**STOP-WAIT**

1. **START PROGRAM**
2. **GENERATE A RANDOM NUMBER THAT GIVES YOU THE NUMBER OF FRAMES TO BE TRANSIMITTED**
3. **TRANSMIT FIRST FRAME**
4. **RECIVER THE ACKNOLEDEGMENT FOR THE FIRST FRAME**
5. **TRANSMIT THE SECOND FRAME**
6. **FIND REMAING FRAMES TO BE TRANSMITTED**
7. **IF AN ACKNOLGMENT IS NOT RECIVED FOR A PARTICULAR FRAME,RESENT THE FRAME ALONE AGAIN**
8. **REPEAT STEP 5-7 TILL NO MORE FRAME REMAIN AND FRAM TO BE SENT BECOME ZERO**
9. **STOP THE PROGRAM**

**TCP-CLIENT**

1. **CREATE SOCKET**
2. **CONNECT SOCKET TO SERVER**
3. **READ STRING TO BE REVERSED USING THE STANDARD I/P**

**AND SENT THE STRING FROM SOCKET TO SERVER**

1. **READ THE REVERSED STRING FROM THE SERVER AND DISPLAY IT ON STANDARD O/P**
2. **CLOSE SOCKET**

**TCP-SERVER**

1. **CREATE LISTENING SOCKET**
2. **BIND IP AND PORT NO TO SOCKET**
3. **LISTEN FOR THE INCOMING REQUEST USING LISTNING SOCKET**
4. **ACCEPT THE REQUEST**
5. **CONNECTION SOCKET IS CREATED WHEN ACCEPT RETURNS**
6. **READ SWTRING USING CONNECTION SOCKET FROM CLIENT**
7. **REVERSE THE STRING**
8. **‘SENT STRING TO CLIENT USING THE CONNECTION SOCKET**
9. **CLOSE CONNECTION SOCKET**
10. **CLOSE LISTNING SOCKET**

**DVR**

1. **INPUT THE NUMBER OF NODES**
2. **GET THE DISTANCE BETWEEN NODES**
3. **GET MAX VALUE IN MATRIX FORM**
4. **COMPARE DIST BETWEEN NODES**
5. **LET THE ADDITION OF TWO DISATNCE IS LESS THAN SPECIFIED THEN PRINT THE NEW DISTANCE**

**LEAKEY BUCKET**

1. **START**
2. **DEFINE NECESSARY VARIABLES**
3. **PROMPT USER TO ENTER BUCKT SIZE AND NO OF I/P**
4. **ENTER PROCESS OUTGOING RATE**
5. **INITALIZE THE BUCKET STORAGE TO 0**
6. **DISPLAY THE INITAIL BUCKET STATUS**
7. **ENTER THE LOOP THAT CONTINUE UNTIL N BECOMNE 0**
8. **WHITHIN LOOP:**

* **PROMPT USER TO ENTER THE INCOMING BUCKET SIZE**
* **CHECK IF THE INCOMING PACKET SIZE IS LESSTHAN OR EQUAL TO VARIABLE SPACE IN BUCKET**
* **IF TRUE ,UPDATE BUCKET STORAGE AND DISPLAY NEW BUCKET STORAGE**
* **IF FALSE ,RELOP THE PARTS THAT EXCEED THE BUCKET CAPACITY ,DISPLAY THE DROPPED PACKETS AND UPDATE THE BUCKET STORAGE TO ITS FULL CAPACITY**
* **REDUCE THE BUCKET STORAGE BY THE OUTGOING PACKETRATE AND DISPLAY THE UPDATED PACKET STATUS**
* **DECREMENT N BY 1**

1. **AFTER THE LOOP ENDS CHECK IF N IS 0 AND THE BUCKET STORAGE IS NOT EMPTY**
2. **IF TRUE EMPTY THE BUCKET AND DISPLAY THE FINAL BUCKKET STATUS**
3. **END**

**GO BACK-SERVER**

1. **START THE PROGRAM**
2. **DECLARE NECESSARY VARIABLES**
3. **INITAILAZE VARIABLES FOR SOCKET DESCRIPTORS,BUFFER SIZE,WINDOW PARAMETERS,AND SOCKET ADDRESS**
4. **GENERTAE TCP SOCKET USING SOCKET() FUNCTION**
5. **BIND THE SOCKET**
6. **START LISTING FOR INCOMING CONNECTIONS FROM CLIENT USING LISTEN()FUNCTION**
7. **ACCEPT THE INCOMING CONNECTIONS FROM CLIENT USING ACCEPT () FUNCTION**
8. **RECIEVE REQUEST FROM CLINT USING RECIEVE () FUCTION**
9. **SET THE SOCKET TO NON BLOCKING MODE USING THE FINT()FUNCTION WITH THE F\_SETLE FLAG AND O\_NON BLOCK OPTION**
10. **LOOP THROUGH THW INDOW SIZE AND SEND PACKETS TO THE CLIENT USING THE SEND()FUNCTION**
11. **CONVERT THE PACKET NUMBER TO A STRING USING SPRINTF() FUNCTION**
12. **RECIEVE ACKLODGMENTS(ACKS)O RETRANSIMISSION REQUESTS FROM THE CLIENT USINGTHE REW() FUNCTION**
13. **IF RETRANSMISSION REUEST IS RECIEVED RESEND THE CORRESPONDINGPACKET TO CLIENT**
14. **IF ACK IS RECIEVED UPDATE WINDOW BOUNDARIES**
15. **CLOSE SOCKETS**

**CLIENT**

1. **Start**
2. **Declare necessary header files**
3. **Initialize variables**
4. **Set the server address structure with appropriate ip address and port no**
5. **Connect to server using connect() function**
6. **Sent request to server using Sennt () function**
7. **Receive the packets from server in x loop using the receive () function**
8. **Convert the received data to integer representing current packet numbr**
9. **Check if received packet is expected packet number**
10. **If the packet is not the expected one and is first time receiving an incorrect packet .Simulate correcption by sending a retransmission**
11. **Otherwise sent acknowledgment for the received packet after a certain number of successful reception**
12. **After receiving expected packet pr after sucesscfull receptions sent ack to server**
13. **Continue receiving packets untilthe last expected packis received**
14. **Close socket conection using socket()functiom**