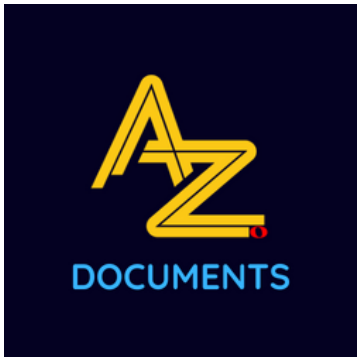
www.azdocuments.in

COMPUTER NETWORKS LAB

18ECL76

PART-B-LAB MANUAL



www.azdocuments.in

COMPUTER NETWORKS LAB-18ECL76
PART-B

Implement the following in C/C++:

1. Write a program for a HLDC flame to perform the following.
1) Bit stuffing
ii) Character stuffing.

2. Write a program for distance vector algorithm to find suitable path for transmission.

3. Implement Dijkstra's algorithm to compute the shortest routing path.

4. For the given data, use CRC-CCITT polynomial to obtain CRC code. Verify the program for the cases
a. Without error
b. With error

5. Implementation of Stop and Wait Protocol and Sliding Window Protocol

6. Write a program for congestion control using leaky bucket algorithm.

Part B

Steps for execution:

Step 1. Open Terminal
Step 2. type nano <filename.c>
example: lab1.c
Step 3. Type the program
Step 4. Save the program (ctrl+x)
Step5. Close the file (shift+y)
Step 6. Type gcc <filename.c>
Example gcc lab1.c
Step 7. type ./a.out

1. Write a program for a HLDC flame to perform the following.

1) Bit stuffing
ii) Character stuffing.

1a. BIT STUFFING PROGRAM

```
#include<stdio.h>
int main()
{
   int a[15];
   int i, j ,k,n,c=0,pos=0;
   printf("\n Enter the number of bits:  ");
   scanf("%d",&n);
   printf("\n Enter the bits: ");
   for(i=0;i<n;i++)
   scanf("%d",&a[i]);
   for(i=0;i<n;i++);
   {
      if(a[i]==1)
      {
         c++;
         if(c==5)
         {
            pos=i+1;
            c=0;
            for(j=n;j>=pos;j--)
            {
               k=j+1;
```

```
                a[k]=a[j];
            }
            a[pos]=0;
            n=n+1;
          }
      }
      else
      c=0;
   }
   printf("\n DATA AFTER STUFFING \n");
   for(i=0;i<n;i++)    {
      printf("%d",a[i]);
   }
}
```

Output:

Enter the number of bits: 12
Enter the bits: 1 0 1 1 1 1 1 1 1 0 1 1
DATA AFTER STUFFING
1011111011011

1b. CHARACTER STUFFING PROGRAM

```
#include<stdio.h>
#include<string.h>
void main()
{
int i=0,j=0,n,pos;
char a[20],b[50],ch;
printf("Enter the Characters:  ");
scanf("%s",&a);
printf("\nOrginal Data:%s",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';
      b[j+1]='l';
      b[j+2]='e';
      j=j+3;
   }
   b[j]=a[i];
   i++;
   j++;
}
b[j]='d';
b[j+1]='l';
b[j+2]='e';
b[j+3]='e';
b[j+4]='t';
b[j+5]='x';
b[j+6]='\0';
printf("\nTransmitted Data: %s\n",b);
printf("Received Data:%s",a);
}
```

Output-1:

Enter the Characters: abcdefabcd
Orginal Data: abcdefabcd
Transmitted Data: dlestxabcdefabcddleetx
Received Data: abcdefabcd

Output-2:

Enter the Characters: abcdabcde
Orginal Data: abcdabcde
Transmitted Data: dlestxabcdabcdedleetx
Received Data: abcdabcde

2. Write a program for distance vector algorithm to find suitable path for transmission.

```
#include<stdio.h>
struct node
{
unsigned dist[20];
unsigned from[20];
}
rt[10];
void main()
{
int costmat[20][20],source,desti;
int nodes,i,j,k,count=0;
printf("\nEnter the number of nodes : ");
scanf("%d",&nodes);
//Enter the nodes
printf("\nEnter the cost matrix :\n");
for(i=0;i<nodes;i++)
for(j=0;j<nodes;j++)
{
scanf("%d",&costmat[i][j]);
costmat[i][i]=0;
rt[i].dist[j]=costmat[i][j];
rt[i].from[j]=j;
}

do
{
count=0;
for(i=0;i<nodes;i++)
for(j=0;j<nodes;j++)
if(i!=j)
for(k=0;k<nodes;k++)
if(rt[i].dist[j]>rt[i].dist[k]+rt[k].dist[j])
{
rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];
rt[i].from[j]=rt[i].from[k];
count++;
}
}
while(count!=0);
for(i=0;i<nodes;i++)
```

```
{
printf("\n\n State Value For Router  %d\n",i+1);
for(j=0;j<nodes;j++)
printf("\t\nnode%d via %d Distance %d ",j+1,rt[i].from[j]
+1,rt[i].dist[j]);
}
printf("\n\n");
}
```

