# TYPES OF FRAMING

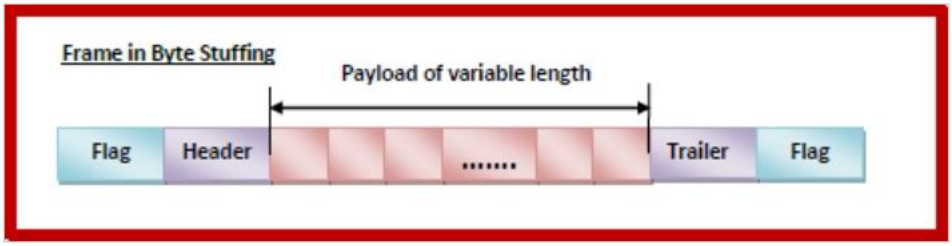
1. Fixed-size framing.

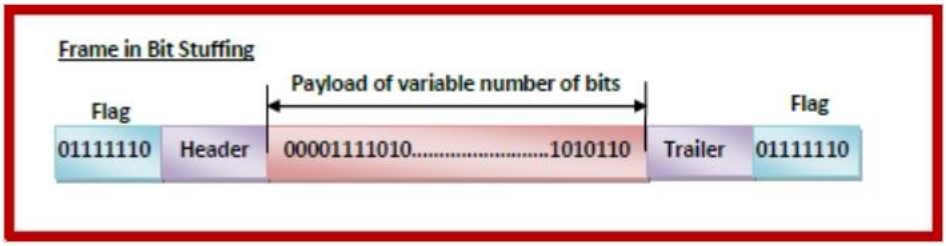
- ★ Here the size of the frame is fixed and so the frame length acts as delimiter of the frame.
- ★ Consequently, it does not require additional boundary bits to identify the start and end of the frame.

2. Variable-size framing.

- ★ Here, the size of each frame to be transmitted may be different.
- ★ So additional mechanisms are kept to mark the end of one frame and the beginning of the next frame.

