**3. Develop a simple data link layer that performs the flow control using the sliding window protocol, and loss recovery using the Go-Back-N mechanism.**

#include<stdio.h>

int main()

{

int w,i,f,frames[50];

printf("Enter window size: ");

scanf("%d",&w);

printf("\nEnter number of frames to transmit: ");

scanf("%d",&f);

printf("\nEnter %d frames: ",f);

for(i=1;i<=f;i++)

scanf("%d",&frames[i]);

printf("\nWith sliding window protocol the frames will be sent in the following manner (assuming no corruption

of frames)\n\n");

printf("After sending %d frames at each stage sender waits for acknowledgement sent by the receiver\n\n",w);

for(i=1;i<=f;i++)

{

if(i%w==0)

{

printf("%d\n",frames[i]);

printf("Acknowledgement of above frames sent is received by sender\n\n");

}

else

printf("%d ",frames[i]);

}

if(f%w!=0)

printf("\nAcknowledgement of above frames sent is received by sender\n");

return 0;

}

**4. Implement Dijsktra’s algorithm to compute the shortest path through a network**

**PROGRAM FOR FINDING SHORTEST PATH**

#include<stdio.h>

#include<conio.h>

void main()

{

int path[5][5],i,j,min,a[5][5],p,st=1,ed=5,stp,edp,t[5],index;

clrscr();

printf("enter the cost matrix\n");

for(i=1;i<=5;i++)

for(j=1;j<=5;j++)

scanf("%d",&a[i][j]);

printf("enter the paths\n");

scanf("%d",&p);

printf("enter possible paths\n");

for(i=1;i<=p;i++)

for(j=1;j<=5;j++)

scanf("%d",&path[i][j]);

for(i=1;i<=p;i++)

{

t[i]=0;

stp=st;

for(j=1;j<=5;j++)

{

edp=path[i][j+1];

t[i]=t[i]+a[stp][edp];

if(edp==ed)

break;

else

stp=edp;

}

}

min=t[st];index=st;

for(i=1;i<=p;i++)

{

if(min>t[i])

{

min=t[i];

index=i;

}

}

printf("minimum cost %d",min);

printf("\n minimum cost path ");

for(i=1;i<=5;i++)

{

printf("--> %d",path[index][i]);

if(path[index][i]==ed)

break;

}

getch();

}

**Output:**

**5. Take an example subnet of hosts and obtain a broadcast tree for the subnet.**

#include<stdio.h>

int a[10][10],n;

void main()

{

int i,j,root;

printf("Enter no.of nodes:");

scanf("%d",&n);

printf("Enter adjacent matrix\n");

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

{

printf("Enter connecting of %d>%d::",i,j);

scanf("%d",&a[i][j]);

}

printf("Enter root node:");

scanf("%d",&root);

adj(root);

}

adj(int k)

{

int i,j;

printf("Adjacent node of root node::\n");

printf("%d\n",k);

for(j=1;j<=n;j++)

{

if(a[k][j]==1 || a[j][k]==1)

printf("%d\t",j);

}

printf("\n");

for(i=1;i<=n;i++)

{

if((a[k][j]==0) && (a[i][k]==0) && (i!=k))

printf("%d",i);

}}