Ex No: 5

Date:

ERROR DETECTION IN DATA FRAME TRANSMISSION BETWEEN SENDER AND RECEIVER USING CHECKSUM.

Aim:

To develop a java code, for data frame transmission, to calculate the checksum value and to check whether the data frames are received in the receiver without error, in a network, assuming that the sender sends ‘m’ frames each of ‘n’ bits.

Theory:

Checksum is the error detection method used by upper layer protocols and is considered to be more reliable than LRC, VRC and CRC. This method makes the use of Checksum Generator on Sender side and Checksum Checker on Receiver side.

At the Sender side, the data is divided into equal subunits of n bit length by the checksum generator. This bit is generally of 16-bit length. These subunits are then added together using one’s complement method. This sum is of n bits. The resultant bit is then complemented. This complemented sum which is called checksum is appended to the end of original data unit and is then transmitted to receiver.

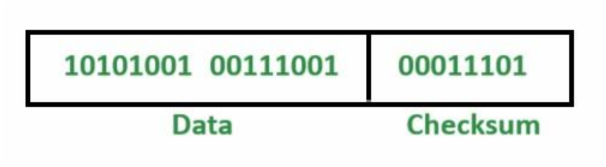
The Receiver after receiving data + checksum passes it to checksum checker. Checksum checker divides this data unit into various subunits of equal length and adds all these subunits. These subunits also contain checksum as one of the subunits. The resultant bit is then complemented. If the complemented result is zero, it means the data is error-free. If the result is non-zero it means the data contains an error and Receiver rejects it.

Example – If the data unit to be transmitted is 10101001 00111001, the following procedure is used at Sender site and Receiver site.

Sender Site:

10101001	subunit 1
00111001	subunit 2
11100010	sum (using 1s complement)
00011101	checksum (complement of sum)

Data transmitted to Receiver is:



Receiver Site :

10101001	subunit 1
00111001	subunit 2
00011101	checksum
11111111	sum
00000000	sum's complement

Result is zero, it means no error.

Algorithm:

1. Start

- Initialize the program by taking input from the user for the number of frames (m) and the number of bits per frame (n).

2. Input Data Frames

- Ask the user to input m frames, each consisting of n bits.
- Store these frames in an array frames[].

3. Checksum Calculation at Sender

- Initialize the checksum with the value of the first frame.
- For each subsequent frame:
    - Add the binary values of the current checksum and the frame using binary addition.
    - Handle overflow (if the result exceeds n bits), add the carry back to the least significant bit.
- After all frames are added, compute the one's complement of the final sum to get the checksum.

4. Transmit Frames and Checksum

- Display the calculated checksum to the user.

- Ask the user to input the received frames, including the checksum (i.e., m+1 frames in total).

5. Checksum Verification at Receiver

- Initialize the sum with the first received frame.
- For each subsequent received frame (including the checksum):
    - Add the binary values of the current sum and the frame.
    - Handle overflow if needed.
- Check if the final sum is all ones:
    - If yes, the data was received without error.
    - If no, there was an error during transmission.

