Đề 01

Lập 2 chương trình khác nhau trên ngôn ngữ lập trình tùy chọn C++/Java/C# với yêu cầu sau:

Lập chương trình CT1(có hàm main):

1. (2 điểm) Viết 1hàm gồm 1 vòng lặp tạo liên tục 1 số nguyên không âm (số nguyên 4byte),

nếu số được tạo chia hết cho 1024 thì thoát khỏi vòng lặp, trái lại thi ghi số nguyên vào file dulieu.txt (file mở ở mode ghi đè nếu file đã tồn tại),

khi ghi được thì đóng file.

2. (3 điểm) Đặt hàm vào 1 luồng, kích hoạt luồng trong hàm main().

Lập chương trình CT2 (có hàm main):

3. (2 điểm) Viết 1 hàm có 1 vòng lặp vô hạn đọc số nguyên 4 byte ở file dulieu.txt.

Đọc được số nguyên thì đóng file. Hiển thị giá trị đọc được. Nếu số nguyên đọc được chia hết cho 1024 thì thoát vòng lặp.

4. (3 điểm) Đặt hàm vào 1 luồng, kích hoạt luồng trong hàm main().

Chạy cả CT1 và CT2 đồng thời để thấy hiệu ứng trao đổi dữ liệu.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 02

Lập 1 chương trình trên ngôn ngữ lập trình tùy chọn C++/Java/C# gồm 2 tác vụ (Task1, Task2) hoạt động ở chế độ không đồng bộ như sau:

Task1:

1. (2 điểm) Viết 1 hàm gồm 1 vòng lặp nhập 1 xâu kí tự string từ bàn phím (string là một biến tổng thể).

Loại bỏ các dấu cách ở đầu và cuối xâu string. Nếu xâu string được nhập và xử lý cắt dấu cách là “exit” thì thoát khỏi vòng lặp.

2. (3 điểm) Thiết lập Task1 bằng cách đặt hàm trên vào 1 timer hoặc 1 luồng, kích hoạt timer/luồng trong hàm main().

Task2:

3. (2 điểm) Viết 1 hàm có 1 vòng lặp vô hạn hiển thị biến xâu kí tự string (xem câu 1). Nếu string là “exit” thì thoát vòng lặp. Bổ sung lệnh gọi hàm trong hàm main().

4. (3 điểm) Thiết lập Task2 bằng cách đặt hàm trên vào 1 timer hoặc 1 luồng, kích hoạt timer/luồng trong hàm main(). Dịch và chạy chương trình.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 03

Lập 2 chương trình khác nhau trên ngôn ngữ lập trình tùy chọn C++/Java/C# gồm CT1(client), CT2 (server)

sử dụng Socket API, giao thức TCPIP truyền nhận dữ liệu tại IP="127.0.0.1" (localhost), port=100 và hoạt động như sau:

CT1 (có hàm main):

1. (2 điểm) Viết 1 hàm gồm 1 vòng lặp tạo ra sô nguyên ngẫu nhiên n,

vòng lặp dừng khi số nguyên sinh ra lớn hơn 20000 và chia hết cho 1024.

Viết hàm main() gọi hàm.

2. (3 điểm) Bổ sung lệnh truyền giá trị số nguyên sinh ra cho CT2 trong vòng lặp.

Dịch chương trình CT1.

CT2 (có hàm main):

3. (4 điểm) Viết 1 hàm có 1 vòng lặp vô hạn nhận giá trị biến nguyên được gửi đến từ CT1.

Kiểm tra, nếu CT2 nhận được giá trị nguyên k thì:

Trường hợp 1: Nếu n > 20000 và chia hết cho 1024 thì thoát khỏi vòng lặp.

Trường hợp 2: Trái lại, hiển thị giá trị lớn nhất được CT1 sinh ra tính tại thời điểm đang chạy (giá trị số lớn nhất có thể luôn thay đổi khi CT1 đang chạy).

