CN CODE

------

BIT STUFFING

------------

#include <stdio.h>

#include <stdlib.h>

#define MAXSIZE 100

int main()

{

char \*p,\*q;

char in[MAXSIZE]; char

stuff[MAXSIZE]; char

destuff[MAXSIZE];

int count=0;

printf("Enter the input character string (0 & 1 only):\n");

scanf("%s",in);

p=in;

q=stuff;

while(\*p!='\0')

{

if(\*p=='0')

{

\*q=\*p;

q++;

p++;

}

else

{

while(\*p=='1' && count!=3)

{

count++;

\*q=\*p;

q++;

p++;

}

if(count==3)

{

\*q='0';

q++;

}

count=0;

}

}

\*q='\0';

printf("\nThe stuffed character string is: ");

printf("\n%s",stuff);

p=stuff;

q=destuff;

while(\*p!='\0')

{

if(\*p=='0')

{

\*q=\*p;

q++;

p++;

}

else

{

while(\*p=='1' && count!=3)

{

count++;

\*q=\*p;

q++;

p++;

}

if(count==3)

{

p++;

}

count=0;

}

}

\*q='\0';

printf("\nThe destuffed character string is:");

printf("\n%s\n",destuff);

return 0;

}

--------------------------------------------------------------

BYTE/CHARACTER STUFFING

-----------------------

#include<stdio.h>

#include<string.h>

int main()

{

char a[30], fs[50] = " ", t[3], sd, ed, x[3], s[3], d[3], y[3];

int i, j, p = 0, q = 0;

printf("Enter characters to be stuffed:");

scanf("%s", a);

printf("\nEnter a character that represents starting delimiter:");

scanf(" %c", &sd);

printf("\nEnter a character that represents ending delimiter:");

scanf(" %c", &ed);

x[0] = s[0] = s[1] = sd;

x[1] = s[2] = '\0';

y[0] = d[0] = d[1] = ed;

d[2] = y[1] = '\0';

strcat(fs, x);

for(i = 0; i < strlen(a); i++)

{

t[0] = a[i];

t[1] = '\0';

if(t[0] == sd)

strcat(fs, s);

else if(t[0] == ed)

strcat(fs, d);

else

strcat(fs, t);

}

strcat(fs, y);

printf("\n After stuffing:%s", fs);

}

-----------------------------------------------------------------------------

CRC

---

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char text[100]; char

key[100];

char rem[100];

void crc()

{

int i,j;

int keylen,textlen;

char temp[100];

strcpy(temp,text);

keylen=strlen(key);

for(i=0;i<keylen-1;i++)

strcat(temp,"0");

textlen=strlen(temp);

strncpy(rem,temp,keylen);

while(i!=textlen)

{

if(rem[0]=='0')

{

strcpy(rem,&rem[1]);

rem[keylen-1]=temp[++i];

rem[keylen]='\0'; continue;

}

for(j=0;j<keylen;j++) rem[j]=((rem[j]-

'0')^(key[j]-'0'))+'0';

}

}

int main()

{

int i;

int choice;

while(1)

{

printf("\n1.Find CRC\t2.Check CRC\t3.Exit CRC\nYour Choice: \t");

scanf("%d",&choice);

switch(choice)

{

case 1:

printf("\nEnter the input string: ");

scanf("%s",text);

printf("\nEnter the key: ");

scanf("%s",key);

crc();

printf("The transmitted message is %s\n",strcat(text,rem)); break;

case 2:

printf("\nEnter the transmitted message: ");

scanf("%s",text);

printf("\nEnter the key: ");

scanf("%s",key);

crc();

for(i=0;i<strlen(key)-1;i++)

if(rem[i]=='1')

break;

if(i==strlen(key)-1)

printf("There is no error in the message\n"); else

printf("There is error in the message\n"); break;

case 3:

exit(0);

}

}

}

-----------------------------------------------------------------------------------

HAMMING CODE

-------------

#include<stdio.h>

void main() {

int data[10];

int dataatrec[10],c,c1,c2,c3,i;

printf("Enter 4 bits of data one by one\n");

scanf("%d",&data[0]);

scanf("%d",&data[1]);

scanf("%d",&data[2]);

scanf("%d",&data[4]);