Output:

```
Enter the number of nodes : 4
Enter the cost matrix :
0 2 999 1
2 0 5 2
999 5 0 6
1 2 6 0
State Value For Router  1

node1 via 1 Distance 0
node2 via 2 Distance 2
node3 via 2 Distance 7
node4 via 4 Distance 1

 State Value For Router 2

node1 via 1 Distance 2
node2 via 2 Distance 0
node3 via 3 Distance 5
node4 via 4 Distance 2

 State Value For Router 3

node1 via 2 Distance 7
node2 via 2 Distance 5
node3 via 3 Distance 0
node4 via 4 Distance 6

 State Value For Router 4

node1 via 1 Distance 1
node2 via 2 Distance 2
node3 via 3 Distance 6
node4 via 4 Distance 0
```

3. Implement Dijkstra's algorithm to compute the shortest routing path.

```
#include<stdio.h>
#define INFINITY 9999
#define MAX 10

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

int main()
{
int G[MAX][MAX],i,j,n,u;
printf("Enter no. of vertices:");
scanf("%d",&n);
printf("\nEnter the adjacency matrix:\n");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
scanf("%d",&G[i][j]);
printf("\nEnter the starting node:");
scanf("%d",&u);
dijkstra(G,n,u);
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;
//pred[] stores the predecessor of each node
//count gives the number of nodes seen so far
//create the cost matrix
for(i=0;i<n;i++)
for(j=0;j<n;j++)
if(G[i][j]==0)
cost[i][j]=INFINITY;
else
cost[i][j]=G[i][j];
//initialize pred[],distance[] and visited[]
for(i=0;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;
//nextnode gives the node at minimum distance
for(i=0;i<n;i++)
if(distance[i]<mindistance&&!visited[i])
{
mindistance=distance[i];
nextnode=i;
}
//check if a better path exists through nextnode
visited[nextnode]=1;
for(i=0;i<n;i++)
if(!visited[i])
if(mindistance+cost[nextnode][i]<distance[i])
{
distance[i]=mindistance+cost[nextnode][i];
pred[i]=nextnode;
}
count++;
}

//print the path and distance of each node
for(i=0;i<n;i++)
if(i!=startnode)
{
printf("\nDistance of node%d=%d",i,distance[i]);
printf("\nPath=%d",i);
j=i;
do
{
j=pred[j];
printf("<-%d",j);
}while(j!=startnode);
}
}
```

Output:

Enter no. of vertices:5

Enter the adjacency matrix:
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0

Enter the starting node:0

Distance of node1=10
Path=1<-0
Distance of node2=50
Path=2<-3<-0
Distance of node3=30
Path=3<-0
Distance of node4=60
Path=4<-2<-3<-0

4. For the given data, use CRC-CCITT polynomial to obtain CRC code.

Verify the program for the cases
a. Without error
b. With error

```
#include<stdio.h>
#include<string.h>
#define N strlen(g)
char t[28],cs[28],g[]="10001000000100001";
int a,i,j;
void xor()
{
for(j = 1;j < N; j++)
cs[j] = (( cs[j] == g[j])?'0':'1');
}
void crc()
{
for(i=0;i<N;i++)
cs[i]=t[i];
do
{
if(cs[0]=='1')
xor();
for(j=0;j<N-1;j++)
cs[j]=cs[j+1];
cs[j]=t[i++];
}
while(i<=a+N-1);
}
int main()
{
printf("\nEnter data : ");
scanf("%s",t);
printf("\n---------------------------------------");
printf("\nGeneratng polynomial : %s",g);
a=strlen(t);
for(i=a;i<a+N-1;i++)
t[i]='0';
printf("\n---------------------------------------");
printf("\nModified data is : %s",t);
```

```
printf("\n--------------------------------------");
crc();
printf("\nChecksum is : %s",cs);
for(i=a;i<a+N-1;i++) t[i]=cs[i-a];
printf("\n--------------------------------------");
printf("\nFinal codeword is : %s",t);
printf("\n--------------------------------------");
printf("\nEnter received message ");
scanf("%s",t);
crc();
for(i=0;(i<N-1) && (cs[i]!='1');i++);
if(i<N-1)
printf("\nError detected\n\n");
else
printf("\nNo error detected\n\n");
printf("\n--------------------------------------\n");
return 0;
}
```

