DCN program

1. Cyclic Redundancy Check (CRC)

```
#include<iostream.h>
#include<stdlib.h>
#include<conio.h>

void main()
{
   int i,j,n,g,a,arr[20],gen[20],b[20],q[20],s;
   clrscr();
   cout<<" Cyclic Redundancy Check "<<endl;
   cout<<"——————————-"<<endl;
   cout<<"\nTransmitter side: "<<endl;
   cout<<"Enter the no.of data bits :- ";
   cin>>n;
   cout<<"\nEnter the data :- ";
   for(i=0;i<n;i++)
      cin>>arr[i];
   cout<<"\nEnter the size of generator :- ";
   cin>>g;
   cout<<"\nEnter the generator :- ";
   for(j=0;j<g;j++) {
      cin>>gen[j];
   }
   cout<<"\nThe Generator Matrix is: ";
   for(j=0;j<g;j++) {
      cout<<gen[j];
   }
   a=n+(g-1);
   cout<<"\nThe Appended Matrix is :- ";
   for(i=0;i<a;++i)
      cout<<arr[i];

   for(i=0;i<n;++i) {
      if(arr[i]==0) {
         for(j=i;j<g+i;++j)
            arr[j]=arr[j]^0;
      }
      else {
         arr[i]=arr[i]^gen[0];
         arr[i+1]=arr[i+1]^gen[1];
```

```
            arr[i+2]=arr[i+2]^gen[2];
            arr[i+3]=arr[i+3]^gen[3];
        }
    }
    cout<<"\n\nThe CRC is := ";
    for(i=n;i<a;++i)
        cout<<arr[i];

    s=n+a;
    for(i=n;i<s;i++)
        q[i]=arr[i];

    cout<<"\n";
    cout<<"Final Data to be transmitted is :- ";
    for(i=0;i<a;i++)
        cout<<q[i];
    getch();
}
```

2. Vertical Redundancy Check (VRC)

```
#include<iostream.h>
#include<conio.h>

void main()
{
    clrscr();
    int i,j,row,col,a[20][20];
    cout<<" Vertical Redundancy Check "<<endl;
    cout<<"–––––––––––––-"<<endl;
    cout<<"Enter the no. of bits in row :- ";
    cin>>row;
    cout<<"\nEnter the no. of bits in col :- ";
    cin>>col;
    cout<<"\nEnter the bit information(in terms of 0's & 1's):-"<<endl;
    for(i=1;i<=row;i++) {
        for(j=1;j<=col;j++) {
            cin>>a[i][j];
        }
    }
    for(i=1;i<=row;i++) {
        int m=0;
        for(j=1;j<=col;j++) {
          if(a[i][j]==1)
              {m++;}
        }
        int k=m%2;
        if(k==0){
          a[i][col+1]=0;
        }
        else{
          a[i][col+1]=1;
        }
    }
    cout<<" After Vertical Redundancy Check, above bits are represented as:- "<<endl;
    for(i=1;i<=row;i++){
      for(j=1;j<=col+1;j++){
        cout<<a[i][j]<<" ";
      }
      cout<<endl;
    }
    getch();
}
```

3. CheckSum

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
#include<stdio.h>
#define size 100

unsigned short int checksum(char []);
void valch(unsigned short int check, char message[]);

int main()
{
    clrscr();
    cout<<" Implementing Checksum "<<endl;
    cout<<"————————————-"<<endl;
    cout<<"\nEnter any data :- ";
    char message[size]={0};
    cin>>message;
    unsigned short int check = checksum(message);
    getch();
    valch(check,message);
    getch();
    return 0;
}

