1. Develop a java program to simulate Bit Stuffing.

PROGRAM:

```
import java.io.*;
public class BitStuffing {
  public static void main(String[] args)throws IOException {
  int i=0,x,q=0;
  InputStreamReader isr = new InputStreamReader(System.in);
  BufferedReader br = new BufferedReader(isr);
  String sy="01111110",sx;
  StringBuilder sby = new StringBuilder(sy);
  System.out.print("ENTER THE DATA: ");
   sx = br.readLine();
   StringBuilder sbx = new StringBuilder(sx);
   x = sx.length();
    while(i+5<=x){
        String s1 = sx.substring(i, i+5);
        if(check(s1)){
           sbx.insert(i+5,0);
            i=i+6;
        }
        else i++;
     }
    System.out.print("BIT STUFFING: ");
    System.out.println(sbx);
    System.out.print("FINAL OUTPUT: ");
    System.out.println(sby+"  "+sbx+"  "+sby);
    System.out.println(sby.append(sbx.append(sby)));
 }
 private static boolean check(String s){
    String s1 = "11111";
    if(s.equals(s1))
       return true;
    else return false;
  }
}
```

---

2. Design and implement a simple reliable internet protocol for the following scenario: A sender has 10 packets to transfer to its destination receiver. Sender will be sending a packet, and waits for the acknowledgement. Once after receiving the acknowledgement the sender will send the next packet to the intended receiver. If the sender didn't receive any acknowledgment, it will be waiting for a timeout and starts retransmitting the same frame. (Stop and Wait)

PROGRAM:

SENDER:

```
import java.io.*;
import java.net.*;
import java.util.Scanner;
class stopwaitsender
{
public static void main(String args[]) throws Exception
```

```
{
stopwaitsender sws = new stopwaitsender();
sws.run();
}
public void run() throws Exception
{
Scanner sc=new Scanner(System.in);
System.out.println("Enter no of frames to be sent:");
int n=sc.nextInt();
Socket myskt=new Socket("localhost",9999);
PrintStream myps=new PrintStream(myskt.getOutputStream());
for(int i=0;i<=n;)
{
if(i==n)
{
myps.println("exit");
break;
}
System.out.println("Frame no "+i+" is sent");
myps.println(i);
BufferedReader bf=new BufferedReader(new
InputStreamReader(myskt.getInputStream()));
String ack=bf.readLine();
if(ack!=null)
{
System.out.println("Acknowledgement was Received from receiver");
i++;
Thread.sleep(4000);
}
else
{
myps.println(i);
}}}}
```

RECEIVER:

```
import java.io.*;
import java.net.*;
class stopwaitreceiver
{
public static void main(String args[])throws Exception
{
stopwaitreceiver swr = new stopwaitreceiver();
swr.run();
}
public void run() throws Exception
{
String temp="any message",str="exit";
ServerSocket myss=new ServerSocket(9999);
Socket ss_accept=myss.accept();
BufferedReader ss_bf=new BufferedReader(new
InputStreamReader(ss_accept.getInputStream()));
PrintStream myps=new PrintStream(ss_accept.getOutputStream());
while(temp.compareTo(str)!=0)
{
```

```
Thread.sleep(1000);
temp=ss_bf.readLine();
if(temp.compareTo(str)==0)
{ break;}
System.out.println("Frame "+temp+" was received");
Thread.sleep(500);
myps.println("Received");
}
System.out.println("ALL FRAMES WERE RECEIVED SUCCESSFULLY");
}}
```

3. Implement the following scenario using java. (Go-Back N)
A sender has 10 packets to transfer to its destination receiver. Sender will be sending a packet, and waits for the acknowledgement. Once after receiving the acknowledgement the sender will send the next packet to the intended receiver. If the sender didn't receive any acknowledgment, it will be waiting for a timeout and started retransmitting all the frames. (hint: The Window Size is=10)

PROGRAM:

```
import java.io.*;
public class GoBackN {
public static void main(String args[]) throws IOException
{
BufferedReaderbr = new BufferedReader(new InputStreamReader(System.in));
System.out.println("Please enter the Window Size: ");
int window = Integer.parseInt(br.readLine());
boolean loop = true;
int sent = 0;
while(loop)
{
for(inti = 0; i< window; i++)
{
System.out.println("Frame " + sent + " has been transmitted.");
sent++;
if(sent == window)
break;
}
System.out.println("Please enter the last Acknowledgement received.");
intack = Integer.parseInt(br.readLine());
if(ack == window)
loop = false;
else
sent = ack;
}}}
```

