Ex No: 4

Date:

BITSTUFFING AND UNSTUFFING IN DATA FRAME TRANSMISSION OF HDLC BETWEEN SENDER AND RECEIVER.

Aim:

To implement bitstuffing and unstuffing in a data frame transmission of HDLC between the sender and receiver.

Theory:

Bitstuffing:

Bit Stuffing is a process of inserting an extra bit as **0**, once the frame sequence encountered **5** consecutive **1's**. Given an array, **arr[]** of size **N** consisting of **0's** and **1's,** the task is to return an array after the bit stuffing.

Examples:

Input: *N = 6, arr[] = {1, 1, 1, 1, 1, 1}*
Output: *1111101*
Explanation: *During the traversal of the array, 5 consecutive 1's are encountered after the 4th index of the given array. Hence, a zero bit has been inserted into the stuffed array after the 4th index*

Input: *N = 6, arr[] = {1, 0, 1, 0, 1, 0}*
Output: *101010*

Bit Unstuffing:

Bit Destuffing or **Bit Unstuffing** is a process of undoing the changes in the array made during the bit stuffing process i.e, removing the extra **0** bit after encountering **5** consecutive **1's**.

Examples:

Input: *N = 7, arr[] = {1, 1, 1, 1, 1, 0, 1}*
Output: *111111*

Explanation: *During the traversal of the array, 5 consecutive 1's are encountered after the 4th index of the given array. Hence, the next 0 bit must be removed to destuffed array.*

Input: *N = 6, arr[] = {1, 0, 1, 0, 1, 0}*
Output: *101010*

Algorithm:

Bitstuffing:

Input*:* A binary string (sequence of '1's and '0's).

Initialize*:*

- A counter count = 0 to track consecutive 1s.
- An empty string stuffedData to store the final stuffed binary sequence.

Traverse the input binary string one bit at a time:

- For each bit:
    1. Append the current bit (0 or 1) to stuffedData.
    2. If the bit is 1, increment count.
        - If count becomes 5 (i.e., five consecutive 1s have been encountered), append a 0 to stuffedData and reset count to 0.
    3. If the bit is 0, reset count to 0.

Output*:* The stuffed binary sequence as stuffedData.

Bit Unstuffing:

Input: A binary string with bit stuffing applied (sequence of '1's and '0's).

Initialize:

- A counter count = 0 to track consecutive 1s.
- An empty string unstuffedData to store the final unstuffed binary sequence.

Traverse the stuffed binary string one bit at a time:

- For each bit:
    1. If the bit is 1, append it to unstuffedData and increment count.

- If count becomes 5, skip the next bit (which is the stuffed 0), then reset count to 0.

2. If the bit is 0, append it to unstuffedData and reset count to 0.

Output: The unstuffed binary sequence as unstuffedData.

Program:

```
import java.util.*;

class Main

{

    public static void main(String args[])

    {

        Scanner g=new Scanner(System.in);

        System.out.println("Enter the binary data:");

        String s=g.next();

        int count=0;

        int index=0;
```

```
ArrayList<Character>obj=new ArrayList<>();

for(int i=0;i<s.length();i++)

{

    if(s.charAt(i)=='1')

    {

    count++;

    }

    else

    count=0;



    if(count==5)

    {

        if(i<s.length())

        obj.add(s.charAt(i));
```

```
            obj.add('0');

            index=i+1;

            continue;

        }

        obj.add(s.charAt(i));

    }



    System.out.println("Input string: "+s);

    System.out.print("Stuffed data: ");

    for(char v:obj)

    System.out.print(v);

    System.out.println();

    obj.remove(index);

    System.out.print("UnStuffed data: ");
```

```
        for(char v:obj)

        System.out.print(v);

        System.out.println();}}
```

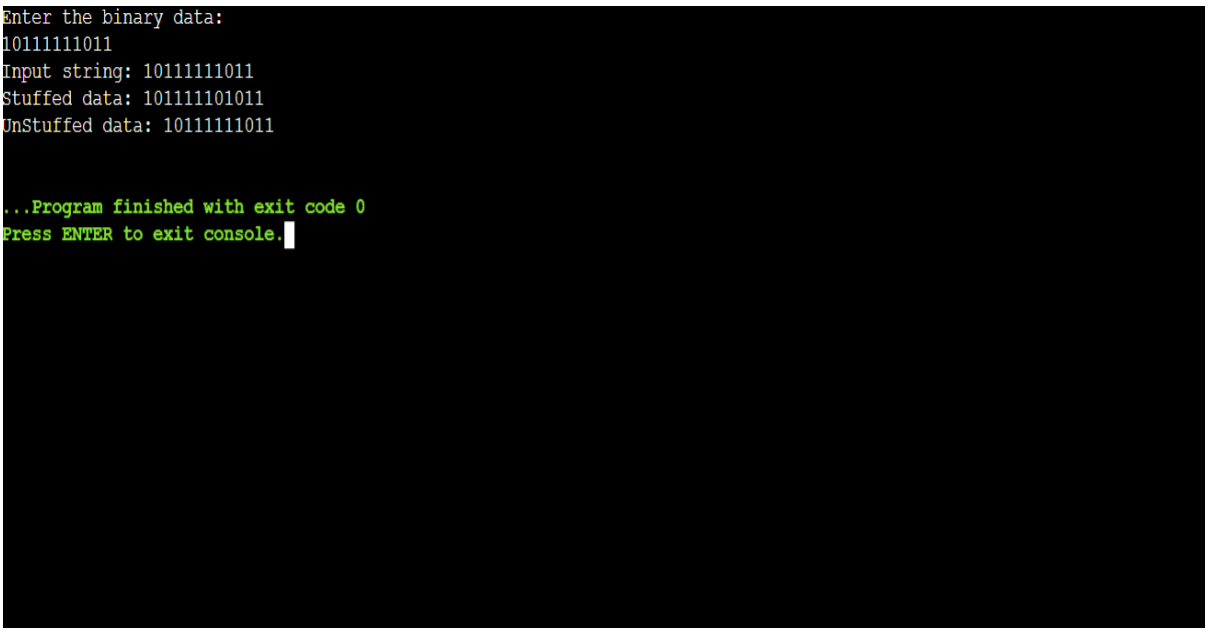
Sample Input & Output:

Enter the binary data: 10111111011

Input String: 10111111011

Stuffed Data: 101111101011 Unstuffed Data: 10111111011

Screenshot of output:



Result:

Thus the Bitstuffing and unstuffing in data frame transmission, between the sender and receiver, was executed successfully, in IntelliJ Idea Java IDE.