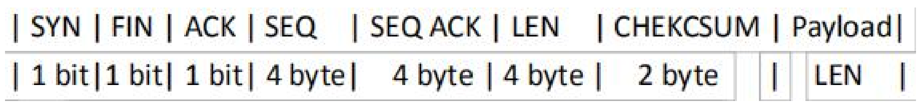
RDT Implementation and Analysis

In this Project, you need our given UDP socket class to implement RDT protocol.
Then implement an Echo Server and Client, test its function and evaluate its performance.

1. Requirement
   a) Your RDT protocol needs to ensure the reliability of data transfer. Packet loss and payload corruption might happen.
      i. To deal with packet loss, using ack and retransmission according to **GBN and SR** the textbook.
      ii. To deal with payload corruption, you need to design a checksum of your payload
   b) Your RDT protocol should be like TCP, which means it's a stream-oriented protocol, not packet-oriented.
      i. To establish a connection, you might need to do things like things in TCP:
         1. SYN
         2. SYN,ACK
         3. ACK
      ii. To close a connection, you might need to do things like things in TCP:
         1. FIN
         2. ACK
         3. FIN
         4. ACK
   c) Message head of protocol
      The message head of protocol might be like this (just for reference):



SYN	FIN	ACK	SEQ	SEQ ACK	LEN	CHEKCSUM	Payload
1 bit	1 bit	1 bit	4 byte	4 byte	4 byte	2 byte	LEN

      i. Checksum Calculation Example(just for reference)

```
def calc_checksum(payload):
    sum = 0
    for byte in payload:
        sum += byte
    sum = -(sum % 256)
    return (sum & 0xFF)
```

   d) API reference:
      i. rdt code example:

```
from udp import UDPsocket # import provided class
class socket(UDPsocket):
    def __init__():
        super(socket, self).__init__()

    def connect():
        # send syn; receive syn, ack; send ack
        # your code here
        pass

    def accept():
        # receive syn; send syn, ack; receive ack
        # your code here
        pass

    def close():
        # send fin; receive ack; receive fin; send ack
        # your code here
        pass

    def recv():
        # your code here
        pass

    def send():
        # your code here
        pass
```

ii. Server code example:

```
from rdt import socket

server = socket()
server.bind((SERVER_ADDR, SERVER_PORT))
while True:
    conn, client = server.accept()
    while True:
        data = conn.recv(2048)
        if not data: break
            conn.send(data)
        conn.close()
```

client code example:

```
from rdt import socket

client = socket()
client.connect((SERVER_ADDR, SERVER_PORT))
client.send(MESSAGE)
data = client.recv(BUFFER_SIZE)
assert data == MESSAGE
client.close()
```

e) Protocol evaluate and analysis (optional)

   i. The protocol should be flexible for users to choose GBN or SR for re-transmission.
   ii. Provide script or code to verify the working mechanism of RDT protocol
   iii. Provide scripts or codes to evaluate the performance of RDT protocol, such as packet loss and error statistics, protocol performance, effective data transmission rate, RTT statistics.
   iv. Make the conclusion based on the evaluation and analysis on the protocol.

2. Submission
   a) Source code for this project
   b) Scripts or code to evaluate and analysis the rdt protocols implemented in this project(optional)
   c) Technical report. Including the design on your rdt protocols, function and performance analysis procedure on the rdt protocols and your solution.
   d) Presentation slides.

3. Reference：
   a) GBN/SR：“computer_networking_a_top-down_approach_7th”
   b) Lua on Wireshark：https://wiki.wireshark.org/Lua/Dissectors
   c) Code performance statistics：https://zhuanlan.zhihu.com/p/83421076