# VARIOUS FRAMING APPROACHES

# 1. Fixed Framing

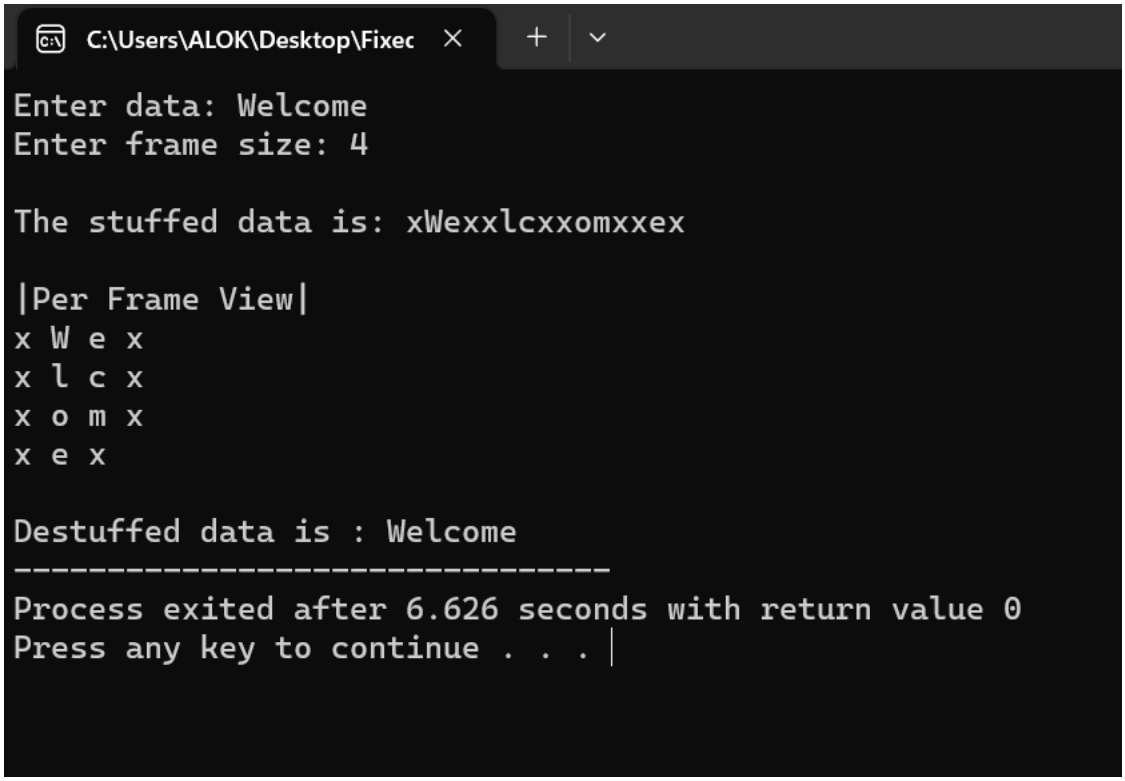
## Program

```
// Byte Stuffing With Fixed Length of the Frame
#include<stdio.h>
#include<string.h>

int main(){
	char OriginalData[50],StuffedData[50],DeStuffedData[50];
	int i,j,k,count,FrameSize,DataSize,DataPerFrame,NumberOfFrames,TotalSize,check=0;
	char flag='x';

	printf("Enter data: ");
	scanf("%s",OriginalData);

	DataSize=strlen(OriginalData);

	printf("Enter frame size: ");
	scanf("%d",&FrameSize);

	DataPerFrame=FrameSize-2; //each frame contains 2 flag bits

	NumberOfFrames=(DataSize / DataPerFrame)+(DataSize % DataPerFrame != 0); // to get division ceil
	// z = x / y + ( x % y != 0 ); z equals ceiling of (x/y)

	TotalSize=DataSize+(NumberOfFrames*2);

	count=1; // 'count' is used to count the no. of frames added
	j=0;	// 'j' is used to navigate the index of Original Data
	i=0;	// 'i' is used to navigate the index of Stuffed Data
```

