Exp no: 6 experiment-6 Hamming code

Date : 25.8.26

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 feature.

code :

```
#include <stdio.h>
#include <math.h>
int calc_parity (int data[], int size, int
    int parity = 0;
    for (int i=0; i< size; i++){
        if (i & (i<<p)){
            parity ^= data[i];
        }
    }
    return parity;
}
for (i=0; ; i<r; i++)
{
    int parity_bit = pow(2, i);
    int parity=0;
    for (j=1; j<=n; j++) {
        if (j & parity_bit)
            parity ^= code[j];
    }
    code[parity_bit] = parity
}
```

```
for (i = n; i >= 1; i--)
printf("%d", code[i]);
printf("\n");
}
void detectAndCorrect()
{
unit n = m + r;
int i, j, errorPos
for (i = 0; i < r; i++)
{
int parityPos = pow(2, i);
int parity = 0;
for(j = 1; j <= n; j++)
{
if (j & parityPos)
parity ^= code[j];
}
if (parity != 0)
errorRes += parityPos;
}
if (counter == 0) {
printf("No error");
}
else if (errorPos <= n) {
printf("Error at Position: %d", errorPos);
code[errorPos] ^= 1;
printf("corrected code:");
for(i = n; i >= 1; i--)
Printf("\n");
}
use a
```

```
printf("Multiple bit errors");
    }
}
int main()
{ int i, pos;
  scanf("%d", &m);
  if (m<1 || m>1) {
    printf("Invalid");
    return 0;
  }
for (i=0; i<m; i++) {
    printf("Enter bit %d:", i+1);
    scanf("%d", &data[i]);
  }
  generateHammingCode();
printf("Position to enter error");
scanf("%d", &pos);
if(pos!=0 && pos<=m+r)
  {
    code[POS] ^= 1;
    for(i=m+r; i>=1; i--)
      printf("%d", code[i]);
    printf("\n");
  }
  detectAndCorrect();
    return 0;
}
```

Sample I/P. O/P

Enter no. of data bits: 4

Enter bit 1: 1

Enter bit 2: 0

Enter bit 3: 1

Enter bit 4: 1

Generated Hamming code: 1100110

Enter position for error: 3

Code after error: 1100010

Error detected at position: 3

Corrected code: 1100110

Result:

Hence the hamming code concept has been implented successfully.