Sender_Reciever:

```c
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

char res[100];

void sender()
{
    int n,i,len;
    char frame[100],l[100];
    printf("Enter the number of frames\n");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter the frame %d\n",i+1);
        scanf("%s",&frame);
        len=strlen(frame);
        printf("Number of bytes in the frame %d = %d\n",i+1,len);
        sprintf(l,"%d",len);
        strcat(l,frame);
        strcat(res,l);

    }
}

void reciever()
{
    int len,i,j;
    printf("Received frame \n");
        for(i=0;i<strlen(res);i++)
        {
        len=res[i]-'0';
        for(j=i+1;j<=(i+len);j++)
        printf("%c",res[j]);
        i=i+len;
        printf("\n");
        }
}

int main()
{
sender();
reciever();
return 0;
}
```

Bitstuffing :

```c
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

void receiver(int *frame, int l2)
{
    int i, j, counter, l3;
    int msg[100];
    l3=l2-8;
    j=0;
    for(i=8;i<l3;i++)
    {
        if(frame[i]==0)
        {
            if(counter==5)
            {
                msg[j]=frame[i];
                j++;
                counter=0;
            }
            else
            {
                 msg[j]=frame[i];
                 j++;
                 counter++;
            }
        }
        else
        {
            msg[j]=frame[i];
            j++;
            counter++;
        }
    }

    printf("Received message is: \n");
    for(i=0;i<j;i++)
        printf("%d",msg[i]);
    printf("\n");
}

void sender()
{
    int data[100], frame[100], framelen=0, n, i, j=8;
    int count, zeroadded=0, zero;
    printf("Enter the number of bits\n");
    scanf("%d",&n);
    printf("Enter data for bits\n");
    for(i=0;i<n;i++)
         scanf("%d",&data[i]);
    frame[0]=0;
    frame[1]=1;
    frame[2]=1;
    frame[3]=1;
    frame[4]=1;
    frame[5]=1;
    frame[6]=1;
    frame[7]=0;
    for(i=0;i<n;i++)
    {
        if(data[i]==0)
        {
            frame[j]=data[i];
            j++;
            count=0;
            zero=1;
        }
        else
        {
            if((count==5)&&(zero==1))
            {
                frame[j]=0;
                j++;
                zeroadded++;
                frame[j]=data[i];
                j++;
                count=0;
            }
            else
            {
                frame[j]=data[i];
                j++;
                count++;
            }
        }
     }

    frame[j++]=0;
    frame[j++]=1;
    frame[j++]=1;
    frame[j++]=1;
    frame[j++]=1;
    frame[j++]=1;
    frame[j++]=1;
    frame[j++]=0;
    framelen=n+16+zeroadded;
    printf("length of frame sent %d \n",framelen);
    printf("Frame sent: ");
    for(i=0;i<framelen;i++)
        printf("%d",frame[i]);
    printf("\n");
    receiver(frame,framelen);
}

void main()
{
    sender();
}
```

CRC:

```
#include <stdio.h>
#include <string.h>

int main()
{
    char rem[50], code[100], gen[50], temp[100];
    int codelen, genlen, i,pos,e;

    printf("Enter the code :");
    scanf("%s", &code);

    printf("Enter the generator:");
    scanf("%s", &gen);

    codelen = strlen(code);
    genlen = strlen(gen);

    code[codelen] = '\0';
    rem[genlen] = '\0';
    // printf("codelen=%d\n",code[codelen]);
    // printf("genlen=%d\n",gen[codelen]);

    // Append 0 at the end
    strcpy(temp, code);
    for (i = 0; i < genlen-1; i++)
    {
        temp[codelen + i] = '0';
    }

    // XOR operation
    for (int j = 0; j < codelen; j++)
    {
        // When you perform XOR operation first position should be 1 itself as
per algorithm
        if (temp[j] != '0')
        {
            for (int k = 0; k < genlen; k++)
            {
               rem[k] = temp[j+k] = (temp[j+k] == gen[k]) ? '0' : '1';
            }
        }
        // printf("%s\n",temp);
        // printf("%s\n",rem);
    }

    // Reduce remainder
    for (int i = 0; i < genlen; i++)
    {
        rem[i] = rem[i+1];
    }

    printf("Message recieved at reciever site: %s",strcat(code,rem));

return 0;
}
```