Viết hàm main(), dịch chương trình CT2. Rồi chạy CT2 và chạy CT1 để xem hiệu ứng nhập mảng ở CT1, hiển thị mảng ở CT2.

4. (1 điểm)

Đặt hàm ở câu 3 vào 1 timer, bổ sung lệnh kích hoạt timer trong hàm main(). Dịch chương trình.

Biên dịch và chạy lần sau cùng:

+ Đóng tất cả CT1, CT2 đang chạy.

+ Chạy chương trình CT2 sau đó chạy CT1 để xem hiệu ứng kết nối, truyền và nhận dữ liệu.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 04

Lập 2 chương trình gồm CT1(client) và CT2 (server), sử dụng giao thức TCPIP cho các hàm SocketAPI với tham số:

socket ip="127.0.0.1", port=2014 cho CT2(server),

+ CT1 có 1 vòng lặp vô hạn nhập 1 mảng số nguyên gồm n phần tử như sau:

Nhập n, nếu n <=0 thì gửi sang CT2 giá trị n bằng socketAPI rồi thoát khỏi vòng lặp.

Nhập tiếp n phần tử của mảng số nguyên, rồi gửi sang CT2 mảng số nguyên bằng socketAPI.

+ CT2. có vòng lặp, chờ kết nối CT1, nhận giá trị n

Nếu n<=0 thì thoát khỏi vòng lặp.

Nhận mảng n phần tử số nguyên. Hiển thị các giá trị nhận được.

Dich CT2 và CT1. Sau đó chạy CT2 rồi chạy CT1.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 05:

Lập trình trên ngôn ngữ tùy chọn với yêu cầu sau:

Viết CT1.

a (2đ). Viết 1 hàm gồm 1 vòng lặp nhập 1 kí tự từ bàn phím, ghi kí tự lên 1 file dulieu.txt. Nếu kí tự là '!' thì thoát khỏi vòng lặp.

b. (4đ) Đặt hàm vào 1 luồng, dịch chương trình CT1.

Viết chương trình 2.

c (2đ). Viết 1 hàm có 1 vòng lặp vô hạn đọc kí tự ở file dulieu.txt. Đọc được kí tự thì đóng file. Hiển thị kí tự. Nếu kí tự đọc được là '!' thì thoát vòng lặp.

d. (4đ) Đặt hàm vào 1 luồng, dịch chương trình CT2.

Chạy CT1 và CT2 để thấy hiệu ứng trao đổi dữ liệu.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 06:

Lập trình trên ngôn ngữ tùy chọn (C/C#/Java) với yêu cầu sau:

Viết chương trình gồm 2 tác vụ hoạt độngkhông đồng bộ như sau:

Task1:

a (2đ). Viết 1 hàm gồm 1 vòng lặp nhập 1 kí tự từ bàn phím, ghi kí tự lên 1 file dulieu.txt. Nếu kí tự là '!' thì thoát khỏi vòng lặp.

b. (4đ) Đặt hàm vào 1 timer cứ 5ms (1ms=1/1000 giây) thực hiện 1 lần.

Task2

c (2đ). Viết 1 hàm có 1 vòng lặp vô hạn đọc kí tự ở file dulieu.txt. Đọc được kí tự thì đóng file. Hiển thị kí tự. Nếu kí tự đọc được là '!' thì thoát vòng lặp.

d. (4đ) Đặt hàm vào 1 timer cứ 10ms thực hiện 1 lần..

