```cpp
#define _WINSOCK_DEPRECATED_NO_WARNINGS
#include <iostream>
#include <winsock2.h>
#include <string>
#include <cstdlib>
#include <ctime>
#include <random>
#pragma comment(lib, "Ws2_32.lib")

int main() {
    WSADATA wsaData;
    if (WSAStartup(MAKEWORD(2, 2), &wsaData) != 0) {
        std::cerr << "WSAStartup failed" << std::endl;
        return 1;
    }

    SOCKET ClientSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (ClientSocket == INVALID_SOCKET) {
        std::cerr << "Error creating socket: " << WSAGetLastError() << std::endl;
        WSACleanup();
        return 1;
    }

    sockaddr_in serverAddr;
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_port = htons(27000);
    serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");

    if (connect(ClientSocket, reinterpret_cast<sockaddr*>(&serverAddr), sizeof(serverAddr)) == SOCKET_ERROR) {
        std::cerr << "Failed to connect to server: " << WSAGetLastError() << std::endl;
        closesocket(ClientSocket);
        WSACleanup();
        return 1;
    }

    char RxBuffer[128] = {};
    if (recv(ClientSocket, RxBuffer, sizeof(RxBuffer), 0) == SOCKET_ERROR) {
        std::cerr << "Failed to receive data from server: " << WSAGetLastError() << std::endl;
        closesocket(ClientSocket);
        WSACleanup();
        return 1;
    }

    if (strcmp(RxBuffer, "Full") == 0) {
        std::cout << "Server full" << std::endl;
        closesocket(ClientSocket);
        WSACleanup();
        return 1;
    }

    std::random_device rd;
    std::uniform_int_distribution<int> dist(-30, 30);

    for (int i = 0; i < 5; ++i) { // Sending 5 pairs of coordinates
        int random_x = dist(rd);
        int random_y = dist(rd);

        std::string TxBuffer = "X=" + std::to_string(random_x) + ", Y=" + std::to_string(random_y);

        if (send(ClientSocket, TxBuffer.c_str(), TxBuffer.length(), 0) == SOCKET_ERROR) {
            std::cerr << "Failed to send data to server: " << WSAGetLastError() << std::endl;
            closesocket(ClientSocket);
            WSACleanup();
            return 1;
        }

        memset(RxBuffer, 0, sizeof(RxBuffer));
        if (recv(ClientSocket, RxBuffer, sizeof(RxBuffer), 0) == SOCKET_ERROR) {
            std::cerr << "Failed to receive data from server: " << WSAGetLastError() << std::endl;
            closesocket(ClientSocket);
            WSACleanup();
            return 1;
        }
        std::cout << "Ack received from server: " << RxBuffer << std::endl;
    }

    // Send termination signal
    if (send(ClientSocket, "[q]", sizeof("[q]"), 0) == SOCKET_ERROR) {
        std::cerr << "Failed to send termination signal to server: " << WSAGetLastError() << std::endl;
    }

    closesocket(ClientSocket);
    WSACleanup();
    return 0;
}
```