void valch(unsigned short int check, char message[])
{
    unsigned short int t[size], ch = 0;
    int i,j,n;
    cout<<endl<<endl;
    cout<<"Checking Checksum at Receiver side :- "<<endl;
    for(i=0,j=0;i<strlen(message);i=i+2,j++) {
        t[j]=message[i];
        t[j]=t[j]<<8;
        t[j]=t[j]+message[i+1];
    }
    n=j;
    for(i=0;i<n;i++) {
        printf("%X\n",t[i]);
        ch+=t[i];
    }
    printf("%X\n",check);
    ch+=check;
    printf("--\n");
    printf("%X\n",ch);
    printf(" 1 \n");
    printf("====\n");
    ch++;
    printf("%X\n",ch);
    ch=~ch;
    printf("%4X\n",ch);
    if(ch==0)
        cout<<"Checksum is valid...!";
    else
        cout<<"Checksum is invalid...!";
}

unsigned short int checksum(char message[])
{
    unsigned short int t[size];
    unsigned short int check = 0;
    int i,j,n;
    cout<<"Generating CheckSum at Transmitter side :- "<<endl;
    for(i=0,j=0;i<strlen(message);i=i+2,j++) {
        t[j]=message[i];
        t[j]=t[j]<<8;
        t[j]=t[j]+message[i+1];
    }
    n=j;
    for(i=0;i<n;i++) {
        printf("%X\n",t[i]);
        check+=t[i];
    }
    printf("0000\n");
    printf("---\n");
    printf("%X\n", check);
    printf(" 1\n");
    check+=1;
    printf("--\n");
    printf("%X\n",check);
    check=~check;
    printf("%X\n",check);
    cout<<"Checksum generated is :-";
    printf("%X\n",check);
    return check;
}
```

4. Parity Check

```
#include<iostream.h>
#include<conio.h>

void main()
{
    clrscr();
    int i,n,m=0,a[20];
    cout<<" Parity check "<<endl;
    cout<<"--------"<<endl;
    cout<<"\nEnter the no. of bits :- ";
    cin>>n;
    cout<<"\nEnter the bit information(in terms of 0's & 1's):-";
    for(i=1;i<=n;i++) {
        cin>>a[i];
    }
    for(i=1;i<=n;i++) {
        if(a[i]==1)
            m++;
    }
    if(m%2==0) {
        cout<<"\nParity is Even number of "<<0<<endl;
        a[n+1]=0;
    }
    else{
        a[n+1]=1;
        cout<<"\nParity is Odd number of "<<1<<endl;
    }
    cout<<"Hence, the bit information after parity check becomes :- ";
    for(i=1;i<=n+1;i++){
        cout<<a[i]<<" ";
    }
    getch();
}
```

5. Bit Stuffing Algorithm

```
#include<iostream.h>
#include<conio.h>
#include<string.h>

void main()
{
    int a[20],b[30],i,j,k,count,n;
    clrscr();
    cout<<" Bit Stuffing "<<endl;
    cout<<"———————-"<<endl;
    cout<<"\nEnter the size of frame :- ";
    cin>>n;
    cout<<"\nEnter the data of frame(interms of 0's & 1's) :-";
    for(i=0;i<n;i++){
        cin>>a[i];
    }
    i=0;count=1;j=0;
    do{
        if(a[i]==1){
            b[j]=a[i];
            for(k=i+1;a[k]==1 && k<n && count<5;k++){
                j++;
                b[j]=a[k];
                count++;
                if(count==5){
                    j++;
                    b[j]=0;
            }
            i=k;
         }
      }
      else{
         b[j]=a[i];
      }
      i++;
      j++;
   }while(i<n);

   cout<<"\nAfter bit stuffing,the frame becomes:-"<<endl;
   for(i=0;i<j;i++)
      cout<<b[i]<<" ";
   getch();
}
```

6. Character Stuffing

```
#include<iostream.h>
#include<conio.h>
#include<string.h>

void main(){
   int i=0,j=0,n;
   char a[20],b[20];
   clrscr();
   cout<<" Character Stuffing "<<endl;
   cout<<"––––––––––"<<endl;
   cout<<"\nEnter the string:-";
   cin>>a;
   n = strlen(a);
   b[0]='d';b[1]='l';b[2]='e';b[3]='s';b[4]='t';b[5]='x';
   j=6;
   while(i<n){
      if(a[i]=='d' && a[i+1]=='l' && a[i+2]=='e'){
         b[j]='d';j++;
         b[j]='l';j++;
         b[j]='e';j++;
      }
      b[j]=a[i];
      i++;
      j++;
```

