

ASSIGNMENT-II

1) STUFFING

```
def bit_stuffing(data: str) -> str:
    """
    Perform bit stuffing on the input binary data.
    If five consecutive '1's are found, a '0' is inserted after them.
    """
    stuffed_data = ""
    count = 0
    for bit in data:
        if bit == '1':
            count += 1
        else:
            count = 0

        stuffed_data += bit

        if count == 5: # Insert '0' after five consecutive '1's
            stuffed_data += '0'
            count = 0

    return stuffed_data


def add_flags(data: str, flag: str = "011111") -> str:
    """
    Add flag bytes at the beginning and end of the frame.
    """
    return flag + data + flag
```

```
# Example usage

original_data = "01111110110111111011111010"

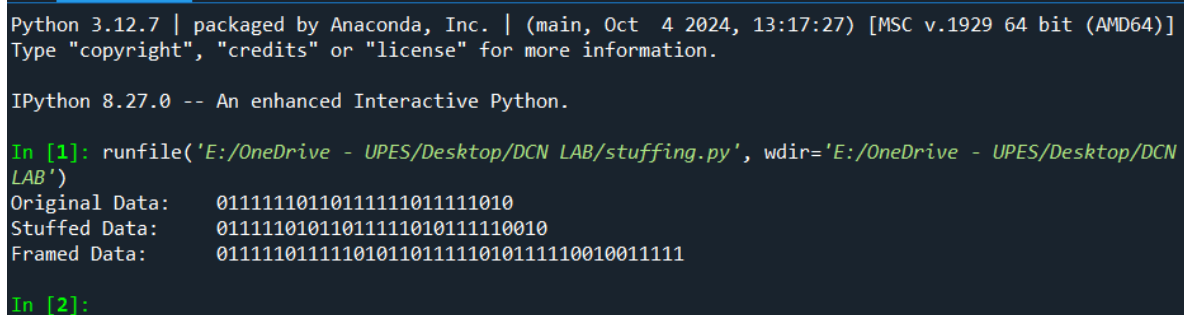
stuffed_data = bit_stuffing(original_data)

framed_data = add_flags(stuffed_data)


print("Original Data: ", original_data)

print("Stuffed Data: ", stuffed_data)

print("Framed Data: ", framed_data)
```



```
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct 4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.27.0 -- An enhanced Interactive Python.

In [1]: runfile('E:/OneDrive - UPES/Desktop/DCN LAB/stuffing.py', wdir='E:/OneDrive - UPES/Desktop/DCN
LAB')
Original Data:      01111110110111111011111010
Stuffed Data:       01111101011011111010111110010
Framed Data:        01111101111101011011111010111110010011111

In [2]:
```

2) DESTUFFING

```
def bit_destuffing(stuffed_data, flag="011111"):

    # Remove flag at the beginning and end (if present)

    if stuffed_data.startswith(flag):

        stuffed_data = stuffed_data[len(flag):]

    if stuffed_data.endswith(flag):

        stuffed_data = stuffed_data[:-len(flag)]

    destuffed_data = ""
```

```
count = 0
```

```
for bit in stuffed_data:
```

```
    destuffed_data += bit
```

```
    if bit == '1':
```

```
        count += 1
```

```
    else:
```

```
        count = 0
```

```
    # If five consecutive '1's are found, skip the next '0'
```

```
    if count == 5:
```

```
        count = 0 # Reset count after skipping
```

```
        continue # Skip the next bit
```

```
return destuffed_data
```

```
# Example usage
```

```
stuffed_bitstream = "011111010110111101011110010" # Example stuffed data with flags
```

```
print("Destuffed Output:", bit_destuffing(stuffed_bitstream))
```

```
Console 1/A X
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct 4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.27.0 -- An enhanced Interactive Python.

In [1]: runfile('E:/OneDrive - UPES/Desktop/DCN LAB/stuffing.py', wdir='E:/OneDrive - UPES/Desktop/DCN LAB')
Original Data:      01111110110111111011111010
Stuffed Data:       0111111010110111111010111110010
Framed Data:        0111111011110101101111101011111001001111

In [2]: runfile('E:/OneDrive - UPES/Desktop/DCN LAB/destuffing.py', wdir='E:/OneDrive - UPES/Desktop/DCN LAB')
Destuffed Output: 01011011111010111110010

In [3]:
```