//calculation of even parity

data[6]=data[0]^data[2]^data[4];

data[5]=data[0]^data[1]^data[4];

data[3]=data[0]^data[1]^data[2];

printf("\nEncoded data is\n");

for(i=0;i<7;i++)

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

printf("\n\nEnter Received data bits one by one\n");

for(i=0;i<7;i++)

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

c1=dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];

c2=dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];

c3=dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];

c=c3\*4+c2\*2+c1 ;

if(c==0) {

printf("\nno error while transmission of data\n");

}

else {

printf("\nerror on position %d",c);

printf("\ndata sent : ");

for(i=0;i<7;i++)

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

printf("\ndata received : ");

for(i=0;i<7;i++)

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

printf("\nCorrect message is\n");

//if errorneous bit is 0 we complement it else vice versa

if(dataatrec[7-c]==0)

dataatrec[7-c]=1;

else

dataatrec[7-c]=0;

for (i=0;i<7;i++) {

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

}

}

}

----------------

OR

---

#include <stdio.h>

#include <math.h>

int input[32];

int code[32];

int ham\_calc(int,int);

void main()

{

int n,i,p\_n = 0,c\_l,j,k;

printf("Please enter the length of the Data Word: ");

scanf("%d",&n);

printf("Please enter the Data Word:\n");

for(i=0;i<n;i++)

{

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

}

i=0;

while(n>(int)pow(2,i)-(i+1))

{

p\_n++;

i++;

}

c\_l = p\_n + n;

j=k=0;

for(i=0;i<c\_l;i++)

{

if(i==((int)pow(2,k)-1))

{

code[i]=0;

k++;

}

else

{

code[i]=input[j];

j++;

}

}

for(i=0;i<p\_n;i++)

{

int position = (int)pow(2,i);

int value = ham\_calc(position,c\_l);

code[position-1]=value;

}

printf("\nThe calculated Code Word is: ");

for(i=0;i<c\_l;i++)

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

printf("\n");

printf("Please enter the received Code Word:\n");

for(i=0;i<c\_l;i++)

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

int error\_pos = 0;

for(i=0;i<p\_n;i++)

{

int position = (int)pow(2,i);

int value = ham\_calc(position,c\_l);

if(value != 0)

error\_pos+=position;

}

if(error\_pos == 1)

printf("The received Code Word is correct.\n");

else

printf("Error at bit position: %d\n",error\_pos);

}

int ham\_calc(int position,int c\_l)

{

int count=0,i,j;

i=position-1;

while(i<c\_l)

{

for(j=i;j<i+position;j++)

{

if(code[j] == 1)

count++;

}

i=i+2\*position;

}

if(count%2 == 0)

return 0;

else

return 1;

}

----------------------------------------------------

SHORTEST PATH

-------------

#include <stdio.h>

#include <limits.h>

#define MAX\_VERTICES 100

// Function to find the vertex with the minimum distance value

int minDistance(int dist[], int sptSet[], int vertices) {

int min = INT\_MAX, minIndex;

for (int v = 0; v < vertices; v++) {

if (!sptSet[v] && dist[v] < min) {

min = dist[v];

minIndex = v;

}

}

return minIndex;

}

// Function to print the constructed distance array

void printSolution(int dist[], int vertices) {

printf("Vertex \tDistance from Source\n");

for (int i = 0; i < vertices; i++) {

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

}

}

// Function to implement Dijkstra's algorithm for a given graph and source vertex

void dijkstra(int graph[MAX\_VERTICES][MAX\_VERTICES], int src, int vertices) {

int dist[MAX\_VERTICES]; // The output array dist[i] holds the shortest distance from src to i

int sptSet[MAX\_VERTICES]; // sptSet[i] will be true if vertex i is included in the shortest path tree or the shortest distance from src to i is finalized

// Initialize all distances as INFINITE and sptSet[] as false

for (int i = 0; i < vertices; i++) {

dist[i] = INT\_MAX;

sptSet[i] = 0;

}

// Distance from source vertex to itself is always 0

dist[src] = 0;

// Find the shortest path for all vertices

for (int count = 0; count < vertices - 1; count++) {

// Pick the minimum distance vertex from the set of vertices not yet processed.

