EXPERIMENT NO. 5

SEMESTER: V   DATE OF PERFORMANCE: 11/08/22

SUBJECT: CN Lab   DATE OF SUBMISSION: 12/08/22

NAME OF THE STUDENT: shrikrishna umbare   ROLL NO.: 57

AIM	Write a program to simulate Hamming code generation, detection and correction.
LEARNING OBJECTIVE	The student will demonstrate the working of (7,4) hamming code.
LEARNING OUTCOME	The student will be able to detect and correct errors using hamming code.
COURSE OUTCOME	CSL502.3: Simulate and explore networking algorithms and protocols.
PROGRAM OUTCOME	PO1,PO2,PO3,PO4,PO5,PO9,PO10,PSO1,PSO2,PSO3
BLOOM'S TAXONOMY LEVEL	Apply
THEORY	**Parity bits:** The bit which is appended to the original data of binary bits so that the total number of 1s is even or odd. **Even parity:** To check for even parity, if the total number of 1s is even, then the value of the parity bit is 0. If the total number of 1s occurrences is odd, then the value of the parity bit is 1. **Odd Parity:** To check for odd parity, if the total number of 1s is even, then the value of parity bit is 1. If the total number of 1s is odd, then the value of parity bit is 0. **Algorithm of Hamming code:** • An information of 'd' bits are added to the redundant bits 'r' to form d+r. • The location of each of the (d+r) digits is assigned a decimal value. • The 'r' bits are placed in the positions $1,2,.....2^{k-1}$. • At the receiving end, the parity bits are recalculated. The decimal value of the parity bits determines the position of an error.

LAB EXERCISE	**Write program in C/C++/Java/Python (Write separate programs for odd and even parity)**
	1. Ask the user whether the program will work for even parity (or for odd) parity.
	2. The user can enter the 4-bit data .
	3. Complete Code Word for this and can be generated by calculating for the parity bits:
	P1=(D3,D5,D7)
	P2=(D3,D6,D7)
	P4=(D5,D6,D7)
	4.Enter the Received codeword (with or without error).Notify the user to introduce error at only one bit position.
	5.Check bits 1,3,5,7.....to generate C1.Check bits 2,3,6,7.....to generate C2.Check bits 4,5,6,7.....to generate C3.Generate the error codeword =C3C2C1.
	6.If error is there,it will be reflected at which position and will display the corrected code word by inverting the respective bit.
	7.Decode the data bits. /* Sample Output
	This is hamming code error detection and correction using EVEN parity
	Enter 4 data bits.D7 D6 D5 D3
	Enter the value of D7:1
	Enter the value of D6:0
	Enter the value of D5:1
	Enter the value of D3:0
	3 parity bits are required for the transmission of data bits.
	SENDER:
	The data bits entered are: 1 0 1 0

The Parity bits are:

Value of P4 is 0

Value of P2 is 1

Value of P1 is 0

The Hamming code is as follows :-

D7 D6 D5 P4 D3 P2 P1

1 0 1 0 0 1 0

Enter the hamming code with error at any position of your choice.

NOTE: ENTER A SPACE AFTER EVERY BIT POSITION.

Error should be present only at one bit position

1 0 1 0 1 1 0

RECEIVER:

Error is detected at position 3 at the receiving end.

Correcting the error....

