**EXPERIMENT NO. 4**

**To implement Cyclic Redundancy Check using Java**

**Aim:**

To implement Cyclic Redundancy Check for error detection.

1. Creating a class.
2. Implementing CRC algorithm to check error in the dataword.

**Requirements:**

Ubuntu operating system/ Windows operating system, Java

**Algorithm:**

1. Consider the sequence of data bits consisting of 1’s and 0’s. (Prompt user to enter number of data bits and actual data bits)
2. Consider a generator polynomial. (Prompt user to enter size of generator and actual generator bits)
3. Append n (size of generator word -1) 0’s to the data word. E.g. If generator word is 5-bit long then we have to append 4-zeros to the data word.
4. Divide appended data word by the generator. This is a binary division.
5. The remainder obtained after the division in step 4 is the n bit CRC.
6. This CRC will replace the n 0’s appended to the data word in step 3, to get the code word to be transmitted.
7. Display the reminder bits and final code word.
8. For receiver implementation, ask user to input received bit sequence.
9. Divide the received sequence by the generator. This is again a binary division.
10. If reminder is zero, received string has no error. If reminder is non zero, received string has error.

**Procedure:**

1. **Solve the CRC examples for which program output is shown.** *(To be solved on journal sheets)*
2. Open Notepad/gedit.
3. Type the source code and save it with classname.java.
4. Open Command Prompt(for Windows)/Terminal(for Ubuntu) and run the code using following:
5. To Compile - javac classname.java
6. To Run - java classname
7. Observe output in the terminal / cmd window.

**Program output and Observations:**

1. Attach complete CRC program with appropriate comments.
2. Produce output of program for at least two examples. (Examples that are solved on journals sheets)

Choose two examples out of following:

1. A bit stream 10011101 is transmitted using CRC. The generator bits are 1001. Find the transmitted string. Suppose the third bit from left is inverted during transmission, show that this error is detected at the receiver end. (Ans: 10011101 100)
2. Consider an error detecting CRC with generator word 10101. Compute the transmitted bit sequence for data bits 1101101. (Ans: 1101101 1011). If the received bit string is 110011001100, is it acceptable?
3. Find transmitted frame if the data is 1101011011 and generator is 10011. (Ans: 1101011011 1110). Assume one bit error in the frame transmission and show that it is detected by CRC.
4. Generate the CRC code for data word 110010101. The divisor is 10101. At the receiver side, detect the error in following codewords: 110010101 1011 and 110010010 1011.

**Conclusion:**

CRC is a systematic cyclic error detection method, where data word and redundancy bits can be identified separately. It is used widely in computer networks especially in Ethernet because of its ease of implementation. CRC is mainly used with backward error correction method.

**Post Experimental Exercise:** *(To be written on journal sheets)*

1. Why is error detection and correction required?
2. What are different types of error detection and correction methods?
3. State advantages and disadvantages of CRC.