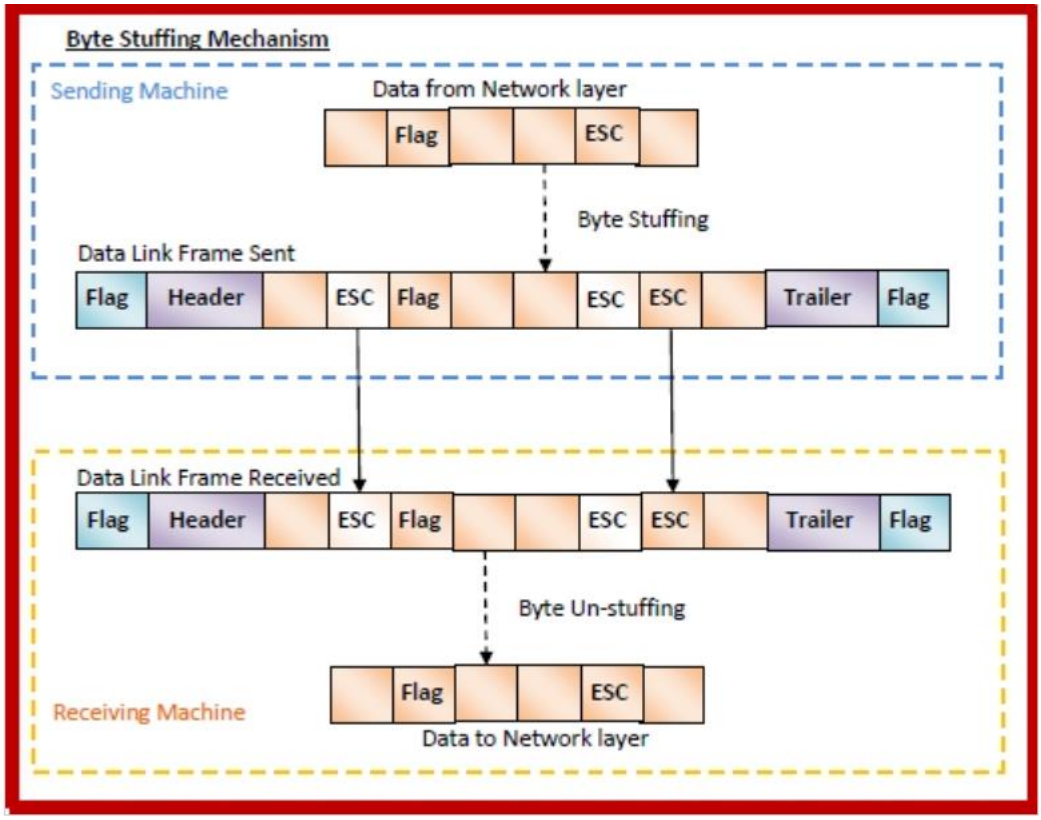
```
for(i=0;i<TotalSize-1;i++){
        if(i%FrameSize==0){ // To add the flag at start of every Frame
                StuffedData[i]=flag;
        }
        else if(i==(FrameSize*count)-1){ // To add the flag at the end of every Frame
                StuffedData[i]=flag;
                count++;
        }
        else{
                StuffedData[i]=OriginalData[j];
                j++;
        }
}
StuffedData[TotalSize-1]=flag; // Adding flag to the last position of last frame
StuffedData[TotalSize]='\0';

//Displaying the StuffedData as Sent by the Sender
printf("\nThe stuffed data is: %s \n",StuffedData);
printf("\n|Per Frame View|\n");
for(i=0;i<TotalSize;i++){
        printf("%c ",StuffedData[i]);
        if((i+1)%FrameSize==0){
                printf("\n");
        }
}

// Destuffing

k=0; // 'k' is used to navigate the index of DeStuffed Data
for(i=0;i<TotalSize;i++){
        if(StuffedData[i]!='x'){
```

```
                    DeStuffedData[k]=StuffedData[i];
                    k++;
                }
        }
        DeStuffedData[k]='\0';

        //Displaying the DeStuffedData as Received by the Reciever
        printf("\n\nDestuffed data is : %s",DeStuffedData);

        return 0;
}
```

