

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Spring,Year:2025),B.Sc.in CSE (Day)**

**LAB REPORT NO - 01**

**Course Title: Data Communication**

**Course Code: CSE307 Section:223-D1**

**Lab Experiment Name : Implementing Byte (Character) Stuffing**

**and De-stuffing**

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | **MD.SHAJALAL** | **223002088** |

**Lab Date : 24 - 02 - 2025**

**Submission Date : 03 – 03 - 2025**

**Course Teacher’s Name : Md.Samin Hossain Utsho**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |

1. **TITLE OF THE LAB REPOT EXPERIMENT**

Implementing Byte (Character) Stuffing and De-stuffing

1. **Objective :**

The objective of this experiment is to implement Byte (Character) Stuffing and De-Stuffing in a programming language (C) and demonstrate how the technique ensures reliable data transmission.

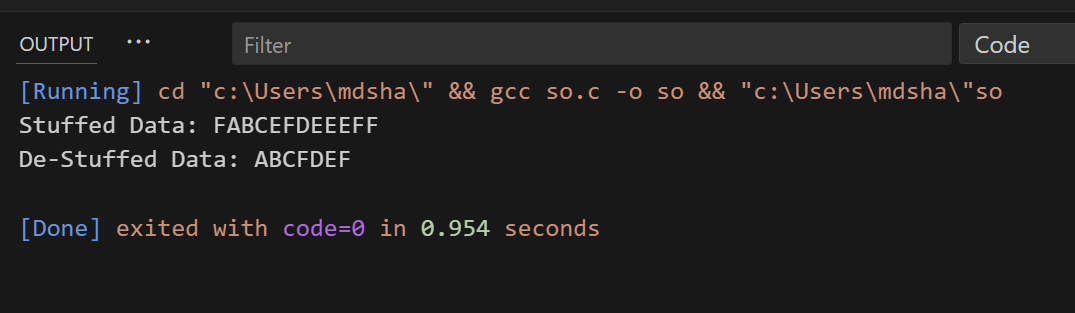
### **3.Byte Stuffing Algorithm:**

1. Define a **flag** character that marks the start and end of a frame.
2. Define an **escape** character to be used for stuffing.
3. Traverse the data string:
   * If a character matches the flag or escape character, prepend it with the escape character.
   * Otherwise, append it normally.
4. Add the flag at the beginning and end of the frame.
5. Return the stuffed data.

### **4.Byte De-Stuffing Algorithm:**

1. Ensure the received frame starts and ends with the flag character.
2. Remove the flag characters from the start and end.
3. Traverse the data:
   * If an escape character is encountered, skip it and append the next character.
   * Otherwise, append the character normally.
4. Return the de-stuffed data.
5. **Implementation (C Code) :**
6. #include <stdio.h>
7. #include <string.h>
8. #define FLAG 'F'
9. #define ESC 'E'
10. void byte\_stuffing(char \*data, char \*stuffed) {
11. int j = 0;
12. stuffed[j++] = FLAG;
13. for (int i = 0; i < strlen(data); i++) {
14. if (data[i] == FLAG || data[i] == ESC) {
15. stuffed[j++] = ESC;
16. }
17. stuffed[j++] = data[i];
18. }
19. stuffed[j++] = FLAG;
20. stuffed[j] = '\0';
21. }
22. void byte\_de\_stuffing(char \*stuffed, char \*destuffed) {
23. int j = 0;
24. for (int i = 1; i < strlen(stuffed) - 1; i++) {
25. if (stuffed[i] == ESC) {
26. i++;
27. }
28. destuffed[j++] = stuffed[i];
29. }
30. destuffed[j] = '\0';
31. }
32. int main() {
33. char data[] = "ABCFDEF";
34. char stuffed[50], destuffed[50];
36. byte\_stuffing(data, stuffed);
37. printf("Stuffed Data: %s\n", stuffed);
39. byte\_de\_stuffing(stuffed, destuffed);
40. printf("De-Stuffed Data: %s\n", destuffed);
42. return 0;
43. }

### **6.Program Output:**



**6. Conclusion** : Byte Stuffing and De-Stuffing are essential techniques in data transmission to handle special control characters within a message. This experiment successfully implemented both stuffing and de-stuffing in C, ensuring that data integrity is maintained during transmission. The output confirms the correctness of the algorithm.