Sort an array using socket programming in C

ajay0007

Given an array of unsorted positive integer, sort the given array using the Socket programming . Examples:

```
Input : 4 5 6 1 8 2 7 9 3 0
Output :0 1 2 3 4 5 6 7 8 9

Input : 9 8 1 4 0
Output : 0 1 4 8 9
```

Compile these files using **gcc** command (gcc client.c -o client and gcc server.c -o server). Run the program using **./server** and **./client** (Please note : First you should run server program which will be waiting for client's response and then client code).

Recommended: Please try your approach on ***{IDE}*** first, before moving on to the solution.

In this program, client will take the input and send it to server and the server will sort the array using the bubble sort.

C

```c
// Client code in C to sort the array
#include <arpa/inet.h>
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <unistd.h>

// Driver code
int main(int argc, char* argv[])
{
    int sock;
    struct sockaddr_in server;
    int server_reply[10];
    int number[10] = { 5, 4, 3, 8, 9, 1, 2, 0, 6 }, i, temp;

    // Create socket
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if (sock == -1) {
        printf("Could not create socket");
    }
    puts("Socket created");

    server.sin_addr.s_addr = inet_addr("127.0.0.1");
    server.sin_family = AF_INET;
    server.sin_port = htons(8880);

    // Connect to remote server
    if (connect(sock, (struct sockaddr*)&server, sizeof(server)) < 0) {
        perror("connect failed. Error");
        return 1;
    }

    puts("Connected\n");

    if (send(sock, &number, 10 * sizeof(int), 0) < 0) {
        puts("Send failed");
        return 1;
    }

    // Receive a reply from the server
    if (recv(sock, &server_reply, 10 * sizeof(int), 0) < 0) {
        puts("recv failed");
        return 0;
    }

    puts("Server reply :\n");
    for (i = 0; i < 10; i++) {
        printf("%d\n", server_reply[i]);
    }

    // close the socket
    close(sock);
    return 0;
```

```
}
```

Note : Save above file as client.c

C

```c
// Server code in C to sort the array
#include <arpa/inet.h>
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <unistd.h>

void bubble_sort(int[], int);

// Driver code
int main(int argc, char* argv[])
{
    int socket_desc, client_sock, c, read_size;
    struct sockaddr_in server, client;
    int message[10], i;

    // Create socket
    socket_desc = socket(AF_INET, SOCK_STREAM, 0);
    if (socket_desc == -1) {
        printf("Could not create socket");
    }
    puts("Socket created");

    // Prepare the sockaddr_in structure
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = INADDR_ANY;
    server.sin_port = htons(8880);

    // Bind the socket
    if (bind(socket_desc, (struct sockaddr*)&server, sizeof(server)) < 0) {

        // print the error message
        perror("bind failed. Error");
        return 1;
    }
    puts("bind done");

    // listen to the socket
    listen(socket_desc, 3);

    puts("Waiting for incoming connections...");
    c = sizeof(struct sockaddr_in);

    // accept connection from an incoming client
    client_sock = accept(socket_desc, (struct sockaddr*)&client, (socklen_t*)&c);

    if (client_sock < 0) {
        perror("accept failed");
        return 1;
```

```
    }

    puts("Connection accepted");

    // Receive a message from client
    while ((read_size = recv(client_sock, &message, 10 * sizeof(int), 0)) > 0) {

        bubble_sort(message, 10);

        write(client_sock, &message, 10 * sizeof(int));
    }

    if (read_size == 0) {
        puts("Client disconnected");
    }
    else if (read_size == -1) {
        perror("recv failed");
    }

    return 0;
}

// Function to sort the array
void bubble_sort(int list[], int n)
{
    int c, d, t;

    for (c = 0; c < (n - 1); c++) {
        for (d = 0; d < n - c - 1; d++) {
            if (list[d] > list[d + 1]) {

                /* Swapping */
                t = list[d];
                list[d] = list[d + 1];
                list[d + 1] = t;
            }
        }
    }
}
```

Note : Save above file as server.c **Output:**

```
 0 1 2 3 4 5 6 7 8 9
```

Last Updated : 28 Mar, 2022 1

Similar Reads

1. Handling multiple clients on server with multithreading using Socket Programming in C/C++

2. Socket Programming in C/C++: Handling multiple clients on server without multi threading
3. Comparison among Bubble Sort, Selection Sort and Insertion Sort
4. Sort an array using Bubble Sort without using loops
5. Explicitly assigning port number to client in Socket
6. Sort an Array which contain 1 to N values in O(N) using Cycle Sort
7. Sort the array using slow sort
8. Count swaps required to sort an array using Insertion Sort
9. Program to sort an array of strings using Selection Sort
10. Sort an array of pairs using Java Arrays.sort() with custom Comparator

Related Tutorials

1. Learn Data Structures with Javascript | DSA Tutorial
2. Introduction to Max-Heap – Data Structure and Algorithm Tutorials
3. Introduction to Set – Data Structure and Algorithm Tutorials
4. Introduction to Map – Data Structure and Algorithm Tutorials
5. What is Dijkstra's Algorithm? | Introduction to Dijkstra's Shortest Path Algorithm

Previous Next

Article Contributed By :

ajay0007
@ajay0007

Vote for difficulty

Current difficulty : Expert

Easy | Normal | Medium | Hard | Expert

Improved By : varshagumber28

Article Tags : C Language, DSA, Sorting

Practice Tags : Sorting

Improve Article | Report Issue

GeeksforGeeks
A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us
Careers
In Media
Contact Us
Terms and Conditions
Privacy Policy
Copyright Policy
Third-Party Copyright Notices
Advertise with us

Trending @GfG

Job Fair For Students
GfG Weekly #100
POTD: Revamped
Python Backend LIVE
Android App Development
DevOps LIVE
DSA in JavaScript

Languages

Python
Java
C++
GoLang
SQL
R Language
Android Tutorial

Data Structures

Array
String
Linked List
Stack
Queue
Tree
Graph

Algorithms

Sorting

Searching

Greedy

Dynamic Programming

Pattern Searching

Recursion

Backtracking

Web Development

HTML

CSS

JavaScript

Bootstrap

ReactJS

AngularJS

NodeJS

Data Science & ML

Data Science With Python

Data Science For Beginner

Machine Learning Tutorial

Maths For Machine Learning

Pandas Tutorial

NumPy Tutorial

NLP Tutorial

Interview Corner

Company Preparation

Preparation for SDE

Company Interview Corner

Experienced Interview

Internship Interview

Competitive Programming

Aptitude

Python

Python Tutorial

Python Programming Examples

Django Tutorial

Python Projects

Python Tkinter

OpenCV Python Tutorial

GfG School

CBSE Notes for Class 8

CBSE Notes for Class 9

CBSE Notes for Class 10

CBSE Notes for Class 11

CBSE Notes for Class 12

English Grammar

UPSC/SSC/BANKING

SSC CGL Syllabus

SBI PO Syllabus

IBPS PO Syllabus

UPSC Ethics Notes

UPSC Economics Notes

UPSC History Notes

Write & Earn

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships

Video Internship

@geeksforgeeks , Some rights reserved