```
    }
    b[j]='e'; j++;
    b[j]='t'; j++;
    b[j]='x'; j++;
    b[j]='d'; j++;
    b[j]='l'; j++;
    b[j]='e'; j++;
    b[j]='\0';

    cout<<"\nAfter character stuffing, the given string becomes :- ";
    for(int k=0;k<j;k++)
        cout<<b[k];
    getch();
}
```

7. Stop and Wait ARQ Protocol

```
#include<iostream.h>
#include<conio.h>

void main()
{
    clrscr();
    int i,j,f;
    char ch;
    cout<<" Stop and Wait ARQ Protocol "<<endl;
    cout<<"-------------------------------"<<endl;
    cout<<"\nEnter the total number of frames you want to send :- ";
    cin>>f;
    if(f<=0)
        cout<<"\nNo, Frames have been requested...!";
    else{
        i=0; j=0;
        while(i<f) {
            cout<<"\nFrame "<<i+1<<" is sent ...!";
            cout<<"\nIs Acknowledgement "<<j+1<<" Received? (y/n) :- ";
            cin>>ch;
            if(ch=='y') {
                i++;
                j++;
            }
            else{
                cout<<"\nSend Again.....!"<<endl;
```

```
         }
      }
      cout<<"\nInformation sent successfully...!"<<endl;
   }
   getch();
}
```

8. Go-Back-N ARQ Protocol

```
#include<iostream.h>
#include<dos.h>
#include<conio.h>
#include<stdlib.h>

void cal();

void main()
{
   int i,n,f,c,ans=0;
   clrscr();
   randomize();
   abc:
   cout<<" Go Back-N ARQ Protocol "<<endl;
   cout<<"========================"<<endl;
   cout<<"\nEnter the total number of frames to be send :-";
   cin>>n;
   f=random(n+1);
   cout<<"\nFrames are going to be transmitted ...."<<endl;
   for(i=1;i<=n;i++) {
      cout<<"\nFrame "<<i<<" is Sending ...";
      cal();
   }
   for(i=1;i<=n;i++) {
      if(i==f){
         cout<<"\n\tFrame "<<f<<" is lost, Resend it ...";
         cal();
         cout<<"\n\tAcknowledgment "<<i<<" is not received.";
         cal();
         ans=1;
      }
      else{
         cout<<"\nAcknowledgment "<<i<<" is Received..";
         cal();
      }
```

```
    }
    if(ans){
        ans=0;
        cout<<"\nResend Frames...";
        cout<<"\nAre You Ready to resend all frames once again(1-yes or 0-exit) :- ";
        int resend;
        cin>>resend;
        if(resend==0)
            goto xyz;
        for(i=1;i<=n;i++){
            cout<<"\n Frame "<<i<<" is sending... ";
            cal();
        }
        for(i=1;i<=n;i++){
            cout<<"\nAcknowledgment "<<i<<" is Received, After Resending...";
            cal();
        }
    }
    else{
        cout<<"\nUr Data Successfully Sent...!";
        cout<<"\nDo u want more Frames of Data to be send(0-exit & 1-continue) ?: ";
        cin>>c;
        if(c==1)
            goto abc;
        else
            goto xyz;
    }
    xyz:
    cout<<"\n Thank U...!"<<endl;
    cal();
    getch();
}

void cal(){
    for(int i=0;i<3;i++){
        sleep(i);
        cout<<".";
    }
}
```

9. Selective Repeat ARQ Protocol

