SLIDING WINDOW PROTOCOL

Ex.no:7
26/09/24

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:-

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_
  i=0                                  size):
 while i<len(frames):
    window=frames[i:i+window_size]
    print(f"sending frames {window}")
    write_to_file('send_buffer.txt',
                                   window)
    time.sleep(3)
   receiver_buffer=read_from_file
                    ('Receiver_buffer.txt')
  if not receiver_buffer:
       print("No acknowledgement
                 received yet")
     continue
ack_frame=receiver_buffer[0]
ack_number, ack_type=int(ack_frame
              [0]), ack_frame[1]
 if ack_type=='ACK':
     print(f"ACK received for
           frame {ack_no}, sending
             next set of frames")
    i+=window_size
    elif ack_type = "NACK":
        print(f"NACK received for
              frame{ack_no}, resending
          frames from frame{ack_no}")

    i=ack_no

def main_sender():
  window_size = input_window_size()
  text_message = input_text_message()
  frames = Create_frames(text_message)
  send.frames(frames, window_size)
  if_name == "_main__":
    main.sender()
```

Receiver.py

```
import random
import time
Import os
defwrite_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
  print(f"Received Frame{frame_
                   number}:{data}")
  if random.choice([true,false]):
    print(f"Sending Ack for frame
                {frame_number}")
    acks.append(f"{frame_number},
                       Ack\n")
   frame_seen.add(frame_number)
else:
    print(f"Sending NACK for frame
            {frame_number}")
   acks.append(f"{frame_number},
                      NACK\n")
   break
return ''.join(acks)
```

```
def main_receiver():
  while True:
      time.sleep(2)
  frame = 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()
```

<u>Output</u>:

```
Enter Window Size: 3
Enter ~~bat~~ message : hii
Sending frames: [(0, 'h'), (1,'i'),(2,'i')]
Ack received for frame 2, sending
             next set of frame.
```

```
Sending frames: [~~(2,'i')~~, (3, 'END')]
Ack received for frame 2.
Sending next set of frames.
```

receiver.py

```
Received frame 2: END
Sending ACK for frame 2
End of transmission received.
```

[illegible] 6/9/24

RESULT:-

Thus the Program is Successfully executed and Verified.