4. Design and implement a simple reliable internet protocol for the following scenario:
A sender has 10 packets to transfer to its destination receiver. Sender will be sending a packet, and waits for the acknowledgement. Once after receiving the acknowledgement the sender will send the next packet to the intended receiver. If the sender didn't receive any acknowledgment, it will be waiting for a timeout and started retransmitting the lost frame. (Hint: The Window Size is=10). (Sliding Window- Selective Repeat)

```
import java.io.*;
public class SelectiveRepeatAlg
{
int intToPkt,intWinSize,intPktNo,intPCount;
public SelectiveRepeatAlg()throws Exception
{
BufferedReader br=new BufferedReader(new InputStreamReader(new
DataInputStream(System.in)));
System.out.println("Enter the window size");
intWinSize=Integer.parseInt(br.readLine());
System.out.println("enter Total no. of packets");
intToPkt=Integer.parseInt(br.readLine());
System.out.println("Do you want to kill any Packets:(y/n)");
String strAns=br.readLine();
if(strAns.equals("y"))
{
System.out.println("Enter NO.of Packets 2 kill");
intPktNo=Integer.parseInt(br.readLine());
intPCount=1;
}}
public void sRepeat(int pn)throws Exception
{
System.out.println("------------------------------");
System.out.println("Selective Repeats of packets"+pn);
System.out.println("------------------------------");
Thread.sleep(1000);
System.out.println("Packet "+(pn)+("send==>"));
Thread.sleep(1000);
System.out.println("Packet"+(pn)+("received==>"));
Thread.sleep(2000);
System.out.println("ACK"+(pn)+" Received");
}
public int calculate(int cos,int pno)
{
int n;
if(cos>pno)
n=intWinSize-(intPktNo%intWinSize);
else
n=intWinSize-intPktNo;
return n;
}
public void trans()throws Exception
{
int intC,intCount=0;
while(intPCount<intToPkt)
{
if(intCount==intWinSize)
{
System.out.println("Next Session");
intCount=0;
}
if(intPktNo==intPCount)
{
System.out.println("Packet"+intPktNo+"discard---->");
Thread.sleep(2000);
```

```
intPCount++;
intC=calculate(intWinSize,intPktNo);
System.out.println("Remaining Packet to be sent in the fashion"+intC);
for(int j=0;j<=intC;j++)
{
Thread.sleep(1000);
System.out.println("Packet "+intPCount+"Sent==>");
Thread.sleep(1000);
System.out.println("\t\tPacket"+intPCount+" Received");
Thread.sleep(2000);
System.out.println("ACK"+intPCount+"Received");
intPCount++;
intCount=0;
}
sRepeat(intPktNo);
Thread.sleep(2000);
System.out.println("next Session");
intCount=0;
}
else{
Thread.sleep(1000);
System.out.println("Packet"+intPCount+"send==>");
Thread.sleep(1000);
System.out.println("\t\t packet"+intPCount+"Received");
intPCount++;
}
intPCount++;
}}
public static void main(String agr[])throws Exception
{
SelectiveRepeatAlg obSRA=new SelectiveRepeatAlg();
obSRA.trans();
}}
```