Ghi chú: có thể sử dụng timer dựa trên thread để cứ 1 lần chu kỳ lại "nhả" CPU cho hệ điều hành hoặc sử dụng timer được hỗ trợ sẵn bởi môi trường lập trình.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 07:

Lập trình trên ngôn ngữ tùy chọn (C/C#/Java) với yêu cầu sau:

Viết chương trình gồm 2 timer hoạt động như sau:

timer1:

a (2đ). Viết 1 hàm gồm 1 vòng lặp nhập 1 kí tự từ bàn phím, lưu vào 1 biến tổng thể c, hiển thị kí tự. Nếu kí tự là '!' thì thoát khỏi vòng lặp.

b. (4đ) Đặt hàm vào 1 timer cứ 7ms (1ms=1/1000 giây) thực hiện 1 lần.

timer2

c (2đ). Viết 1 hàm có 1 vòng lặp phát ra tiếng beep, nếu biến tổng thể c là '!' thì thoát vòng lặp.

d. (4đ) Đặt hàm vào 1 timer cứ 10ms thực hiện 1 lần..

Chạy chương trình để xem hiệu ứng timer1, timer2.

Ghi chú: có thể sử dụng timer dựa trên thread để cứ 1 lần chu kỳ lại "nhả" CPU cho hệ điều hành hoặc sử dụng timer được hỗ trợ sẵn bởi môi trường lập trình.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 08:

Lập trình trên ngôn ngữ tùy chọn (C/C#/Java) với yêu cầu sau:

Viết 2 chương trình gồm CT1(client) , CT2 (server) sử dụng Socket API, giao thức TCPIP truyền nhận dữ liệu tại IP="127.0.0.1", port=100, hoạt động như sau:

CT1:

a(2đ). Viết 1 hàm gồm 1 vòng lặp vô hạn nhập 1 kí tự từ bàn phím, khi nhập phím ESC thì thoát khỏi vòng lặp. Viết hàm main() gọi hàm, dịch và chạy chương trình.

b(2đ).Bổ sung lệnh truyền byte kí tự đến cho CT2 trong vòng lặp.

c(1đ) Đặt hàm vào 1 thread sao cho CT1 không chiếm 100% CPU khi chạy.

Dịch và chạy thử CT1 (chưa có hiệu ứng truyền dữ liệu)

CT2:

d (4đ). Viết 1 hàm gồm 1 vòng lặp vô hạn nhận 1 byte kí tự. Nếu nhận được kí tự thì:

TH1: Nếu kí tự là phím ESC thì thoát khỏi vòng lặp.

TH2: Nếu kí tự không là ESC thì hiển thị kí tự.

Viêt hàm main() gọi hàm.

e(1đ). Đặt hàm vào 1 thread sao cho CT2 không chiếm 100% CPU khi chạy.

Biên dịch lần cuối:

+ Đóng tất cả CT1,CT2 đang chạy.

+ Chạy chương trình CT2 sau đó chạy CT1 để xem hiệu ứng kết nối, truyền và nhận dữ liệu.

+ Chạy Task manager (nếu chạy trên hệ điều hành Windows) để xem CT1 và CT2 khi chạy chiếm bao nhiêu % CPU.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 09:

Lập 2 chương trình trên ngôn ngữ lập trình tùy chọn C++/Java/C#/Python

gồm CT1(client), CT2 (server) sử dụng Socket API, giao thức TCPIP truyền nhận dữ liệu tại IP="127.0.0.1"(localhost), port=100 và hoạt động như sau:

CT1:

1. Viết 1 hàm gồm 1 vòng lặp nhập vào từ bàn phím mảng số thực 64bit (nhập số phần tử n>=2, sau đó nhập từng phần tử mảng), nếu nhập n thỏa n < 2 thì thoát khỏi vòng lặp.

2. Bổ sung lệnh truyền giá trị n và truyền biến mảng đến cho CT2 trong vòng lặp.

CT2:

3. Viết 1 hàm có 1 vòng lặp vô hạn nhận giá trị biến nguyên n được gửi đến từ CT1, Nếu nhận được n thì nếu n < 2 thì thoát khỏi vòng lặp.Trái lại, nhận tiếp mảng n số thực 64bit được gửi đến từ CT1 và

hiển thị 2 giá trị đầu tiên xếp theo thứ tự từ cao xuống thấp của mảng sau khi nhận được (2 giá trị đầu tiên này có thể luôn thay đổi khi CT1 đang chạy).

