Ex:No:7 Sliding Window Protocol.

Date: 14.09.24

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.

Program should achieve atleast below requirements. You can make it a bidirectional program wherein receiver is sending its data frames with acknowledgement.

Student observation:

Program

sender.py

```
import time
import os
def input-window-size():
    return int(input("Enter window size: "))
def input-text-message():
    return input("Enter text Message: ")
def create-frames(text-Message):
  frames = [(i, char) for i, char in enumerate
                 (text-message)]
  frames.append((len(text-Message), 'END'))
   return frames.
```

```
def write_to_file (filename, data):
    with open (filename, 'w') as file:
      for frame in data:
           file.write (f "{frame[0]},
                          {frame[1]}\n")
def read_from_file (filename):
    if not os.path.exists (filename):
       return []
    with open (filename, 'r') as file:
       return [line.strip().split (',')
  for line in file.readlines ()]
def send_frames (frames, window_size):
      i =0
    while i< len (frames):
       window = frames [i:i + window_size]
     print (f" Sending frames: {window}")
     write_to_file ('Sender_Buffer.txt', winda
     time.sleep (3)
   receiver_buffer = read_from_file ('Receiver
                          Buffer.txt')
  if not receiver_buffer:
     print (" No Acknowledgement received yet.")
     continue
if _name_ == "__main__":
     main_sender ().
```

receiver.py

```
import random
import time
import os
def write_to_file (filename, data):
    with open ( filename, 'w') as file :
            file.write (data)
def read_from_file (filename):
    if not os.path.exists (filename):
            return []
   with   open (filename, 'r') as file :
            return [ line.strip().split(',') for line
  in   file.readlines ()]
def process_frames (frames):
      acks  =  []
   frame_seen  =  set ()
   for frame in frames:
        frame_number = int (frame [0])
           data  =  frame [1]
    if frame_number in frame_seen :
            continue
def main_receiver ():
   while True :
           time.sleep (3)
     frames  =  read_from_file ('Sender_Buffer.txt')
   if not frames :
         print ("No frames to process, waiting ...")
         continue
```

```
acks = process_frames (frames)
write_to_file ('Receiver_Buffer.txt', acks)
if any (frame[1] == 'END' for frame in frames
    print ("End of transmission received").
    break.
if __name__ == "__main__":
    main_receiver ().
```

O/P :

```
=> python sender.py
   Enter window size : 3
   Enter text message : hello
   Sending frames : [(0, 'h') (1, 'e') (2, 'l'
   Ack received for frame 0.
   Sending frames : [(3, 'l'), (4, 'o') (5, 'END')]
   Ack received for frame 1.
=> python receiver.py
   No frames to process, waiting ....
   Received Frame 0: h
   Sending ACK
   Received frame 1 : e
   Sending ACK
   Received frame 2 : l
   Sending ACK.
   Received frame 3 : l
   Sending ACK
   Received frame 4 : 0
   Sending Ack
   End of transmission
```

Result:

The program was successfully executed and the O/P is verified