# *CYCLIC REDUNDANCY CHECK*

**#include<stdio.h>**

**#define n=10**

|  |  |
| --- | --- |
| **void main()**  **{**  **int i,j,data,key,d[n],k[n],t[n],l,m,r,q=0;**  **printf("Enter the no. of bits in data : ");**  **scanf("%d",&l);**  **printf("Enter data that is being sent : ");**  **scanf("%d",&data);**  **printf("Enter the no. of bits in key : ");**  **scanf("%d",&m);**  **printf("Enter key to check that data : ");**  **scanf("%d",&key);**    **con(data,d,l);**  **con(key,k,m);**  **for(i=0;i<l;i++)**  **{**  **if(l-i-1>=m-1)**  **{**  **if(d[i]==1)**  **{**  **t[q++]=1;**  **for(j=0,r=i;j<m;j++,r++)**  **{**  **if((d[r]==0 && k[j]==1)||(d[r]==1 && k[j]==0))**  **{**  **d[r]=1;**  **}**  **else if((d[r]==0 && k[j]==0)||(d[r]==1 && k[j]==1))**  **{**  **d[r]=0;**  **}**  **}**  **}**  **else**  **t[q++]=0;**  **}**  **else**  **break;**  **}** | **printf("\nThe quotient : ");**  **for(j=0;j<l-m+1;j++)**  **{**  **printf("%d",t[j]);**  **}**  **printf("\nThe remainder : ");**  **for(j=0;j<l;j++)**  **{**  **printf("%d",d[j]);**  **}**  **i=1;**  **for(j=0;j<l;j++)**  **{**  **if(d[j]==1)**  **{**  **i=0;**  **break;**  **}**  **}**    **if(i==1)**  **printf("\nThe data sent is correct");**  **if(i==0)**  **printf("\nThe data sent is corrupted");**  **}**  **------------x----------x-------**  **void con(int num, int arr[n],int z)**  **{**  **int k,i=num,j=0,a[n];**  **while(i!=0)**  **{**  **k=i%10;**  **i/=10;**  **a[j++]=k;**  **}**  **for(i=0,k=z-1;k>=0;i++,k--)**  **{**  **arr[i]=a[k];**  **}**  **}** |

# *HAMMING CODE*

**#include<stdio.h>**

**#include<math.h>**

**#define m 20**

|  |  |
| --- | --- |
| **void main()**  **{**  **int i,j,k,x,e,r=0,p,n,hc[m],par[6],temp[m];**  **long int ham;**  **printf("Enter the parity:\n1. even\n2. odd : ");**  **scanf("%d",&p);**  **printf("Enter no.of bits in hamming code : ");**  **scanf("%d",&n);**  **printf("Enter the hcode that is being sent : ");**  **scanf("%ld",&ham);**  **printf("\nham = %ld",ham);**  **con(ham,hc,n);**  **printf("\nThe string is : ");**  **for(i=1;i<=n;i++)**  **printf("%d",hc[i]);**  **while(pow(2,r)<(n+1))**  **//returns the array going from 1 to n for lsb-msb**  **r++;**  **for(i=0;i<r;i++)**  **{**  **j=1;**  **e=0;**  **x = pow(2,i);**  **printf(“Bits for parity bit 2^%d= %d :",i,x);**    **for(k=1;k<x;k++)**  **j++;**  **while(j<=n)**  **{**  **for(k=1;k<=x;k++)**  **{**  **if(j<=n)**  **{**  **printf("%d ",j);**  **temp[e++]=hc[j++];**  **}**  **else**  **break;**  **}**  **if(j>n)**  **break;**  **for(k=1;k<=x;k++)**  **j++;**  **}**  **printf("\n The bits : ");**  **for(j=0;j<e;j++)**  **printf("%d",temp[j]);** | **if(p==1)**  **{**  **x=0;**  **for(j=0;j<e;j++)**  **{**  **if(temp[j]==1)**  **x++;**  **}**  **if(x%2==0)**  **par[i]=0;**  **else**  **par[i]=1;**  **}**    **else if(p==2)**  **{**  **x=0;**  **for(j=0;j<e;j++)**  **{**  **if(temp[j]==1)**  **x++;**  **}**  **if(x%2==1)**  **par[0]=0;**  **else**  **par[1]=1;**  **}**  **}**    **printf("\n The error bit is : ");**  **for(i=0;i<r;i++)**  **{**  **printf("%d",par[i]);**  **}**  **printf("\n");**  **}**  **------------x---------x---------------**  **void con(long int num, int arr[],int z)**  **{**  **int j=1;**  **long int k,i=num;**  **printf("\ni=%ld",num);**  **while(i!=0)**  **{**  **k=i%10;**  **i/=10;**  **arr[j++]=k;**  **}**  **}** |