Dịch và chạy CT2 rồi CT1 để kiểm tra.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 10: <bai8>

Lập 2 chương trình gồm CT1(client) và CT2 (server),

CT1 có 1 vòng lặp thực hiện gửi file ảnh 'I04.JPG; qua socket ip="127.0.0.1", port=2014

CT2(server), khi nhận đc file thì thoát khỏi vòng lặp

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 11:

Viết chương trình tạo 2 luồng Thr1 và Thr2, mỗi luồng gồm 1 vòng lặp, sao cho:

+ Trong hàm main() khai báo 2 biến mảng a và b của các số nguyên (mảng không quá 10 phần tử)

+ Thr1 có vùng nhớ riêng là vùng nhớ cho a.

+ Thr2 có vùng nhớ riêng là vùng nhớ cho b.

Khi Thr1 hoạt động thì người dùng nhập số phần tử của mảng a, và Thr2 sinh ra số n ngẫu nhiên và tiếp tục n phần tử của mảng b.

Khi nhập số phần tử không vượt quá 10 thì Thread tương ứng dừng.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 12:

Viết chương trình gồm 2 luồng, mỗi luồng có 1 vòng lặp.

+ Luồng 1 thực hiện nhập phím từ bàn phím, nếu người dùng nhấn phím ESCAPE thì thoát khỏi vòng lặp,

trái lại thì nạp kí tự vào 1 hàng đợi (queue).

+ Luồng 2, hiển thị kí tự trong hàng đợi bàn phím theo thứ tự lưu trữ vào hàng đợi.

+ Khi luồng 1 dừng thì luồng 2 cũng dừng.

+ Viết hàm main() khởi tạo và kích hoạt các luồng trên. Dịch và chạy chương trình.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 13:

Bài tập 19. Viết 2 CT, CT1 theo chu kỳ 10ms thì tạo ra 1 số ngẫu nhiên và ghi lên file vd.dat (khi ghi xóa nội dung cũ),

CT2 theo chu kỳ 5ms thì đọc file vd.dat, và hiển thị số nguyên đọc được lên màn hình.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 14:

Lập 1 chương trình trên ngôn ngữ lập trình tùy chọn C++/Java/C# gồm 2 tác vụ (Task1, Task2) hoạt động ở chế độ không đồng bộ như sau:

Task1:

a. Viết 1 hàm gồm 1 vòng lặp sinh số nguyên ngẫu nhiên cho đến khi gặp số chia hết cho 10011 thì dừng lại.

Task2:

b. Viết 1 hàm có 1 vòng lặp chỉ dừng lặp khi vòng lặp của Task1 dừng lại. Vòng lặp này luôn hiển thị số chia hết cho 3 mà Task1 tạo ra tại mọi thời điểm chạy.

Bổ sung lệnh gọi hàm trong hàm main(), dịch và chạy lại chương trình.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 15:

Viết chương trình tạo 2 luồng Thr1 và Thr2, mỗi luồng gồm 1 vòng lặp, sao cho:

+ Trong hàm main() khai báo 2 biến mảng a và b của các số nguyên (mảng không quá 10 phần tử)

+ Thr1 có vùng nhớ riêng là vùng nhớ cho a.

+ Thr2 có vùng nhớ riêng là vùng nhớ cho b.

Khi Thr1 hoạt động thì người dùng nhập số phần tử của mảng a, và Thr2 sinh ra số n ngẫu nhiên và tiếp tục n phần tử của mảng b.

Khi nhập số phần tử không vượt quá 0 thì Thread tương ứng dừng.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 16: Lập một chương trình trên ngôn ngữ lập trình tùy chọn tạo ra 2 luồng(Thread1, Thread2) hoạt động như sau:

Thread1:

1. Viết một hàm gồm một mở 1 file chằng hạn vd.txt ở chế độ text (giả định đây là môt file có kích thước rất lớn).

Hàm có 1 vòng lặp đọc từng ký tự của file và hiển thị dạng mã hexa của ký tự, vòng lặp dùng khi không đọc tiếp được (đã đọc hết).

