Aayush Shah
19BCE245
27 August 2021

Practical 3

Frame Generation

- **Write a program to read a stream of data from data file to create frames by implementing bit stuffing and character stuffing.**

- **Code :**

```
#!/usr/bin/env python3

def byte_stuff_msg(message,flag_seq,esc_seq):
     return message.replace(esc_seq,
esc_seq+esc_seq).replace(flag_seq, esc_seq+flag_seq)

def byte_unstuff_msg(message,flag_seq,esc_seq):
     return
message.replace(esc_seq+flag_seq,flag_seq).replace(esc_seq+esc
_seq, esc_seq)

def bit_stuff_msg(message,flag_seq):
     flag_seq = '011111'
     return message.replace(flag_seq, '0111110')

def bit_unstuff_msg(message,flag_seq):
     flag_seq = '011111'
     return message.replace(flag_seq+'0', flag_seq)

message =
'01111110asdfsdf0111111001111110011111110abcabcasdfasdfasdfabc'
flag_seq = '01111110'
esc_seq = 'abc'
print('Original          : ',message)
```

```
#bin_data = ' '.join(format(ord(x), 'b') for x in message)
#decoded = "".join([chr(int(binary, 2)) for binary in
bin_data.split(" ")])

bit_stuffed_msg = bit_stuff_msg(message, flag_seq)
print('BIT stuffing      : ',bit_stuffed_msg)

bit_unstuffed_msg = bit_unstuff_msg(bit_stuffed_msg, flag_seq)
print('BIT unstuffing    : ',bit_unstuffed_msg)

byte_stuffed_msg = byte_stuff_msg(message, flag_seq,esc_seq)
print('BYTE stuffing     : ',byte_stuffed_msg)

byte_unstuffed_msg = byte_unstuff_msg(byte_stuffed_msg,
flag_seq,esc_seq)
print('BYTE unstuffing   : ',byte_unstuffed_msg)

print('Are bit unstuffed, byte unstuffed and original strings
same :',bit_unstuffed_msg==byte_unstuffed_msg==message)
```

- **Execution :**