Ex:NO:7
Date: 6.9.24

Aim:

write a program to implement flow control at data link layer using sliding window protocol simulate the flow of frames from one node to another.

Create a sender program with following features

1) Input window size from the user
2) Input a text message from the user
3) Consider 1 character per frame
4) create a frame with following field [frame no, Data]
5) send the frames.
6) wait for the ack from the receiver.
7) Reader a file called receiver-buffer.
8) Check ack field for the ACK number
9) If the ack number is as expected send now set of frames accordingly

Create a receiver file with

features:

1) Reader a file called sender_buffer

2) check the frame no

3) If the frame no are as expected write the appropriate ack no in the receiver buffer file. else write nack no in the receiver buffer file

Code:

```
import time
import threading
import os
Class sliding window protocol
def __init__(self, window.size, message)
self.window_size = window_size
self.frame_no = 0
self.expected_ack_no = 0
self.expected_frame_no = 0
self.sender_buffer = "sender_buffer.txt";
self.receiver_buffer = "receiver_buffer.txt";
self.sender_done = False
self.receiver_done = False
```

```
def sender(self):
    while not self.sender_done:
        frames = []
        print(f"\n--- SENDING FRAMES (window size
{self.window_size}) ---")
        for i in range(self.window_size):
            if self.frame_no < len(self.message):
                frame_data = f"[Frame no:
{self.frame_no} DATA : {self.message[self
frame]}]
                frames.append(frame_data)
                print(f"send: {frame_data}")
                self.frame_no += 1
        if frames:
            with open(self.sender_buffer, "w") as f:
                f.write("\n".join(frames) + "\n")
        if self.frame_no >= len(self.message):
            self.sender_done = True
            print("All frame sent")
            print("waiting for ack")
            time.sleep(2)
            self.wait_for_ack()
def wait_for_ack(self):
    try:
        with open(self.receiver_buffer, "r") as f:
            ack_data = f.readlines()
```

```
except FileNotFoundError:
    print(f"receiver buffer file {self.receiver_buffer} not found")
    return
for ack in ack_data:
    ack_type, ack_no = ack.strip().split(
    ack_no = int(ack_no)
    if ack_type == "ACK"
    if ack_no == self.expected_ack_no+1:
        print({ACK} ack_no } received. sending next frames")
        self.frame_expected_ack_no = ack_no
    else:
        print(f"unexpected ACK received expected {self.expected_ack_no+1}, got {ack_no} resending")
    elif ack_type = "NACK":
        print(f"NACK {ack no} received. resending frames)
        self.frame_no = self.windowsize
def receiver(self):
    while not self.receiver_done
        time.sleep(2)
    if not os.path.exists(self.sender_buffer
        print(f"sender buff...
```

```
does not exist waiting for frames...")
        continue
    with open(self.sender_buffer, "r") as f:
        frames = f.readlines()
    if not frames:
        continue
    print("\n--Receiving frames--")
    ack_to_send = []
    for frame in frames:
        frame_no, data = frame.strip().split(",")
        frame_no = int(frame_no.split(":")[1])
        data = data.split(":")[1]
        with open(self.receiver_buffer, "w") as f:
            f.write("\n".join(ack_to_send) + "\n")
        if self.expected_frame_no >= len(self.message):
            print("All frames received stopping receiver")
            time.sleep(2)
            break
    if __name__ == "__main__":
        window_size = int(input("Enter window size"))
        message = input("Enter data message")
        protocol = SlidingWindowProtocol(window_size, message)
        sender_thread = threading.Thread(target=
            protocol.receiver)
        sender_thread.start()
        receiver_thread.start()
```

sender_thread.join()
receiver_thread.join()
print("\n Transmission completed")

Output:

Enter window size: 4
Enter text message: cute
Sending frame (windowsize=4)
sent: [Frame No: 0, Data: C]
sent: [Frame No: 1, Data: U]
sent: [Frame No: 2, Data: T]
sent: [Frame No: 3, Data: E]
All frames sent
waiting for ack
unexpected Ack received expected got 4 resending
-- Receiving Frames -- --
received [frame No: 0, Data: C]
received [frame No: 1, Data: U]
received [frame No: 2, Data: T]
received [frame No: 3, Data: E]

All frames
completed.

Sender Bu
[frame No: 0
[frame No. 1,
[frame No: 2
[frame No: 3

Result:
Thus
successfully

All frames received. stopping receiver transmission completed.

Sender Buffer	Receiver Buffer
[frame NO: 0, Data : C]	Ack 1
[frame NO. 1, Data: U]	Ack : 2
[frame NO: 2, Data : T]	Ack: 3
[frame NO: 3, Data: e]	ACK: 4

Result:

Thus program e Output is verified successfully.