The correct code is 1 0 1 0 0 1 0

The decoded data is:1010

*/

Program:-

```c
#include<stdio.h>
#include<conio.h>
int dash(int n)
{
    for(int i = 0;i<n;i++)
    {
    printf("-");
    }
}
```

```c
int main()
{
    int a,b,c,d;
    int data[7];
    int i;
    int ch;
    again:
    printf("\nPress 1 for Even Parity\t[OR]\tPress 2 for Odd
Parity : \n");
    scanf("%d",&ch);
    start:
    dash(10);
    printf("\nEnter 4 data bits - D7 D6 D5 D3 : \n");

    printf("Enter the value of D7 : ");
    scanf("%d",&a);
    if(a!=0&&a!=1)
    {
        printf("Invalid number! Return to first step");
        goto start;
    }
    printf("Enter the value of D6 : ");
    scanf("%d",&b);
    if(b!=0&&b!=1)
    {
        printf("Invalid number! Return to first step");
        goto start;
    }
    printf("Enter the value of D5 : ");
    scanf("%d",&c);
    if(c!=0&&c!=1)
    {
        printf("Invalid number! Return to first step");
        goto start;
    }
    printf("Enter the value of D4 : ");
    scanf("%d",&d);
    if(d!=0&&d!=1)
    {
        printf("Invalid number! Return to first step");
        goto start;
    }
    printf("\n----------------------Sender--------------------
--------\n");
    printf("The data bits entered are : %d %d %d %d ",a,b,c,d);
```

```c
    data [0] = a;
    data [1] = b;
    data [2] = c;
    data [3] = 0;
    data [4] = d;
    data [5] = 0;
    data [6] = 0;
    int size = sizeof(data)/sizeof(data[0]);


    if(ch == 1)
    {
    data[6] = data[4]^data[2]^data[0];
    data[5] = data[4]^data[1]^data[0];
    data[3] = data[2]^data[1]^data[0];

    // Hamming Code - Output :
    printf("\nValue of P1 : %d\nValue of P2 : %d\nValue of P4 :
%d",data[6],data[5],data[3]);
    printf("\nThe Hamming code is as follows :\n");
    printf("\nD7 D6 D5 P4 D3 P2 P1\n");
    for(i = 0;i<size;i++)
    {
        printf(" %d ",data[i]);
    }
    printf("\n------------------------------------------------
----------------------\n");
    }
    else if(ch==2)
    {
    data[6] = !(data[4]^data[2]^data[0]);
    data[5] = !(data[4]^data[1]^data[0]);
    data[3] = !(data[2]^data[1]^data[0]);

    // Hamming Code - Output :
    printf("\nValue of P1 : %d\nValue of P2 : %d\nValue of P4 :
%d",data[6],data[5],data[3]);
    printf("\nThe Hamming code is as follows :\n");
    printf("\nD7 D6 D5 P4 D3 P2 P1\n");
    for(i = 0;i<size;i++)
    {
        printf(" %d ",data[i]);
    }
```

```c
    printf("\n------------------------------------------------
------------------------\n");
    }
    else
    {
        printf("Wrong choice! Try again");
        goto again;
    }
    // Enter error 7 bit stream

    int err[7];
    int c1, c2, c3;
    int ch1;
    int val;
    printf("\n------------------------Reciever----------------
---------------------------\n");
    printf("\nEnter Choice for Correction of Even Parity or Odd
Parity :\n");
    printf("\nPress 1 for Even Parity\t[OR]\tPress 2 for Odd
Parity : ");
    scanf("%d",&ch1);
    printf("\nEnter 7 bit error stream : ");
    int size1 = sizeof(err)/sizeof(err[0]);
    for(i=0;i<size1;i++)
    {
        scanf("%d",&err[i]);
    }
    printf("\nEntered Bit Stream : ");
    for(i=0;i<size1;i++)
    {
        printf(" %d " ,err[i]);
    }
    // checking the error bit

    if(ch==1)
    {
    c1 = err[6]^err[4]^err[2]^err[0];
    c2 = err[5]^err[4]^err[1]^err[0];
    c3 = err[3]^err[2]^err[1]^err[0];
        val = c1*1+c2*2+c3*4;
        printf("\nError at location : %d",val);
    }
    else if(ch==2)
    {
    c1 = !(err[6]^err[4]^err[2]^err[0]);
```

```c
    c2 = !(err[5]^err[4]^err[1]^err[0]);
    c3 = !(err[3]^err[2]^err[1]^err[0]);
        val = c1*1+c2*2+c3*4;
        printf("\nError at location : %d",val);
    }
    err[size-val] = !(err[size-val]);
    printf("\n----------------------------------------------------
----------------------\n\n");
    printf("\nCorrected Code : ");
    for(i=0;i<size1;i++)
    {
        printf("%d",err[i]);
    }
    printf("\nThe Decoded data is : %d %d %d
%d",err[0],err[1],err[2],err[4]);
    printf("\n\n-----------------------------------------------------
------------------------\n");
    getch();
}
```

Output:-

Case 1: Even Parity:

```
PS C:\C++\AOA> cd "c:\C++\AOA\" ; if ($?) { gcc hamming2.c -o hamming2 } ; if ($?) { .\hamming2 }

Press 1 for Even Parity [OR]    Press 2 for Odd Parity :
1
----------
Enter 4 data bits - D7 D6 D5 D3 :
Enter the value of D7 : 1
Enter the value of D6 : 1
Enter the value of D5 : 1
Enter the value of D4 : 1

-----------------------Sender----------------------------
The data bits entered are : 1 1 1 1
Value of P1 : 1
Value of P2 : 1
Value of P4 : 1
The Hamming code is as follows :

D7 D6 D5 P4 D3 P2 P1
 1  1  1  1  1  1  1
-----------------------------------------------------------------

------------------------Reciever-----------------------------------

Enter Choice for Correction of Even Parity or Odd Parity :

Press 1 for Even Parity [OR]    Press 2 for Odd Parity : 1

Enter 7 bit error stream : 1 1 1 1 1 1 0

Entered Bit Stream :  1  1  1  1  1  1  0
Error at location : 1
-----------------------------------------------------------------


Corrected Code : 1111111
The Decoded data is : 1 1 1 1
-----------------------------------------------------------------
```

Case 2: Odd Parity:

```
PS C:\C++> cd "c:\C++\AOA\" ; if ($?) { gcc hamming2.c -o hamming2 } ; if ($?) { .\hamming2 }

Press 1 for Even Parity [OR]    Press 2 for Odd Parity :
2
----------
Enter 4 data bits - D7 D6 D5 D3 :
Enter the value of D7 : 1 1 1 1
Enter the value of D6 : Enter the value of D5 : Enter the value of D4 :
-----------------------Sender----------------------------
The data bits entered are : 1 1 1 1
Value of P1 : 0
Value of P2 : 0
Value of P4 : 0
The Hamming code is as follows :

D7 D6 D5 P4 D3 P2 P1
 1  1  1  0  1  0  0
-----------------------------------------------------------------

------------------------Reciever-----------------------------------

Enter Choice for Correction of Even Parity or Odd Parity :

Press 1 for Even Parity [OR]    Press 2 for Odd Parity : 2

Enter 7 bit error stream : 1 1 1 1 1 0 0

Entered Bit Stream :  1  1  1  1  1  0  0
Error at location : 4
-----------------------------------------------------------------


Corrected Code : 1110100
The Decoded data is : 1 1 1 1
-----------------------------------------------------------------
```

REFERENCES	• B.A. Forouzan, "Data Communications and Networking", TMH, Fourth Edition. • https://www.javatpoint.com/computer-network-error-correction