```
#include<dos.h>
```

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>

void cal();

void main()
{
   int i,n,f,c,ans=0;
   do{
      clrscr();
      cout<<" Selective Repeat ARQ Protocol "<<endl;
      cout<<"-----------------"<<endl;
      randomize();
      abc:
      cout<<"\nEnter the number of frames to send :- ";
      cin>>n;
      f=random(n+1);
      cout<<"\nFames are going to be transmitted.... ";
      for(i=1;i<=n;i++){
         cout<<"\nFrame "<<i<<" Sending... ";
         cal();
      }
      for(i=1;i<=n;i++){
         if(i==f){
            cout<<"\n\tAcknowledgment "<<i<<" is not recevied... ";
            ans=1;
         }
         else{
            cout<<"\nAcknowledgement "<<i<<" Received...";
            cal();
         }
      }
      if(ans){
         ans=0;
         again:
         cout<<"\nR U Ready to Resend the Selected Frames(1-yes or 0-exit):- ";
         int resend;
         cin>>resend;
         if(resend==0)
            goto xyz;
         cout<<"\nU need to Resend only Frame "<<f<<"... ";
         cout<<"\nEnter The Frame to be Send :- ";
         int rsend;
```

```
            cin>>rsend;
            cout<<"\nSending the frame....";
            cal();
            if(f==rsend){
                cout<<"\nAcknowledment "<<f<<" received Successfully......!";
            }
            else{
                cout<<"\nUr Sended Frame Is Rejected,Resend It...";
                cal();
                goto again;
            }
        }
        else{
            cout<<"\nUr Data successfully sended...!";
            cout<<"\nDo U Want Another Frames to be Transmitted (0-exit and 1-continue...)? :- ";
            cin>>c;
            if(c==1)
                goto abc;
            else
                goto xyz;
        }
        xyz:
        cout<<"\nThank u !!!";
        cal();
        getch();
    }while(0);
}

void cal(){
    for(int i=0;i<3;i++){
        sleep(i);
        cout<<".";
    }
}
```

10. Dijkstra's Algorithm

```
#include<iostream.h>
#include<conio.h>
#define INFINITY 99
#define MAX 10

void dijkstra(int G[MAX][MAX], int n, int startnode);
```

```
int main()
{
   int G[MAX][MAX],i,j,n,u;
   clrscr();
   cout<<" Dijisktra's Algorithm "<<endl;
   cout<<"————————————-"<<endl;
   cout<<"\nEnter the total number of vertices :- ";
   cin>>n;
   cout<<"\nEnter the cost matrix: \n";
   for(i=1;i<=n;i++)
      for(j=1;j<=col;j++) // Note: pdf had cin>>G[i][j] in nested loops
         cin>>G[i][j];
   cout<<"\nEnter the Initial vertex :- ";
   cin>>u;
   dijkstra(G,n,u);
   getch();
   return 0;
}

void dijkstra(int G[MAX][MAX],int n,int startnode)
{
   int cost[MAX][MAX], distance[MAX], pred[MAX];
   int visited[MAX],count, mindistance, nextnode,i,j;
   for(i=1;i<=n;i++) {
      for(j=1;j<=n;j++) {
         if(G[i][j]==0)
            cost[i][j]=INFINITY;
         else
            cost[i][j]=G[i][j];
      }
   }
   for(i=1;i<=n;i++) {
      distance[i]=cost[startnode][i];
      pred[i]=startnode;
      visited[i]=0;
   }
   distance[startnode]=0;
   visited[startnode]=1;
   count=1;
   while(count<n-1) {
      mindistance=INFINITY;
      for(i=1;i<=n;i++) {
         if(distance[i]<mindistance&&!visited[i]) {
            mindistance=distance[i];
            nextnode=i;
         }
      }
      visited[nextnode]=1;
      for(i=1;i<=n;i++){
         if(!visited[i]){
            if(mindistance+cost[nextnode][i]<distance[i]){
               distance[i]=mindistance+cost[nextnode][i];
               pred[i]=nextnode;
            }
         }
      }
      count++;
   }
   cout<<"\nShortest path is :"<<endl;
   for(i=1;i<=n;i++){
      if(i!=startnode){
         cout<<"V"<<startnode<<" to V"<<i<<", Cost is=> "<<distance[i]<<endl;
      }
   }
}
```