5. Develop a java program to Distance Vector Routing.

PROGRAM:

```
class GFG
{
static void BellmanFord(int graph[ ][ ], int V, int E, int src)
{
// Initialize distance of all vertices as infinite.
int []dis = new int[V];
for (int i = 0; i &lt; V; i++)
dis[i] = Integer.MAX_VALUE;
// initialize distance of source as 0
dis[src] = 0;
for (int i = 0; i &lt; V - 1; i++)
{
for (int j = 0; j &lt; E; j++)
{
if (dis[graph[j][0]] != Integer.MAX_VALUE &amp;&amp; dis[graph[j][0]] + graph[j][2] &lt;
dis[graph[j][1]])
```

```
dis[graph[j][1]] =
dis[graph[j][0]] + graph[j][2];
}
}
//check for negative-weight cycles.
for (int i = 0; i &lt; E; i++)
{
int x = graph[i][0];
int y = graph[i][1];
int weight = graph[i][2];
if (dis[x] != Integer.MAX_VALUE &amp;amp;&amp;amp;
dis[x] + weight &lt; dis[y])
System.out.println(&quot;Graph contains negative&quot; +&quot; weight cycle&quot;);
}
System.out.println(&quot;Vertex Distance from Source&quot;);
for (int i = 0; i &lt; V; i++)
System.out.println(i + &quot;\t\t&quot; + dis[i]);
}

// Main program
public static void main(String[] args)
{
int V = 6; // Number of vertices in graph
int E = 8; // Number of edges in graph
/* Every edge has three values (u, v, w) where the edge is from vertex u to v. And weight of
the edge is w.
int graph[][] = { { 0, 1, 10 }, { 0, 5, 8 },
{ 1, 3, 2 }, {2,1,1}, { 3, 2, -2 },
{ 4, 1, -4 }, { 4, 3, -1 }, {5, 4, 1} };
BellmanFord(graph, V, E, 0);
}
}
```

---

6. Develop a java program to simulate Link State routing.

PROGRAM:

```
import java.util.*;
import java.lang.*;
import java.io.*;
class ShortestPath
{
static final int V = 4;
int minDistance(int dist[], Boolean sptSet[])
{
int min = Integer.MAX_VALUE, min_index = -1;
for (int v = 0; v &lt; V; v++)
if (sptSet[v] == false &amp;amp;&amp;amp; dist[v] &lt;= min)
{
min = dist[v];
min_index = v;
}
return min_index;
}
void printSolution(int dist[], int n)
```

```
{
System.out.println(&quot;Vertex Distance from Source&quot;);
for (int i = 0; i &lt; V; i++)
System.out.println(i + &quot; tt &quot; + dist[i]);
}

void dijkstra(int graph[][], int src)
{
int dist[] = new int[V];
Boolean sptSet[] = new Boolean[V];
for (int i = 0; i &lt; V; i++)
{
dist[i] = Integer.MAX_VALUE;
sptSet[i] = false;
}
dist[src] = 0;
for (int count = 0; count &lt; V - 1; count++)
{
int u = minDistance(dist, sptSet);
sptSet[u] = true;
for (int v = 0; v &lt; V; v++)
if (!sptSet[v] &amp;&amp; graph[u][v] != 0 &amp;&amp;
dist[u] != Integer.MAX_VALUE &amp;&amp;
dist[u] + graph[u][v] &lt; dist[v])
dist[v] = dist[u] + graph[u][v];
}
printSolution(dist, V);
}

// Driver method
public static void main(String[] args)
{
/* Let us create the example graph discussed above */
int graph[][] = new int[][] {
{0,5,10,0},
{5,0,3,11},
{10,3,0,2},
{0,11,2,0},
};
ShortestPath t = new ShortestPath();
t.dijkstra(graph, 0);
}
}
```

7. Write a java socket program to implement echo client and echo server using TCP.

PROGRAM:

ECHO CLIENT:

```
import java.util.*;
import java.lang.*;
import java.io.*;
class ShortestPath
{
```

```
static final int V = 4;
int minDistance(int dist[], Boolean sptSet[])
{
int min = Integer.MAX_VALUE, min_index = -1;
for (int v = 0; v &lt; V; v++)
if (sptSet[v] == false &amp;&amp; dist[v] &lt;= min)
{
min = dist[v];
min_index = v;
}
return min_index;
}
void printSolution(int dist[], int n)
{
System.out.println(&quot;Vertex Distance from Source&quot;);
for (int i = 0; i &lt; V; i++)
System.out.println(i + &quot; tt &quot; + dist[i]);
}

void dijkstra(int graph[][], int src)
{
int dist[] = new int[V];
Boolean sptSet[] = new Boolean[V];
for (int i = 0; i &lt; V; i++)
{
dist[i] = Integer.MAX_VALUE;
sptSet[i] = false;
}
dist[src] = 0;
for (int count = 0; count &lt; V - 1; count++)
{
int u = minDistance(dist, sptSet);
sptSet[u] = true;
for (int v = 0; v &lt; V; v++)
if (!sptSet[v] &amp;&amp; graph[u][v] != 0 &amp;&amp;
dist[u] != Integer.MAX_VALUE &amp;&amp;
dist[u] + graph[u][v] &lt; dist[v])
dist[v] = dist[u] + graph[u][v];
}
printSolution(dist, V);
}

// Driver method
public static void main(String[] args)
{
/* Let us create the example graph discussed above */
int graph[][] = new int[][] {
{0,5,10,0},
{5,0,3,11},
{10,3,0,2},
{0,11,2,0},
};
ShortestPath t = new ShortestPath();
t.dijkstra(graph, 0);
}
```

```
}
```

