**NETWORKS AND COMMUNICATION**

**LAB ASSESSMENT 1**

Shreya Maheshwari

18BCE0167

**1. Parity Check**

**Code:**

class Solution {

public static void main(String[] args) {

// Sender side

// Even Parity is considered

int x, count = 0;

int[] arr = { 1, 0, 1, 1 };

// count total number of 1's

for (int i = 0; i < 4; i++) {

if (arr[i] == 1) {

count++;

}

}

if (count % 2 == 0) {

x = 0;

} else {

x = 1;

}

// appending into the new array

int[] newarr = new int[5];

for (int i = 0; i < 4; i++) {

newarr[i] = arr[i];

}

newarr[4] = x;

System.out.println("The parity bit is: " + x);

System.out.print("The codeword is:");

for (int i = 0; i < 5; i++) {

System.out.print(newarr[i]);

}

System.out.println();

// Receiver Side

// count the total number of 1's

int counter = 0;

for (int i = 0; i < 4; i++) {

if (newarr[i] == 1) {

counter++;

}

}

if (counter % 2 == 0) {

System.out.println("No error occurred: The data sent is accepted");

} else {

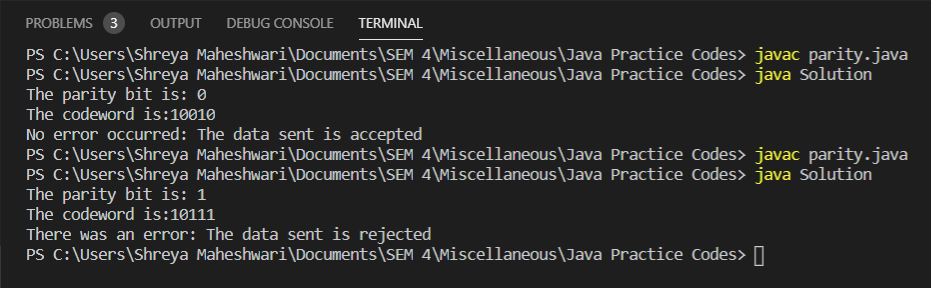
System.out.println("There was an error: The data sent is rejected");

}

}

}

**Output:**



**2. Hamming Code**

**Code:**

class HammingCode {

static void print(int ar[]) {

for (int i = 1; i < ar.length; i++) {

System.out.print(ar[i]);

}

System.out.println();

}

static int[] calculation(int[] ar, int r) {

for (int i = 0; i < r; i++) {

int x = (int) Math.pow(2, i);

for (int j = 1; j < ar.length; j++) {

if (((j >> i) & 1) == 1) {

if (x != j)

ar[x] = ar[x] ^ ar[j];

}

}

System.out.println("r" + x + " = " + ar[x]);

}

return ar;

}

static int[] generateCode(String str, int M, int r) {

int[] ar = new int[r + M + 1];

int j = 0;

for (int i = 1; i < ar.length; i++) {

if ((Math.ceil(Math.log(i) / Math.log(2)) - Math.floor(Math.log(i) / Math.log(2))) == 0) {

ar[i] = 0;

} else {

// codeword[i] = dataword[j]

ar[i] = (int) (str.charAt(j) - '0');

j++;

}

}

return ar;

}

public static void main(String[] args) {

String str = "0101";

int M = str.length();

int r = 1;

while (Math.pow(2, r) < (M + r + 1)) {

r++;

}

int[] ar = generateCode(str, M, r);

System.out.println("Generated hamming code ");

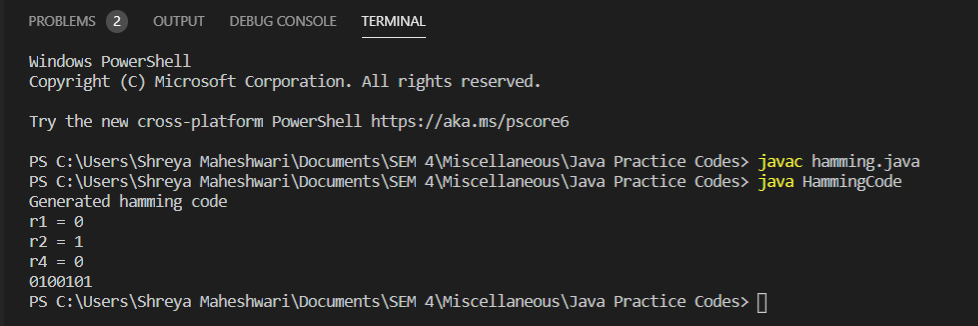
ar = calculation(ar, r);

print(ar);

}

}

**Output:**



**3. Cyclic Redundancy Check**

**Code:**

import java.util.\*;

class CRC{

public static void main(String sap[])

{

Scanner sc = new Scanner(System.in);

int i,j,flag=0;

System.out.print("\nEnter dividend: ");

String dividend = sc.next();

System.out.print("\nEnter divisor: ");

String divisor = sc.next();

int dividend\_length = dividend.length();

int divisor\_length = divisor.length();

int dividend\_array[] = new int[dividend\_length + divisor\_length - 1];

int send\_arr[] = new int[dividend\_length + divisor\_length - 1];

int divisor\_array[] = new int[divisor\_length];

for(i=0;i<dividend\_length;i++)

{

dividend\_array[i] = Integer.parseInt(""+dividend.charAt(i));

send\_arr[i] = dividend\_array[i];

}

for(i=dividend\_length;i<dividend\_array.length;i++)

{

dividend\_array[i] = 0;

}

for(i=0;i<divisor\_length;i++)

{

divisor\_array[i] = Integer.parseInt(""+divisor.charAt(i));

}

int temp[] = new int[divisor\_length];

for(i=0;i<dividend\_array.length;i++)

{

if(dividend\_array[i] == 1)

{

for(j=0;j<divisor\_length;j++)

{

if(i+j>dividend\_array.length-1)

{

flag=1;

break;

}

else

{

temp[j] = dividend\_array[i+j];

}

}

if(flag==0)

{

for(j=0;j<divisor\_length;j++)

{

if(temp[j] == divisor\_array[j])

{

dividend\_array[i+j] = 0;

}

else

{

dividend\_array[i+j] = 1;

}

}

}

}

}

for(i=dividend\_length;i<dividend\_array.length;i++)

{

send\_arr[i] = dividend\_array[i];

}

System.out.print("\nMessage to be transmitted is: ");

for(i=0;i<send\_arr.length;i++)

{

System.out.print(send\_arr[i]);

}

}

}

**Output:**