<u>Distance Vector :</u>

```
#include<stdio.h>

struct node
{
    int dist[20];
    int from[20];
}rt[10];

int main()
{
    int dmat[20][20];
    int n, i, j, k, count=1;

    printf ("\nEnter the number of nodes :\n");
    scanf ("%d", &n);

    printf ("\nEnter the cost matrix :\n");
```

```c
    for (i=1; i<=n; i++)
        for (j=1; j<=n; j++)
        {
            scanf ("%d", &dmat[i][j]);
            dmat [i][i] = 0;
            rt[i].dist[j] = dmat[i][j];
            rt[i].from[j] = j;
        }

    do
    {
        for (i=1; i<=n; i++)
        for (j=1; j<=n; j++)
        for (k=1; k<=n; k++)
        if (rt[i].dist[j] > dmat[i][k] + rt[k].dist[j])
        {
            rt[i].dist[j] = rt[i].dist[k] + rt[k].dist[j];
            rt[i].from[j] = k;
        }
        count++;
    }
    while (count < n);

    for (i=1; i<=n; i++)
    {
        printf ("\nDistance Table for router %c is \n", i+64);
        for (j=1; j<=n; j++)
        printf ("\tNode %d Via %d, Distance : %d\n", j, rt[i].from[j],
rt[i].dist[j]);
    }
    return 0;
}
```

Leaky Bucket :

```c
#include<stdio.h>
#include<stdlib.h>

int main()
{
    int i,j,qs,ns,t,count,size,a,choice,p[10],cap,rate,delay,flag=1,t1,t2;

    printf("enter the queue size:");
    scanf("%d",&size);
    count=size;

    printf("enter leaky bucket capacity:");
    scanf("%d",&cap);
    qs=cap;

    printf("enter the size of the packets in the queue:");
    for(i=0;i<size;i++)
    {
        scanf("%d",&a);
        if(a>cap)
        {
            i--;
            printf("packets cannot be entered");
        }
        else
            p[i]=a;
    }

    printf("enter the output rate:");
    scanf("%d",&rate);

    delay=cap/rate;
    printf("delay=%d\n",delay);

    while(flag)
    {
        qs=cap;

        while(qs>=p[0]&&count>0)
        {
            printf("\npacket of size %d is put into the bucket\n",p[0]);
            qs=qs-p[0];
            printf("\navailable space %d\n",qs);

            count--;

            for(i=0;i<count;i++)
                p[i]=p[i+1];
        }

        t=delay;
        long int t1=(long)time(NULL);
        long int t2=(long)time(NULL);

        while((t2-t1)<delay)
        {
            t2=(long)time(NULL);
            if((delay-t)==(t2-t1))
            {
                printf("\ntransmitting packets in the leaky bucket:%d
seconds\n",t);
                t--;
            }
        }

        printf("\npackets in the queue:\n");
        for(i=0;i<count;i++)
            printf("%d\t",p[i]);

        if(count==0)
        flag=0;
        }
}
```

TCP :

server.c :

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#define PORT_ID 8000


int main()
{
    char buf[300];
    int fd1,fd2,size,n;
    struct sockaddr_in s;
    system("clear");
    printf("Server is getting ready\n");

    s.sin_family=AF_INET;
    s.sin_port=htons(PORT_ID);
    s.sin_addr.s_addr=inet_addr("127.0.0.1");
    fd1=socket(AF_INET,SOCK_STREAM,0);

    if((bind(fd1,(struct sockaddr *)&s,sizeof(struct sockaddr)))==-1)
        printf("Error in socket binding\n");
    if((listen(fd1,5))==-1)
    printf("Error in listening\n");
    printf("Waiting for client request\n");
    size=sizeof(struct sockaddr);

    fd2=accept(fd1,(struct sockaddr *)&s,&size);
    size=recv(fd2,buf,sizeof(buf),0);
    buf[size]='\0';
    printf("Filename received is %s\n",buf);
    if((fd1=open(buf,O_RDONLY))!=-1)
    {
        while((n=read(fd1,buf,sizeof(buf)))>0)
            send(fd2,buf,n,0);

