Computer Networks

~~Written~~ Assignment -2

Aim:-

Implementation of Hamming Code for Error Detection and Correction.

Theory:-

Hamming Code:-

It is a block code that is capable of detecting up to two simultaneous bit errors and correcting single bit errors.

In this coding method, the source encodes the message by inserting redundant bits within the message. These redundant bits are extra bits that are generated and inserted at specific positions in the message itself to enable error detection and correction. When the destination receives this message, it performs recalculations to detect errors and find position of the bit which has an error.

Steps to Encoding:-

1] Calculation of Number of Redundant Bits.
2] Positioning of Redundant Bits
3] Calculating values of each redundant bit.

~~Coding~~

Decoding a message in Hamming Code :-

Once the receiver gets an incoming message, it performs recalculations to detect errors and correct them if encountered any.

Steps are as follows:-

1] Calculation of the number of redundant bits

2] Positioning of Redundant Bits

3] Parity Checking

4] Error Detection and correction.
- Decimal equivalent of parity bits is converted to binary. If it is 0, there is no error. Otherwise the decimal value gives the bit position which has error.

eg:- if $C_1C_2C_3C_4 = 1001$, the bit at position 9 i.e decimal equivalent of 1001 has an error.

The bit is flipped to get the correct message.

Conclusion:-

Hamming Code is successfully implemented and studied.

Program:-

```
#include<stdio.h>

int main() {
    int data[10];
    int dataatrec[10],c,c1,c2,c3,i;

    printf("Enter 4 bits of data consecutively:-\n");
    scanf("%d",&data[0]);
    scanf("%d",&data[1]);
    scanf("%d",&data[2]);
    scanf("%d",&data[4]);

    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("\nThe Encoded data is:-\n");
for(i=0;i<7;i++)
     printf("%d",data[i]);

    printf("\n\nEnter received data bits consecutively:-\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 encountered while transmission of data\n");
    }
else {
printf("\nError ecnountered on position %d",c);

printf("\nData sent is:- ");
     for(i=0;i<7;i++)
      printf("%d",data[i]);

printf("\nData received is:- ");
     for(i=0;i<7;i++)
      printf("%d",dataatrec[i]);
printf("\nCorrect message is:-\n");
```
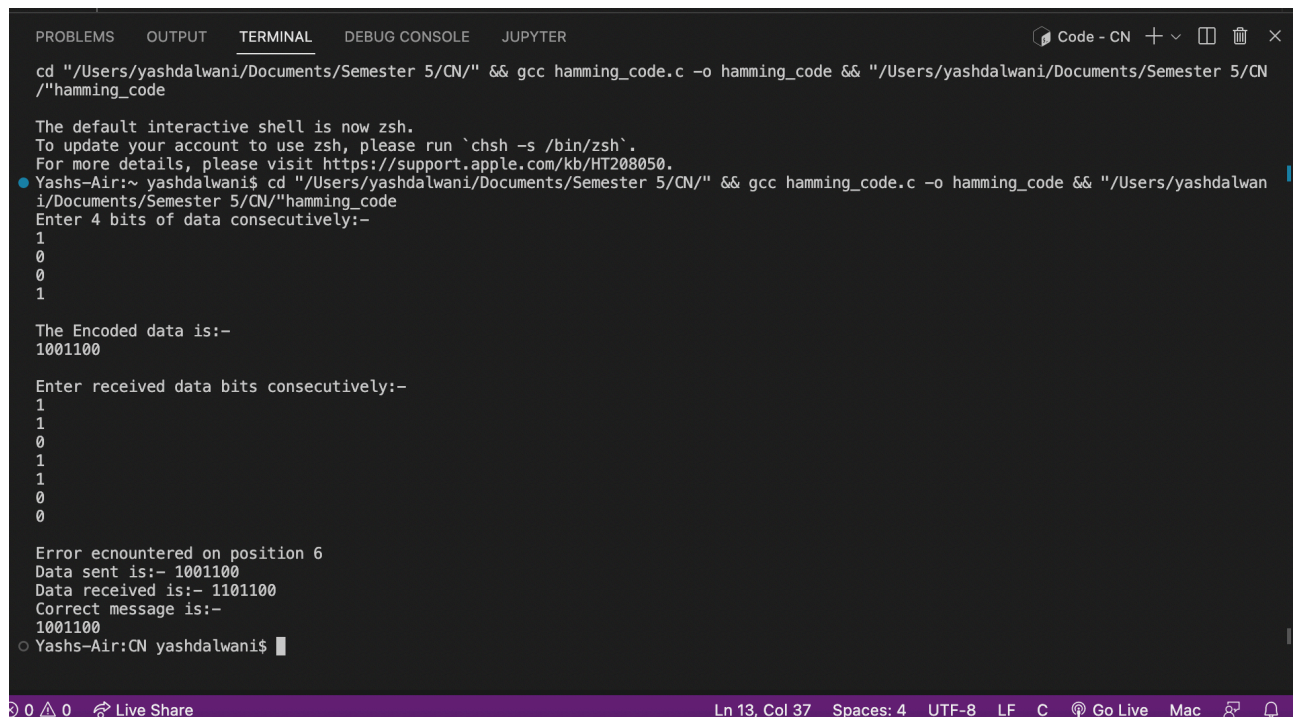
```
if(dataatrec[7-c]==0)
dataatrec[7-c]=1;
      else
dataatrec[7-c]=0;
for (i=0;i<7;i++) {
printf("%d",dataatrec[i]);
}
printf("\n");}
}
```

Output:-