# SOCKET PROGRAMMING – TCP

|  |
| --- |
| **CLIENT SIDE CODE – CLIENT.PY**  **import socket**  **s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**  **#creates socket**  **port = 12345**  **s.connect(('localhost', port))**  **#connecting to the socket on the port**  **print(s.recv(1024).decode('utf-8'))**  **data=input("Enter the string : ")**  **s.send(data.encode('utf-8'))**  **print("The reversed string is",str(s.recv(1024).decode('utf-8')))**  **print(s.recv(1024).decode('utf-8'))**  **s.close()**  **'''Output :**  **> python tcp\_client.py**  **Thank you for connecting**  **Enter the string : neen**  **The reversed string is neen**  **The string is palindrome'''** |
| **SERVER SIDE CODE – SERVER.PY**  **import socket**  **s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**  **print("Socket successfully created")**  **port = 12345**  **s.bind(('localhost', port))**  **print("socket binded to %s" %(port))**  **s.listen(5)**  **print("socket is listening")**  **conn, addr = s.accept()**  **print('Got connection from : ',addr)**  **conn.send(b'Thank you for connecting')**  **data = conn.recv(1024).decode('utf-8')**  **print("The received string is : "+str(data))**  **data1 = data[::-1]**  **conn.send(data1.encode('utf-8'))**  **if(data1==data):**  **conn.send(b'The string is palindrome')**  **else:**  **conn.send(b'The string is not palindrome')**  **conn.close()**  **'''Output :**  **> python tcp\_server.py**  **Socket successfully created**  **socket binded to 12345**  **socket is listening**  **Got connection from : ('127.0.0.1', 52551)**  **The received string is : neen’’’** |

# SOCKET PROGRAMMING – UDP

|  |
| --- |
| **CLIENT SIDE CODE – CLIENT.PY**  **import time**  **import socket**  **client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)**  **#creating a socket connection**  **message = b'Thanks for connecting'**  **addr = ('localhost', 12000)**  **client\_socket.sendto(message, addr)**  **#sending a request to server through message**  **data, server = client\_socket.recvfrom(1024)**  **print(str(data.decode('utf-8')))**  **message=input("Enter the string : ").encode('utf-8')**  **client\_socket.sendto(message, addr)**  **data, server = client\_socket.recvfrom(1024)**  **print(str(data.decode('utf-8')))**  **'''Output:**  **> python udp\_client.py**  **Connected to server**  **Enter the string : neen**  **The string is palindrome'''** |
| **SERVER SIDE CODE – SERVER.PY**  **import random**  **import socket**  **server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)**  **server\_socket.bind(('localhost', 12000))**  **#creating socket connection**  **message, address = server\_socket.recvfrom(1024)**  **print(str(message.decode('utf-8')))**  **#recieving and accepting a request**  **data=b'Connected to server'**  **server\_socket.sendto(data, address)**  **message, address = server\_socket.recvfrom(1024)**  **message = message.decode('utf-8')**  **message1 = message[::-1]**  **print("The string sent is : "+str(message))**  **#reversing string**  **if(message1==message):**  **server\_socket.sendto(b'The string is palindrome', address)**  **else:**  **server\_socket.sendto(b'The string is not palindrome', address)**  **'''Output:**  **> python udp\_server.py**  **Thanks for connecting**  **The string sent is : neen'''** |