Great! Since you've confirmed with "yes," I'll proceed with the **Java programs (Q11–Q17)** by pasting them here with clean formatting and brief explanations.

🔷 Q11. Java Client-Server to return server date & time

Server.java

```
import java.io.*;

import java.net.*;

import java.util.Date;


public class Server {

    public static void main(String[] args) throws IOException {

        ServerSocket ss = new ServerSocket(5000);

        System.out.println("Server started...");

        Socket s = ss.accept();

        PrintWriter out = new PrintWriter(s.getOutputStream(), true);

        out.println("Server date and time: " + new Date());

        ss.close();

    }

}
```

Client.java

```
import java.io.*;

import java.net.*;


public class Client {

    public static void main(String[] args) throws IOException {

        Socket s = new Socket("localhost", 5000);

        BufferedReader in = new BufferedReader(new InputStreamReader(s.getInputStream()));

        System.out.println(in.readLine());

        s.close();

    }

}
```

◆ Q12. Java program to get local IP address and class

```
import java.net.*;

public class LocalIPClass {
    public static void main(String[] args) throws Exception {
        InetAddress ip = InetAddress.getLocalHost();
        System.out.println("IP Address: " + ip.getHostAddress());

        String[] parts = ip.getHostAddress().split("\\.");
        int firstOctet = Integer.parseInt(parts[0]);

        if (firstOctet >= 1 && firstOctet <= 126)
            System.out.println("Class A");
        else if (firstOctet >= 128 && firstOctet <= 191)
            System.out.println("Class B");
        else if (firstOctet >= 192 && firstOctet <= 223)
            System.out.println("Class C");
        else
            System.out.println("Other");
    }
}
```

◆ Q13. Java program: client sends string, server returns uppercase

UpperServer.java

```
import java.io.*;
import java.net.*;

public class UpperServer {
    public static void main(String[] args) throws IOException {
        ServerSocket ss = new ServerSocket(5001);
        Socket s = ss.accept();
        BufferedReader in = new BufferedReader(new InputStreamReader(s.getInputStream()));
        PrintWriter out = new PrintWriter(s.getOutputStream(), true);
        String input = in.readLine();
        out.println(input.toUpperCase());
        ss.close();
    }
}
```

UpperClient.java

```
import java.io.*;
import java.net.*;

public class UpperClient {
    public static void main(String[] args) throws IOException {
        Socket s = new Socket("localhost", 5001);
        PrintWriter out = new PrintWriter(s.getOutputStream(), true);
        out.println("hello world");
        BufferedReader in = new BufferedReader(new InputStreamReader(s.getInputStream()));
        System.out.println("Uppercase from server: " + in.readLine());
        s.close();
    }
}
```

🔷 Q14. Stop-and-Wait ARQ – Concept (Simulated Example)

This is a simulation (not real network ARQ, just logic):

```
public class StopAndWaitARQ {
    public static void main(String[] args) {
        int totalFrames = 5;
        for (int i = 0; i < totalFrames; i++) {
            System.out.println("Sending Frame: " + i);
            System.out.println("Waiting for ACK of Frame: " + i);
            System.out.println("ACK received for Frame: " + i + "\n");
        }
    }
}
```

🔷 Q15. Subnetting based on site address and host count

This requires detailed subnet calculation logic. Sample logic:

```
import java.util.Scanner;

public class SubnetCalculator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of hosts required: ");
        int hosts = sc.nextInt();
        int bits = 0;
        while (Math.pow(2, bits) - 2 < hosts) bits++;
        int subnetMask = 32 - bits;
        int totalAddresses = (int) Math.pow(2, bits);
        System.out.println("Subnet Mask: /" + subnetMask);
        System.out.println("Total IPs: " + totalAddresses);
        System.out.println("Usable Hosts: " + (totalAddresses - 2));
        sc.close();
    }
}
```

Here are the Java programs for:

✅ Q16. Echo Client and Server in Java

This program implements a basic echo server that sends back the same message it receives from the client.

- **EchoServer.java**

```
import java.io.*;

import java.net.*;


public class EchoServer {

    public static void main(String[] args) throws IOException {

        ServerSocket serverSocket = new ServerSocket(6000);

        System.out.println("Server started. Waiting for client...");


        Socket clientSocket = serverSocket.accept();

        BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));

        PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);


        String received = in.readLine();

        System.out.println("Received from client: " + received);

        out.println("Echo: " + received);


        clientSocket.close();

        serverSocket.close();

    }

}
```

- **EchoClient.java**

```
import java.io.*;

import java.net.*;


public class EchoClient {
    public static void main(String[] args) throws IOException {
        Socket socket = new Socket("localhost", 6000);
        PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
        BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));

        out.println("Hello from Client");
        String response = in.readLine();
        System.out.println("Server replied: " + response);

        socket.close();
    }
}
```

✅ Q17. Java Program to Generate CRC of a Binary String

This program computes the **Cyclic Redundancy Check (CRC)** for a given binary string using a divisor (generator polynomial).

🔹 CRCGenerator.java

```
import java.util.Scanner;

public class CRCGenerator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter binary data: ");
        String data = sc.nextLine();
        System.out.print("Enter generator polynomial: ");
        String divisor = sc.nextLine();

        String dividend = data + "0".repeat(divisor.length() - 1);
        String remainder = divide(dividend, divisor);
        String crc = remainder;

        System.out.println("CRC: " + crc);
        System.out.println("Transmitted Data: " + data + crc);
        sc.close();
    }

    private static String divide(String dividend, String divisor) {
        int pick = divisor.length();
        String tmp = dividend.substring(0, pick);

        while (pick < dividend.length()) {
            if (tmp.charAt(0) == '1')
                tmp = xor(divisor, tmp) + dividend.charAt(pick);
            else
                tmp = xor("0".repeat(pick), tmp) + dividend.charAt(pick);
            pick += 1;
        }

        if (tmp.charAt(0) == '1')
            tmp = xor(divisor, tmp);
        else
            tmp = xor("0".repeat(pick), tmp);

        return tmp;
    }

    private static String xor(String a, String b) {
        StringBuilder result = new StringBuilder();
        for (int i = 1; i < a.length(); i++)
```

```
            result.append(a.charAt(i) == b.charAt(i) ? '0' : '1');
        return result.toString();
    }
}
```

Would you like these two programs added to the PDF as well?