6. End

Program:

```
import java.util.Scanner;


 class Main {


    public static String addBinary(String
a, String b) {

        StringBuilder result = new
StringBuilder();

        int carry = 0;

        StringBuilder aBuilder = new
StringBuilder(a);

        StringBuilder bBuilder = new
StringBuilder(b);

        while (aBuilder.length() <
bBuilder.length()) {

            aBuilder.insert(0, '0');
        }

        while (bBuilder.length() <
aBuilder.length()) {

            bBuilder.insert(0, '0');

        }

        for (int i = aBuilder.length() - 1; i
>= 0; i--) {

            int bitA = aBuilder.charAt(i) - '0';

            int bitB = bBuilder.charAt(i) -
'0';

            int sum = bitA + bitB + carry;

            result.append(sum % 2);

            carry = sum / 2;

        }


        if (carry > 0) {

            result.append(carry);

        }


        return result.reverse().toString();

    }


    public static String
onesComplement(String binary) {

        StringBuilder complement = new
StringBuilder();

        for (char bit :
binary.toCharArray()) {
```

```
            complement.append(bit == '0'
? '1' : '0');

        }

        return complement.toString();

    }


    public static boolean allOnes(String
binary) {

        for (char bit :
binary.toCharArray()) {

            if (bit != '1') {

                return false;

            }

        }

        return true;

    }


    public static void main(String[] args)
{

        Scanner scanner = new
Scanner(System.in);


        System.out.println("Enter the
number of frames: ");

        int m = scanner.nextInt();

        System.out.println("Enter the
number of bits per frame: ");

        int n = scanner.nextInt();

        scanner.nextLine();
        String[] frames = new String[m];
        System.out.println("Enter the
frames (each frame should be " + n + "
bits): ");
        for (int i = 0; i < m; i++) {
            frames[i] = scanner.nextLine();
        }

        String checksum = frames[0];
        for (int i = 1; i < m; i++) {
            checksum =
addBinary(checksum, frames[i]);

            if (checksum.length() > n) {
                checksum =
addBinary(checksum.substring(1),
"1");
            }
        }
        checksum =
onesComplement(checksum);
        System.out.println("Calculated
checksum at sender: " + checksum);

        System.out.println("Enter
received frames (including the
checksum): ");
        String[] receivedFrames = new
String[m + 1];
```

```
        for (int i = 0; i <= m; i++) {

            receivedFrames[i] =
scanner.nextLine();

        }



        String receivedSum =
receivedFrames[0];

        for (int i = 1; i <= m; i++) {

            receivedSum =
addBinary(receivedSum,
receivedFrames[i]);



            if (receivedSum.length() > n) {

                receivedSum =
addBinary(receivedSum.substring(1),
"1");

            }

        }



        if (allOnes(receivedSum)) {

            System.out.println("No error in
received data.");

        } else {

            System.out.println("Error
detected in received data.");

        }

        scanner.close();

    }

}
```

Sample Input & Output:

10011001

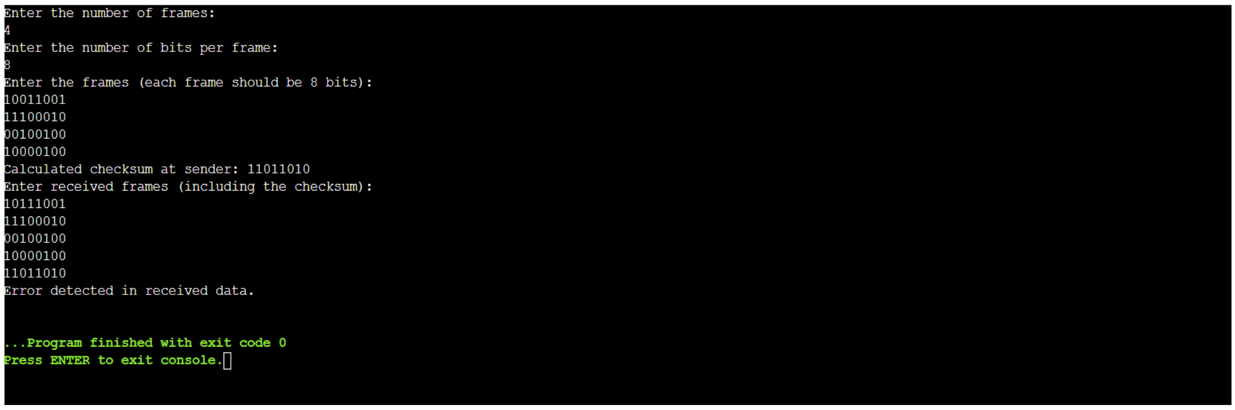
11100010

00100100

10000100

Checksum:11011010

Screenshot of output:



```
Enter the number of frames:
4
Enter the number of bits per frame:
8
Enter the frames (each frame should be 8 bits):
10011001
11100010
00100100
10000100
Calculated checksum at sender: 11011010
Enter received frames (including the checksum):
10111001
11100010
00100100
10000100
11011010
Error detected in received data.


...Program finished with exit code 0
Press ENTER to exit console.
```

Result:

Thus the Java code to calculate the checksum value at sender side, and to check whether the data frames are received in the receiver end without error, for a data frame transmission in a network, where sender sends ‘m’ frames each of ‘n’ bits, was developed and executed successfully in IntelliJ Idea Java IDE.