Reg No: 220701053

**Practical 7**

**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.

**ALGORITHM:**

Initialize Frames:

* Input the window size from the user.
* Input a message to be sent as a sequence of frames.
* Create a list of frames from the message, where each frame consists of:
  + A frame number (frame\_no).
  + The data (a single character from the message).
  + An acknowledgment status (acknowledged), initially set to False.

Set Initial Variables:

* Set base to 0, representing the starting position of the sliding window.
* The window\_size determines how many frames can be sent without waiting for acknowledgments.

Loop Until All Frames are Sent:

* Send Frames:
  + Send up to window\_size frames starting from the current base.
  + Display the frame numbers and data being sent.
  + Introduce a delay (2 seconds) to simulate transmission time.
* Receive Acknowledgments:
  + Simulate acknowledgment for each frame in the window:
    - With an error probability of 20%, mark the frame as not acknowledged.
    - Otherwise, mark the frame as successfully acknowledged.
  + Display the acknowledgment status for each frame (OK for success, ERROR for failure).
  + Introduce a delay to simulate acknowledgment processing time.

Update Window Base:

* + Shift the window (base) to the next unacknowledged frame:
    - Move base forward as long as frames at base are acknowledged.
    - If base has moved to the next set of unacknowledged frames, resend the remaining frames in the window.

Repeat Until All Frames are Acknowledged:

* Continue until base reaches the end of the frame list.
* If there are still frames left unacknowledged after a complete cycle, resend them.
* Introduce a delay for retransmission.

End Protocol:

* Print a confirmation message that all frames have been sent and acknowledged.

**Code:**

import time

import random

class Frame:

def \_\_init\_\_(self, frame\_no, data):

self.frame\_no = frame\_no

self.data = data

self.acknowledged = False

def send\_frames(frames, window\_size):

print("\n--- Sending Frames ---")

for i in range(window\_size):

if i < len(frames) and not frames[i].acknowledged:

print(f"Sent Frame {frames[i].frame\_no}: {frames[i].data}")

print("Frames sent, waiting for acknowledgments...\n")

def receive\_frames(frames, window\_size):

print("\n--- Receiving Frames ---")

for i in range(window\_size):

if i < len(frames) and not frames[i].acknowledged:

if random.random() < 0.2:

print(f"Received Frame {frames[i].frame\_no}: {frames[i].data} [ERROR]")

frames[i].acknowledged = False

else:

print(f"Received Frame {frames[i].frame\_no}: {frames[i].data} [OK]")

frames[i].acknowledged = True

def sliding\_window\_protocol():

window\_size = int(input("Enter window size: "))

message = input("Enter a message to send: ")

frames = [Frame(i, message[i]) for i in range(len(message))]

base = 0

while base < len(frames):

send\_frames(frames[base:], window\_size)

time.sleep(2)

receive\_frames(frames[base:], window\_size)

while base < len(frames) and frames[base].acknowledged:

base += 1

if base < len(frames):

print("\nResending unacknowledged frames...\n")

time.sleep(2)

print("\nAll frames sent and acknowledged!")

if \_\_name\_\_ == "\_\_main\_\_":

sliding\_window\_protocol()