Output-1:

Enter data : 1011101

Generatng polynomial : 10001000000100001

Modified data is : 10111010000000000000000

Checksum is : 1000101101011000

Final codeword is : 10111011000101101011000

Enter received message 10111001100010110101000
Error detected

Output-2:

Enter data : 1011101

Generatng polynomial : 10001000000100001

Modified data is : 10111010000000000000000

Checksum is : 1000101101011000

Final codeword is : 10111011000101101011000

Enter received message 10111011000101101011000
No error detected
---

5. Implementation of Stop and Wait Protocol and Sliding Window Protocol

5a. Stop and Wait Protocol

```
#include <stdio.h>
int main()
{
   int i,f,frames[50];
   printf("\n Enter number of frames to transmit: ");
   scanf("%d",&f);
   printf("\n Enter the %d frames: ",f);
   for(i=1;i<=f;i++)
   scanf("%d",&frames[i]);

    for(i=1;i<=f;i++)
    {
       if((random()%2)==1)
       {
          printf("%d\n",frames[i]);
          printf("Acknowledgement of above frames sent is recived by
sender\n\n");
       }
       else
       {
          sleep(3);
          printf("negative acknowledgement resend %d frame\n",i);
          i=i-1;
          sleep(1);
       }
    }
    return 0;
}
```

Output-1:

Enter number of frames to transmit: 6

Enter the 6 frames: 1

4

5

7

8

9

1

Acknowledgement of above frames sent is recived by sender

negative acknowledgement resend 2 frame

4

Acknowledgement of above frames sent is recived by sender

5

Acknowledgement of above frames sent is recived by sender

7

Acknowledgement of above frames sent is recived by sender

8

Acknowledgement of above frames sent is recived by sender

negative acknowledgement resend 6 frame

negative acknowledgement resend 6 frame

9

Acknowledgement of above frames sent is recived by sender

5b. Sliding Window Protocol Program

```
#include <stdio.h>
int main()
{
   int w,i,f,frames[50];
   printf("Enter Window Size: ");
   scanf("%d",&w);
   printf("\n Enter number of frames to transmit: ");
   scanf("%d",&f);
   printf("\n Enter the %d frames: ",f);
   for(i=1;i<=f;i++)
   scanf("%d",&frames[i]);
    for(i=1;i<=f;i++)
    {
       if(i%w==0)
       {
          printf("%d\n", frames[i]);
          printf("Acknowledgment of above frames sent is recived by
sender\n\n");
       }
       else
       printf("%d\n",frames[i]);
    }
    if(f%w!=0)
    printf("Acknowledgment of above frames sent is recived by
sender\n\n");
    return 0;
```

}

Output-1:

Enter Window Size: 3

Enter number of frames to transmit: 5

Enter the 5 frames: 12 20 87 65 4

12

20

87

Acknowledgment of above frames sent is recived by sender

65

4

Acknowledgment of above frames sent is recived by sender

Output-2:

Enter Window Size: 5

Enter number of frames to transmit: 10

Enter the 10 frames: 10 20 30 40 50 60 70 80 90 100

10

20

30

40

50

Acknowledgment of above frames sent is recived by sender

60

70

80

90

100

Acknowledgment of above frames sent is recived by sender

6. Write a program for congestion control using leaky bucket algorithm.

```
#include<stdio.h>
#include<stdlib.h>
int bucket_size;
void bucket_input (int pkt_sz, int op_rt)
{
   if(pkt_sz>bucket_size)
   printf("\n\n Bucket Overflow\n");
   else
   {
      sleep(1);
      while(pkt_sz>op_rt)
      {
         printf("\n %d bytes outputted",op_rt);
         pkt_sz-=op_rt;
         sleep(1);
      }
      if(pkt_sz>0)
      printf("\n Last %d bytes sent\n", pkt_sz);
      printf("\n Bucket output Successful \n");

   }
}
```

```
int main()
{
   int i, op_rate, packet_size;
   printf("\n Enter Bucket Size: ");
   scanf("%d",&bucket_size);
   printf("\n Enter Output rate: ");
   scanf("%d",&op_rate);
   for(i=1; i<=5; i++)
   {
      sleep(1);
      packet_size = random()%1000;
      printf("\n packet number [%d] \t Packet
size=%d",i,packet_size);
      bucket_input(packet_size, op_rate);
   }
   return 0;
}
```