2. Đặt hàm trên vào 1 luồng, kích hoạt luồng trong hàm main().

Thread2:

3. Viết 1 hàm có 1 vòng lặp chỉ dùng khi vòng lặp của Thread 1 dùng lại. Vòng lặp này luôn hiển thị các ký tự đã đọc được từ file ở Thread1.

Bổ sung lệnh gọi hàm trong hàm main().

4. Thiết lập Thread2 bằng cách đặt hàm trên vào một luồng, sửa hàm main() để kích hoạt luồng Thread2 trước rồi kích hoạt luồng Thread1. Dịch và chạy chương trình.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 17: Lập trình trên ngôn ngữ tùy chọn (C/C#/Java) với yêu cầu sau:

Viết 2 chương trình gồm CT1(client) , CT2 (server) sử dụng Socket API, giao thức TCPIP truyền nhận dữ liệu tại IP="127.0.0.1", port=100, hoạt động như sau:

CT1:

1. (2 điểm) Viết 1 hàm gồm 1 vòng lặp nhập vào từ bàn phím mảng số nguyên (nhập số phần tử n, sau đó nhập từng phần tử mảng),

nếu nhập n thỏa n < 0 thì thoát khỏi vòng lặp. Viết hàm main() gọi hàm, dịch và chạy chương trình.

2. (3 điểm) Bổ sung lệnh truyền giá trị n và truyền n phần tử mảng theo thứ tự ngược lại đến cho CT2 trong vòng lặp.

Dịch và chạy lại chương trình CT1.

CT2:

3. (4 điểm) Viết 1 hàm có 1 vòng lặp vô hạn nhận giá trị biến nguyên n được gửi đến từ CT1, Nếu nhận được n thì:

Trường hợp 1: Nếu n < 0 thì thoát khỏi vòng lặp.

Trường hợp 2: Trái lại, nhận tiếp mảng n số nguyên được gửi đến từ CT1, đảo ngược thứ tự nhận được của n phần tử nhận được và hiển thị giá trị hexa của n phần tử.

Viết hàm main(), dịch chương trình CT2.

Rồi đóng CT1 đang chạy, sau đó lần lượt chạy CT2 và chạy CT1 để xem hiệu ứng nhập mảng ở CT1, hiển thị mảng ở CT2.

4. (1 điểm) Đặt hàm ở câu 3 vào 1 thread, bổ sung lệnh kích hoạt luồng trong hàm main(). Dịch chương trình.

Biên dịch lần cuối:

+ Đóng tất cả CT1, CT2 đang chạy.

+ Chạy chương trình CT2 sau đó chạy CT1 để xem hiệu ứng kết nối, truyền và nhận dữ liệu.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Đề 18: Lập 1 chương trình trên ngôn ngữ lập trình tùy chọn C++/Java/C# gồm 2 tác vụ (Task1, Task2)

hoạt động ở chế độ không đồng bộ như sau:

Task1:

1. Viết 1 hàm gồm 1 vòng lặp nhập liên tục các số nguyên từ bàn phím, lưu các số

nhập được vào một biến cấu trúc A kiểu bộ đệm vòng, và vòng lặp dùng lại khi

nhập số nguyên < 0.

2. Thiết lập Task1 bằng cách đặt hàm trên vào 1 luồng, kích hoạt luồng trong main().

Task2:

3. Viết 1 hàm vó 1 vòng lặp chỉ dùng lại khi vòng lặp của task1 dừng lại.

Vòng lặp này luôn hiển thị các số nguyên được lưu trong biến A.

Bổ sung lệnh gọi hàm trong hàm main().

4. Thiết lập Task2 bằng cách đặt hàm trên vào 1 luồng, kích hoạt luồng trong main().

Hiển thị trung bình cộng các phần tử của mảng. Dịch và chạy chương trình.