# OUTPUT



# 2.Variable Framing ( Byte Stuffing)



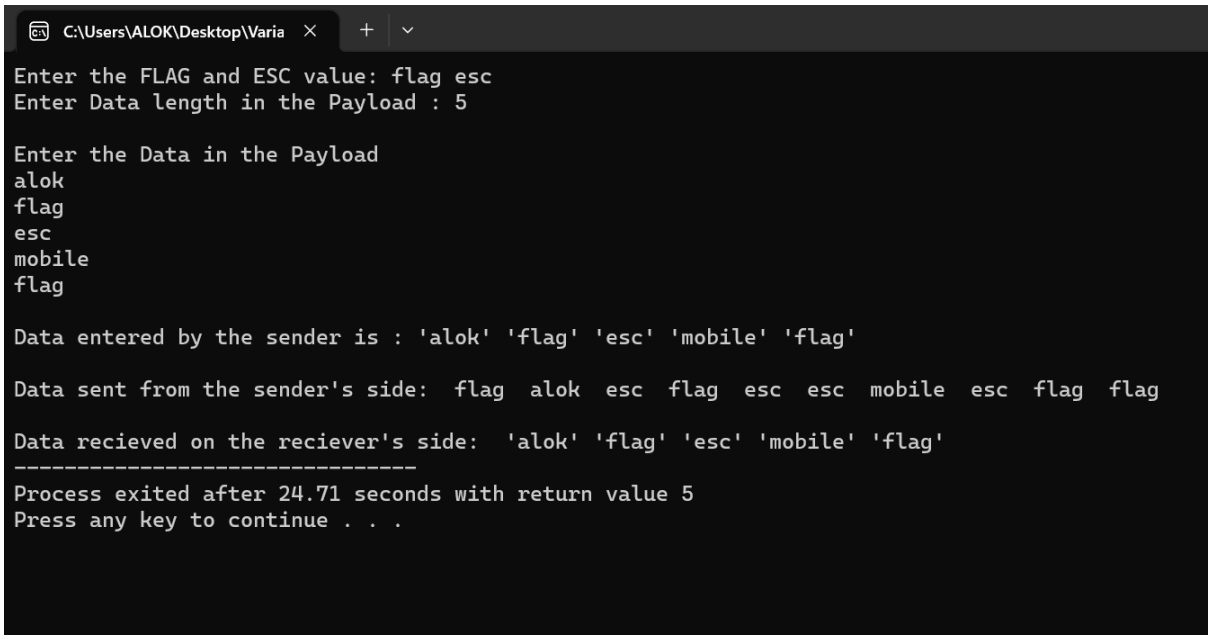

## Program

```
// Byte oriented Variable Size framing
#include <stdio.h>
#include <string.h>
void main()
{
        int i,j,k,n,m;
   char StuffedData[50][50], OriginalData[50][50], DeStuffedData[50][50];

   char flag[10],esc[10];
   printf("Enter the FLAG and ESC value: ");
   scanf("%s%s",flag,esc);
```

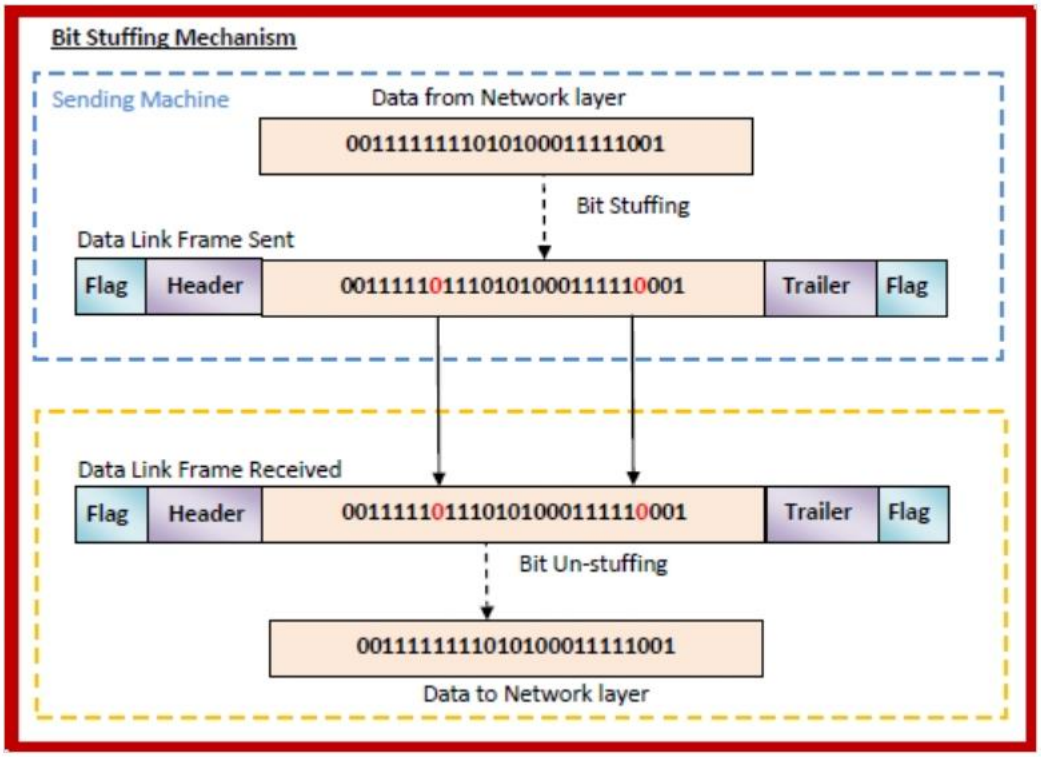
```
   printf("Enter Data length in the Payload : ");
   scanf("%d", &n);

   printf("\nEnter the Data in the Payload\n");
   for (i = 0; i <n; i++) {
      scanf("%s",OriginalData[i]);
   }

   printf("\nData entered by the sender is : ");
   for (i = 0; i <n; i++){
      printf("'%s' ",OriginalData[i]);
   }

         i=0; //    'i' is used to navigate the index for OriginalData
   j=0; // 'j' is used to navigate the index for StuffedData

   strcpy(StuffedData[j++],flag); // Adding the FLAG in the begining of Frame

   for (i = 0; i < n; i++){
      if (strcmp(OriginalData[i], flag) != 0 && strcmp(OriginalData[i], esc) != 0){
         strcpy(StuffedData[j++], OriginalData[i]);
      }
      else{
         strcpy(StuffedData[j++], esc); // Adding ESC sequence before FLAG and ESC value in
DATA
         strcpy(StuffedData[j++], OriginalData[i]);
      }
   }
   strcpy(StuffedData[j++], flag); // Adding the FLAG in the End of Frame
   m=j;  // Storing the size of StuffedData in 'm'


   printf("\n\nData sent from the sender's side:  ");
```

```
    for (i = 0; i < m; i++)
    {
        printf("%s  ", StuffedData[i]);
    }

    // destuffing

    j=1; // 'j' is used to navigate the index for StuffedData
    k=0; // 'k' is used to navigate the index for DeStuffedData

    for(j=1;j<(m-1);j++){
            if(strcmp(StuffedData[j],esc)==0){
                    j++;
                    }
                    strcpy(DeStuffedData[k++], StuffedData[j]);
            }

    printf("\n\nData recieved on the reciever's side:  ");

    for (i = 0; i < k; i++){
        printf("'%s' ", DeStuffedData[i]);
    }
}
```