// u is always equal to src in the first iteration.

int u = minDistance(dist, sptSet, vertices);

// Mark the picked vertex as processed

sptSet[u] = 1;

// Update dist value of the adjacent vertices of the picked vertex.

for (int v = 0; v < vertices; v++) {

// Update dist[v] only if it is not in the sptSet, there is an edge from u to v,

// and the total weight of path from src to v through u is smaller than the current value of dist[v]

if (!sptSet[v] && graph[u][v] && dist[u] != INT\_MAX && dist[u] + graph[u][v] < dist[v]) {

dist[v] = dist[u] + graph[u][v];

}

}

}

// Print the constructed distance array

printSolution(dist, vertices);

}

int main() {

int vertices;

// Input the number of vertices

printf("Input the number of vertices: ");

scanf("%d", &vertices);

if (vertices <= 0 || vertices > MAX\_VERTICES) {

printf("Invalid number of vertices. Exiting...\n");

return 1;

}

int graph[MAX\_VERTICES][MAX\_VERTICES];

// Input the adjacency matrix representing the weighted graph

printf("Input the adjacency matrix for the graph (use INT\_MAX for infinity):\n");

for (int i = 0; i < vertices; i++) {

for (int j = 0; j < vertices; j++) {

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

}

}

int source;

// Input the source vertex

printf("Input the source vertex: ");

scanf("%d", &source);

if (source < 0 || source >= vertices) {

printf("Invalid source vertex. Exiting...\n");

return 1;

}

// Perform Dijkstra's algorithm

dijkstra(graph, source, vertices);

return 0;

}

----------------------------------------------------------------------------------------------------------

DISTANCE VECTOR ROUTING

------------------------

#include<stdio.h>

int dist[50][50],temp[50][50],n,i,j,k,x;

void dvr();

int main()

{

printf("\nEnter the number of nodes : ");

scanf("%d",&n);

printf("\nEnter the distance matrix :\n");

for(i=0;i<n;i++)

{

for(j=0;j<n;j++)

{

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

dist[i][i]=0;

temp[i][j]=j;

}

printf("\n");

}

dvr();

printf("enter value of i &j:");

scanf("%d",&i);

scanf("%d",&j);

printf("enter the new cost");

scanf("%d",&x);

dist[i][j]=x;

printf("After update\n\n");

dvr();

return 0;

}

void dvr()

{

for (i = 0; i < n; i++)

for (j = 0; j < n; j++)

for (k = 0; k < n; k++)

if (dist[i][k] + dist[k][j] < dist[i][j])

{

dist[i][j] = dist[i][k] + dist[k][j];

temp[i][j] = k;

}

for(i=0;i<n;i++)

{

printf("\n\nState value for router %d is \n",i+1);

for(j=0;j<n;j++)

printf("\t\nnode %d via %d Distance%d",j+1,temp[i][j]+1,dist[i][j]);

}

printf("\n\n");

}

output:-

Enter the number of nodes : 4

Enter the distance matrix :

0

12

9

16

20

0

18

4

15

8

0

32

41

24

51

0

State value for router 1 is

node 1 via 1 Distance0

node 2 via 2 Distance12

node 3 via 3 Distance9

node 4 via 4 Distance16

State value for router 2 is

node 1 via 1 Distance20

node 2 via 2 Distance0

node 3 via 3 Distance18

node 4 via 4 Distance4

State value for router 3 is

node 1 via 1 Distance15

node 2 via 2 Distance8

node 3 via 3 Distance0

node 4 via 2 Distance12

State value for router 4 is

node 1 via 1 Distance41

node 2 via 2 Distance24

node 3 via 2 Distance42

node 4 via 4 Distance0

enter value of i &j:

1

3

enter the new cost

68

After update

State value for router 1 is

node 1 via 1 Distance0

node 2 via 2 Distance12

node 3 via 3 Distance9

node 4 via 4 Distance16

State value for router 2 is

node 1 via 1 Distance20

node 2 via 2 Distance0

node 3 via 3 Distance18

node 4 via 3 Distance30

State value for router 3 is

node 1 via 1 Distance15

node 2 via 2 Distance8

node 3 via 3 Distance0

node 4 via 2 Distance12

State value for router 4 is

node 1 via 1 Distance41

node 2 via 2 Distance24

node 3 via 2 Distance42

node 4 via 4 Distance0

--------------------------------