Client_data

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <arpa/inet.h>
#include <unistd.h>

int main() {
    int sock = 0;
    struct sockaddr_in serv_addr;
    char buffer[1024] = {0};

    // Create socket
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if (sock < 0) {
        printf("\nSocket creation error\n");
        return -1;
    }

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(5000);

    // Convert IPv4 and IPv6 addresses from text to binary form
    if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr) <= 0) {
        printf("\nInvalid address/ Address not supported\n");
        return -1;
    }

    // Connect to server
    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) <
0) {
        printf("\nConnection Failed\n");
        return -1;
    }

    // Read date and time from server
    read(sock, buffer, 1024);
    printf("Server Date and Time: %s\n", buffer);

    close(sock);
    return 0;
}
```

Server_data

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <arpa/inet.h>
#include <unistd.h>
int main() {
    int server_fd, new_socket;
    struct sockaddr_in address;
    int addrlen = sizeof(address);

    // Create socket
    server_fd = socket(AF_INET, SOCK_STREAM, 0);
    if (server_fd == 0) {
        perror("Socket failed");
        exit(EXIT_FAILURE);
    }

    // Bind socket to port 5000
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons(5000);

    if (bind(server_fd, (struct sockaddr *)&address, sizeof(address)) <
0) {
        perror("Bind failed");
        exit(EXIT_FAILURE);
    }
    // Listen for incoming connections
    if (listen(server_fd, 3) < 0) {
        perror("Listen failed");
        exit(EXIT_FAILURE);
    }

    printf("Server listening on port 5000...\n");

    // Accept client connection
    new_socket = accept(server_fd, (struct sockaddr *)&address,
(socklen_t*)&addrlen);
    if (new_socket < 0) {
        perror("Accept failed");
        exit(EXIT_FAILURE);
    }

    // Get current date and time
    time_t now = time(NULL);
    char *date_time = ctime(&now);

    // Send date and time to client
    send(new_socket, date_time, strlen(date_time), 0);
    printf("Date and time sent to client.\n");

    close(new_socket);
    close(server_fd);
    return 0;
}
```