*~PROGRAM & OUTPUT*  **RIYA INDAP,44**

*import java.util.Scanner;*

*public class Main {*

*public static boolean[] rem = new boolean[100];*

*public static void xorOper(boolean[] temp, boolean[] g, int fs, int gs) {*

*int j;*

*int k;*

*int i;*

*for (i = 0; i < fs; i++) {*

*j = 0;*

*k = i;*

*// check whether it is divisible or not*

*if ((temp[k] == true && g[k] == false) || (temp[k] == true && g[k] == true)) {*

*for (j = 0, k = i; j < gs; j++, k++) {*

*if ((temp[k] == true && g[j] == true) || (temp[k] == false && g[j] == false))*

*temp[k] = false;*

*else*

*temp[k] = true;*

*}} } }*

*public static void main(String[] args) {*

*int i;*

*int j;*

*// Get Frame*

*int fs;*

*System.out.print("\n Enter the number of data bits: ");*

*Scanner sc = new Scanner(System.in);*

*fs = sc.nextInt();*

*boolean[] f = new boolean[100];*

*System.out.print("\n Enter data bits:");*

*for (i = 0; i < fs; i++) {*

*int temp = sc.nextInt();*

*if (temp == 1)*

*f[i] = true;*

*else*

*f[i] = false;*

*}*

*// Get Generator*

*System.out.print("\n Enter no of Generator bits: ");*

*Scanner s = new Scanner(System.in);*

*int gs = s.nextInt();*

*boolean[] g = new boolean[100];*

*System.out.print("\n Enter generator bits:\n");*

*for (i = 0; i < gs; i++) {*

*int temp = sc.nextInt();*

*if (temp == 1)*

*g[i] = true;*

*else*

*g[i] = false;*

*}*

*System.out.print("\n At Sender's end=>");*

*// Append 0's*

*int rs = gs - 1;*

*System.out.print(rs);*

*for (i = fs; i < fs + rs; i++)*

*f[i] = false;*

*boolean[] temp = new boolean[100];*

*for (i = 0; i < 100; i++)*

*temp[i] = f[i];*

*System.out.print("\n The Appended data word is:");*

*for (i = 0; i < fs + rs; i++) {*

*if (temp[i] == true)*

*System.out.print(1);*

*else*

*System.out.print(0);*

*}*

*// xor operation on two bool arrays*

*xorOper(temp, g, fs, gs);*

*// remainder*

*for (i = 0, j = fs; i < rs; i++, j++)*

*rem[i] = temp[j];*

*System.out.print("\n The reminder after division of appended dataword by generator bits is:");*

*for (i = 0; i < rs; i++) {*

*if (rem[i] == true)*

*System.out.print(1);*

*else*

*System.out.print(0);*

*}*

*System.out.print("\n Transmitted code word is:");*

*boolean[] tf = new boolean[150];*

*for (i = 0; i < fs; i++)*

*tf[i] = f[i];*

*for (i = fs, j = 0; i < fs + rs; i++, j++)*

*tf[i] = rem[j];*

*for (i = 0; i < fs + rs; i++) {*

*if (tf[i] == true)*

*System.out.print(1);*

*else*

*System.out.print(0);*

*}*

*System.out.print("\n");*

*System.out.print("\n At Receiver side=>");*

*System.out.print("\n Enter the Frame size : ");*

*Scanner scat = new Scanner(System.in);*

*int fr = scat.nextInt();*

*boolean[] a = new boolean[fr];*

*if (fs + rs != fr) {*

*System.out.print("Error in Transmission ");*

*System.exit(1);*

*}*

*System.out.print("Enter the code word:");*

*for (i = 0; i < fs + rs; i++) {*

*int tem = scat.nextInt();*

*if (tem == 1)*

*a[i] = true;*

*else*

*a[i] = false;*

*}*

*for (i = 0; i < fs + rs; i++)*

*temp[i] = a[i];*

*xorOper(temp, g, fs, gs);*

*System.out.print("The reminder after dividing received codeword by generator bits is:");*

*boolean[] rrem = new boolean[100];*

*for (i = fs, j = 0; i < fs + rs; i++, j++)*

*rrem[j] = temp[i];*

*for (i = 0; i < rs; i++) {*

*if (rrem[i] == true)*

*System.out.print(1);*

*else*

*System.out.print(0);*

*}*

*int flag = 0;*

*for (i = 0; i < rs; i++) {*

*if (rrem[i] != false) {*

*flag = 1;*

*break;*

*}}*

*if (flag == 0)*

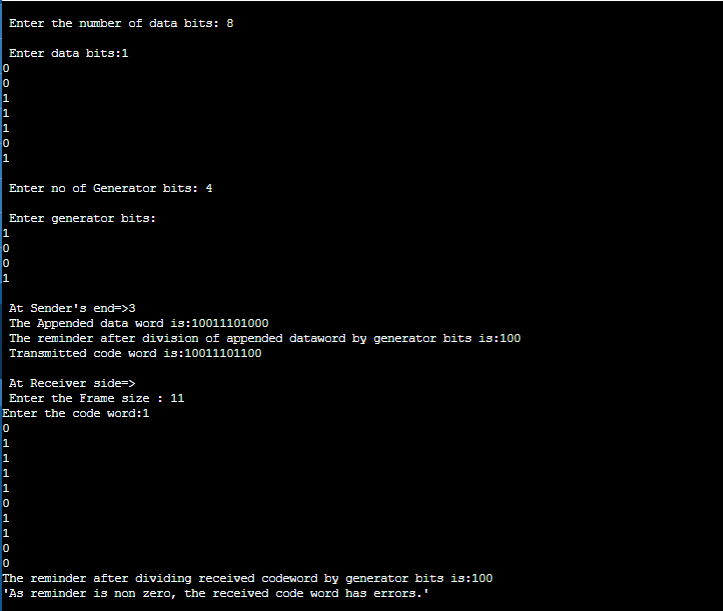
*System.out.print("\n'As reminder is zero, received code word has no errors.'");*

*else*

*System.out.print("\n'As reminder is non zero, the received code word has errors.'");*

*}}*

Q1)A bit stream 10011101 is transmitted using CRC. The generator bits are 1001. Find the transmitted string. Suppose the third bit from left is inverted during transmission, show that this error is detected at the receiver end. (Ans: 10011101 100)

****

Q2)Consider an error detecting CRC with generator word 10101. Compute the transmitted bit sequence for data bits 1101101. (Ans: 1101101 1011). If the received bit string is 110011001100, is it acceptable?

