghiệm thực hoặc phức.

Mục tiêu socket: Truyền và xử lý mảng dữ liệu thực (float), nhiều giá trị trả về.

Bài 4: Tính tổng dãy số

Yêu cầu:

- **Client:** Nhập một dãy số (cách nhau bằng dấu phẩy), gửi tới server.

- **Server:** Tính tổng các phần tử trong dãy, trả về kết quả.

Kiến thức toán học: Tổng dãy số.

Mục tiêu socket: Xử lý chuỗi, tách số từ chuỗi, tổng hợp.

Bài 5: Máy chủ giải toán biểu thức hậu tố (postfix)

Yêu cầu:

- **Client:** Gửi một biểu thức hậu tố, ví dụ: "3 4 + 2 *"
- **Server:** Phân tích và tính giá trị biểu thức, gửi kết quả về client.

Kiến thức toán học: Biểu thức hậu tố, ngăn xếp (stack).

Mục tiêu socket: Phân tích biểu thức, xử lý ngăn xếp trên server.

Bài 6: Kiểm tra tính đối xứng số

Viết chương trình TCP client-server:

- **Client** gửi một số nguyên dương đến **server**.
- **Server** kiểm tra xem số đó có phải là **số đối xứng** (palindrome) hay không, và gửi kết quả về client.

Ví dụ: 121 là đối xứng, 123 thì không.

Bài 7: Máy chủ chuyển đổi hệ số

- **Client** gửi một số nguyên dương và hệ đếm muốn chuyển đổi (2, 8 hoặc 16) đến **server**.
- **Server** trả về biểu diễn của số đó trong hệ nhị phân, bát phân hoặc thập lục phân tương ứng.

Bài 8: Tính giai thừa song song

- **Client** gửi một số nguyên nnn đến **server**.
- **Server** chia công việc tính $n!$ thành các đoạn nhỏ (nhiều thread hoặc xử lý tuần tự tùy yêu cầu), sau đó gửi kết quả về cho client.

Bài 9: Tính trung bình danh sách số

- **Client** gửi một danh sách các số thực (dạng chuỗi: "2.5,3.7,5.0") đến **server**.
- **Server** tính **giá trị trung bình, số lớn nhất, số nhỏ nhất**, và gửi tất cả các kết quả về client.

Bài 10: Kiểm tra số Fibonacci

- **Client** gửi một số nguyên dương n đến **server**.
- **Server** kiểm tra xem số đó có thuộc dãy **Fibonacci** hay không và gửi kết quả về client.

PHẦN NÂNG CAO:

Bài 11: Hệ phân tán tính định thức ma trận lớn

- **Client** gửi một ma trận vuông kích thước $n \times n$ (dạng chuỗi JSON hoặc binary) đến server.
- **Server** phân chia thành nhiều **nhánh con** (child sockets hoặc thread workers) để tính định thức theo phép khử Gauss hoặc Laplace expansion.
- Sau khi tính xong, kết quả được **tổng hợp** và gửi lại cho client.
- Yêu cầu: Kết quả chính xác với ma trận kích thước $n \geq 10$.

Bài 12: Giao thức xác thực số học ba bước (Three-way Arithmetic Handshake)

- **Client** gửi một số bí mật x .
- **Server** gửi lại một phép toán ngẫu nhiên (ví dụ: " $2x + 3$ ", " $x^2 - 4x + 1$ "), client phải **giải ngược lại** để tìm x , rồi xác minh.
- Nếu client giải đúng trong 2 lần, server mở cổng gửi bài toán chính (challenge).
- Đây là hệ thống xác thực client theo kiểu **Zero Knowledge Proof** nhẹ nhàng.

Bài 13: Truy vấn số nguyên tố trong khoảng bằng luồng liên tục

- Client gửi một cặp (a, b) và giữ kết nối mở.
- **Server** gửi lần lượt từng số nguyên tố trong khoảng $[a, b]$ theo kiểu **TCP streaming** (chunk hoặc socket send theo dòng).
- Yêu cầu:
 - Giữa mỗi lần gửi chờ 1 giây.
 - Client có thể gửi **"STOP"** bất kỳ lúc nào để ngắt luồng.
 - Không dùng gửi hết rồi mới trả — phải **duy trì phiên làm việc lâu dài**.

Bài 14: Mã hóa Hill Cipher trên mạng

- **Client** gửi 1 văn bản và ma trận khóa K (kích thước 2×2 hoặc 3×3) đến server.
- **Server** thực hiện mã hóa Hill Cipher (yêu cầu: $\det(K) \not\equiv 0 \pmod{26}$) rồi gửi kết quả về.
- Sau đó, client tiếp tục gửi mã đã mã hóa và yêu cầu giải mã, server thực hiện **tìm nghịch đảo** của ma trận khóa rồi giải mã và gửi lại kết quả.

Bài 15: Giải hệ phương trình tuyến tính qua mạng

- **Client** gửi hệ phương trình tuyến tính gồm n ẩn và n phương trình (dạng JSON: danh sách ma trận hệ số A và vector hằng số B).
- **Server** sử dụng phương pháp **Gauss-Jordan** hoặc **pivoting** để giải $AX = B$, gửi kết quả vector nghiệm X về client.
- Yêu cầu: Xử lý sai định dạng, ma trận suy biến ($\det = 0$), hoặc thiếu số phương trình.