    }
    else
        send(fd2,"File not found",20,0);

        close(fd1);
        close(fd2);
        printf("Server terminated");
        return 0;


}
```

Client.c:

```
#include<stdio.h>
#include<fcntl.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#define PORT_ID 8000
int main()
{
    char buf[30000];
    int fd1,n;
    struct sockaddr_in s;
    system("clear");
    printf("Enter the filename to be sent to the server\n");
    scanf("%s",buf);

    s.sin_family=AF_INET;
    s.sin_port=htons(PORT_ID);
    s.sin_addr.s_addr=inet_addr("127.0.0.1");

    fd1=socket(AF_INET,SOCK_STREAM,0);

    if((connect(fd1,(struct socketaddr *)&s,sizeof(struct sockaddr)))==-1)
         printf("Error in socket binding\n");

    send(fd1,buf,strlen(buf),0);

    printf("****Contents of the requested file is **** \n");
    while((n=recv(fd1,buf,sizeof(buf),0))>0)
    {
        buf[n]='\0';
        printf("%s",buf);
    }
    printf("\n");
    close(fd1);
    return 0;

}
```

UDP:

server.c:

```
#include <stdio.h>
#include <stdlib.h>
#include<fcntl.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
```

```c
#include<string.h>

#define PORT_ID 8000

void clearBuf(char* b){
    for (int i = 0; i < 300; i++)
        b[i] = '\0';
}

int main(){
    char buf[300];
    int fd1,n,size;
    FILE *fp;
    struct sockaddr_in s1, s2;
    int s_length = sizeof(s2);
    system("clear");
    printf("Server is getting ready.... \n");

    s1.sin_family = AF_INET;
    s1.sin_port = htons(PORT_ID);
    s1.sin_addr.s_addr = inet_addr("127.0.0.1");


    fd1 = socket(AF_INET, SOCK_DGRAM, 0);

    if((bind(fd1,(struct sockaddr *)&s1, sizeof(struct sockaddr)))==-1)
        printf("Error in socket binding!!!!\n");

    printf("Waiting for client request....\n");

    size = recvfrom(fd1,buf,sizeof(buf), 0,(struct sockaddr *)&s2, &s_length);
    buf[size] = '\0';

    fp = fopen(buf, "r");
    printf("File Name recieved is : %s\n",buf);

    if (fp == NULL){
        sendto(fd1,"File not found .... ", 20, 0,(struct sockaddr *)&s2,
s_length);
    }
    else{
        printf("\nFile Successfully opened!\n");
        clearBuf(buf);
        while(fscanf(fp, "%s", buf)!=EOF){
            sendto(fd1,buf,sizeof(buf),0,(struct sockaddr *)&s2, s_length);
            clearBuf(buf);
        }
        sendto(fd1,NULL,0,0,(struct sockaddr *)&s2, s_length);
        fclose(fp);
    }

    close(fd1);
    printf("Server terminated....\n");
    return 0;
}
```

client.c:

```
#include<stdio.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>

#define PORT_ID 8000

void clearBuf(char* b){
    for (int i = 0; i < 300; i++)
        b[i] = '\0';
}

int main(){
    char buf[30000];
    int fd1,n;
    struct sockaddr_in s;
    int s_length = sizeof(s);
    system("clear");
    printf("Enter the filename to be sent to the server \n");
    scanf("%s",buf);

    s.sin_family = AF_INET;
    s.sin_port = htons(PORT_ID);
    s.sin_addr.s_addr = inet_addr("127.0.0.1");


    fd1 = socket(AF_INET, SOCK_DGRAM, 0);

    if((connect(fd1,(struct sockaddr *)&s, sizeof(struct sockaddr)))==-1)
        printf("Error in socket connecting!!!!\n");

    sendto(fd1, buf, strlen(buf),0,(struct sockaddr *)&s,sizeof(s));
    clearBuf(buf);
    printf("**** contents of the requested file is ****\n");
    while(recvfrom(fd1,buf,sizeof(buf),0,(struct sockaddr *)&s,&s_length)){
        puts(buf);
        clearBuf(buf);
    }

    printf("\n");
    close(fd1);
    printf("Client terminated....\n");
    return 0;
}
```