Output:

Enter Bucket Size: 500

Enter Output rate: 80

packet number [1] Packet size=383

80 bytes outputted

80 bytes outputted

80 bytes outputted

80 bytes outputted

Last 63 bytes sent

Bucket output Successful

packet number [2] Packet size=886

Bucket Overflow

packet number [3] Packet size=777

Bucket Overflow

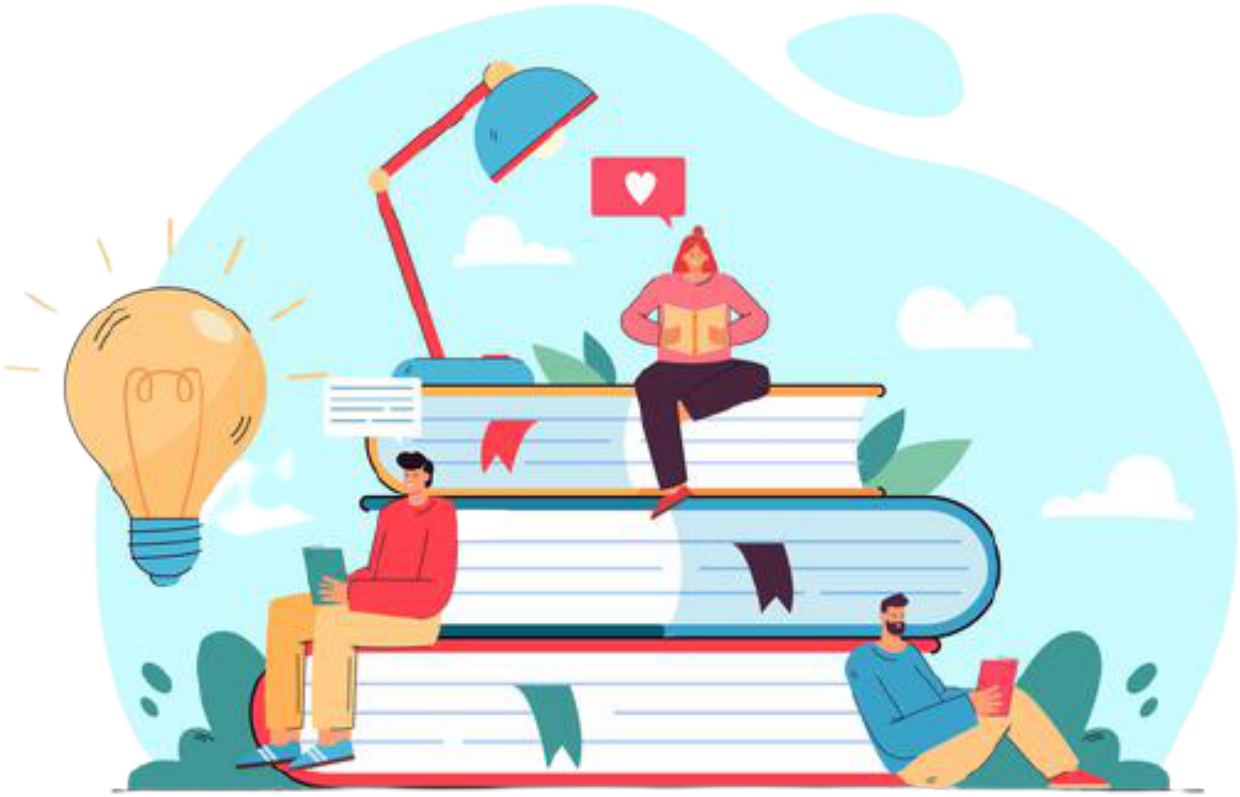
packet number [4] Packet size=915

Bucket Overflow

packet number [5] Packet size=793

Bucket Overflow

For more please do visit www.azdocuments.in



@azdocuments

https://www.azdocuments.in/