practical - 6

Date : 21/8/24

AIM:

write a program to implement error detection and correction using HAMMING code concept. Make a test run to input data stream and verify error correction features.

Error correction at Data Link Layer:

Hamming code is a set of error correction codes that can be used to detect and correct the errors that can occur when the data is transmitted from the sender to the receiver. Developed by R.W. Hamming for error correctione.

create sender program with below features:

1) Input to sender file should be a text of any length. program should convert ~~of~~ text to ~~b~~ binary.

2) Apply hamming code concept on the binary data and add redundant bits.

3) Save this output in a file called channel.

create a receiver program with below features:

1) Receiver program should read the input from channel file.

2) Apply hamming code on the binary data to check for errors.

3) If there is an error, display the position of the error.

4) Else remove the redundant bits and convert the binary data to ascii and display the output.

student observation:

```
import java.util.Scanner;
public class HammingCode {
        static int [ ]
calculateParityBits (int[] data, int r )
{
        for (int i=0; i<r; i++) {
        int x = (int) Math.pow(2 ,i);
        for (int j= 1; j < data.length;
                                j++ ){
```

```
            if (((j >> i) & 1) == 1) {
                if (x != j) {
                    data [x] = data [x] ^ data [j];
                }
            }
        }
    }
    return data ;
}

static int [] generateHammingcode (string input,
                                   int m, int r) {

    int [] data = new int [r+m +1] ;
    int j = m-1 ;
    for (int i=1 ; i < data.length ; i++) {
        if ((Math.ceil (Math.log (i) / Math.log(2))
             - Math.floor (Math.log (i)/Math.log(2)))
                                              == 0) {
            data [i] = 0 ;
        } else {
            data [i] = input .charAt (j) - '0' ;
            j -- ;
        }
    }
    return data ;
}
```

```
static void printHammingcode (int[] data) {
for (int i = 1 ; i < data.Length ; i++) {
    System.Out.print (data[i]);
}
system.out.println ();
}
static void introduceError (int [] data, int position)
{
    data [position] = data [position] = ~~0~~ ~~?1?~~
                                  == 0 ? 1 : 0;
}
static void detectAndcorrection ( int[] data , int
                                               r)
{
    int errorposition =0;
    for ( int i = 0 ; i < r ; i++) {
        int x = (int) Math.pow (2, i) ;
        int parity = 0 ;
        for (int j = 1 ; j < data.Length ; j++) {
            if (((j << i) & 1) == 1) {
                parity = parity ^ data [j] ;
            }
}
if ( parity != 0 ) {
        errorposition += x ;
    }
}
if (errorposition != 0){
    system.out.println("Error detected at position:"
                            + errorposition);
    data[errorposition ]= data[errorposition]
                                    ==0?1:0;
    system.out.println ("Error correted.");
 }
  else {
      system.out.println ("No error detected.");
  }
}

public static void main (string[] args){
     Scanner scanner =new scanner (system.in);
     system.out.print("Enter the data bits:");
     string input = scanner .nextline ();
     int m = input .length();
     int x= 1;
```

```
while (Math.pow(2,r) < (m+r+1)) {
        r++;
    }
    int[] data = generateHammingCode (input, m, r);
     data = calulate parity Bits (data, r);
    System.out.println ("Generating Hamming code :");
    printHammingCode (data);
    System.out.print ("Enter the position to introduce
                        an error :");
    int errorposition = scanner.nextInt ();
    introduceError (data, errorposition);
    System.out.println ("Hamming code with error :");
    printHamming code (data);
    detectAnd correctError (data, r);
    System.out.println ("corrected Hamming code :");
    printHamming code (data);
  }
}.
```

Input & output :

```
Enter the input string : hello.

Generated hamming code : 1101110110000 1110
    01 01 011 011 000 111 011 00011 01111

Enter the position ~~to stimulate~~ of error : 3

Hamming code with error : 111111 0110 0001110
    01 01 011 011 000 111 011 00011 01 411

Error detected at position : 3.

corrected Hamming code : 11011101100001110 01
    01 011 011 000 111 01 1 00011 01111

corrected bit at position 3 : 0

corrected string : hello.
```

Result :

The output have been verified.

[illegible] 28/8/24