# OUTPUT


C:\Users\ALOK\Desktop\Varia
Enter the FLAG and ESC value: flag esc
Enter Data length in the Payload : 5
Enter the Data in the Payload
alok
flag
esc
mobile
flag
Data entered by the sender is : 'alok' 'flag' 'esc' 'mobile' 'flag'
Data sent from the sender's side: flag alok esc flag esc esc mobile esc flag flag
Data recieved on the reciever's side: 'alok' 'flag' 'esc' 'mobile' 'flag'
Process exited after 24.71 seconds with return value 5
Press any key to continue . . .

# 3.Variable Framing (BIT Stuffing)



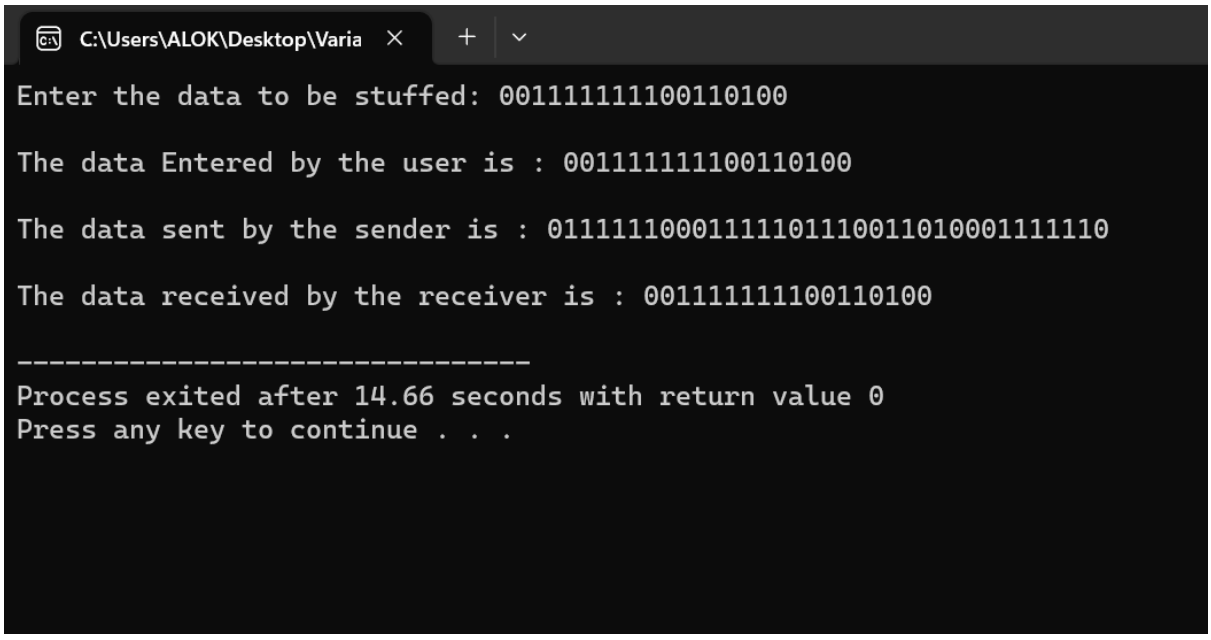

## Program

```
// bit stuffing and destuffing
#include<stdio.h>
#include<string.h>
int main(){
        char OriginalData[100],StuffedData[100],DeStuffedData[100],flag[9]="01111110";
        int i,j,k,count,ODlen,SDlen,len;

        printf("Enter the data to be stuffed: ");
        gets(OriginalData);
        ODlen=strlen(OriginalData);

        strcpy(StuffedData,flag);   // ADD flag in the beginning of the stuffed Data
        //i=0;  'i' is used to navigate the index of OriginalData.
        //j=8;  'j' is used to navigate the index of StuffedData.
```

```
//                  the value of j starts from 8 because 8 bit flag is added in it at the beginning.
count=0; // 'count' is used to count the number of consecutive 1's.
for(i=0,j=8;i<ODlen;i++){
        if(OriginalData[i]=='1'){
                count++;
        }
        else{
                count = 0;
        }
        StuffedData[j++]=OriginalData[i];
        if(count == 5){
                StuffedData[j++] = '0';
                count = 0;
        }
}
StuffedData[j] = '\0';
strcat( StuffedData, flag); // ADD flag in the end of the stuffed Data

printf("\nThe data Entered by the user is : %s\n\n",OriginalData);
printf("The data sent by the sender is : %s \n",StuffedData);

//DeStuffing
SDlen=strlen(StuffedData);
len=(SDlen-8);
count=0;
//j=8; and j<(SDlen-8)  'j' is used to navigate the index of StuffedData.
//                                                  the value of j starts from 8 and ends 8
charaters before as to discard the flag bits.
//k=0;    'k' is used to navigate the index of DeStuffedData.
for(k=0,j=8;j<(SDlen-8);j++){
        if(StuffedData[j]=='1'){
                count++;
        }
```

```
            else{
                count=0;
            }
            DeStuffedData[k++]=StuffedData[j];
            if(count == 5){
                DeStuffedData[k]=StuffedData[j++];
                count=0;
            }
        }
        DeStuffedData[k]='\0';

        printf("\nThe data received by the receiver is : %s \n",DeStuffedData);
        return 0;
}
```

# OUTPUT



# REFERENCES


1. COMPUTER NETWORKS ( Neso Academy) playlist ( video no. 36 to 43) https://www.youtube.com/playlist?list=PLBlnK6fEyqRgMCUAG0XRw78UA8qnv6jEx

2. https://www.tutorialspoint.com/difference-between-byte-stuffing-and-bit-stuffing