ECHO SERVER:

```
import java.io.*;
import java.net.*;
class echos {
   public static void main(String args[]) throws Exception
   {
      String echoin;
      ServerSocket svrsoc;
      Socket soc;
      BufferedReader br;
      try
      {
         svrsoc = new ServerSocket(2000);
         soc = svrsoc.accept();
         br = new BufferedReader (new InputStreamReader(soc.getInputStream()));
         PrintStream ps = new PrintStream(soc.getOutputStream());
         System.out.println("Connected for echo:");
         while((echoin=br.readLine())!=null)
         {
            if(echoin.equals("end"))
            {
               System.out.println("Client disconnected");
               br.close();
               break;
            }
            else
               ps.println(echoin);
         }
      }
      catch(UnknownHostException e)
      {
         System.out.println(e.toString());
      }
      catch(IOException ioe)
      {
         System.out.println(ioe);
      }
   }
}
```

8. Write a socket program to contact a given DNS server to resolve a given host name using UDP

UDPDNSSERVER:

```
import java.io.*;
import java.net.*;
public class udpdnsserver
{
private static int indexOf(String[] array, String str)
{
str = str.trim();
for (int i=0; i < array.length; i++)
```

```
{
if (array[i].equals(str))
return i;
}
return -1;
}
public static void main(String arg[])throws IOException
{
String[] hosts = {"yahoo.com", "gmail.com","cricinfo.com", "facebook.com"};
String[] ip = {"68.180.206.184", "209.85.148.19","80.168.92.140","69.63.189.16"};
System.out.println("Press Ctrl + C to Quit");
while (true)
{
DatagramSocket serversocket=new DatagramSocket(1362);
byte[] senddata = new byte[1021];
byte[] receivedata = new byte[1021];
DatagramPacket recvpack = new
DatagramPacket(receivedata, receivedata.length);
serversocket.receive(recvpack);
String sen = new String(recvpack.getData());
InetAddress ipaddress = recvpack.getAddress();
int port = recvpack.getPort();
String capsent;
System.out.println("Request for host " + sen);
if(indexOf (hosts, sen) != -1)
capsent = ip[indexOf (hosts, sen)];
else
capsent = "Host Not Found";
senddata = capsent.getBytes();
DatagramPacket pack = new DatagramPacket(senddata,
senddata.length,ipaddress,port);
serversocket.send(pack);
serversocket.close();
}}}
```

• UDPDNSCLIENT:

```
import java.io.*;
import java.net.*;
public class udpdnsclient
{
public static void main(String args[])throws IOException
{
BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
DatagramSocket clientsocket = new DatagramSocket();

InetAddress ipaddress;
if (args.length == 0)
ipaddress = InetAddress.getLocalHost();
else
ipaddress = InetAddress.getByName(args[0]);
byte[] senddata = new byte[1024];
```

```
byte[] receivedata = new byte[1024];
int portaddr = 1362;
System.out.print("Enter the hostname : ");
String sentence = br.readLine();
senddata = sentence.getBytes();
DatagramPacket pack = new DatagramPacket(senddata,senddata.length,
ipaddress,portaddr);

clientsocket.send(pack);
DatagramPacket recvpack =new

DatagramPacket(receivedata,receivedata.length);

clientsocket.receive(recvpack);
String modified = new String(recvpack.getData());
System.out.println("IP Address: " + modified);
clientsocket.close();
}}
```

9. Create a LAN using CISCO packet Tracer. Configure the created LAN using PING and IPCONFIG commands

10. Implement a Traffic generator scenario for a FTP application using CISCO Packet Tracer.