```
import random
import time

TOTAL_FRAMES = 10
WINDOW_SIZE = 4
LOSS_PROBABILITY = 0.3

class Peer:
    def __init__(self, mode="GBN"):
        self.mode = mode
        self.window_size = WINDOW_SIZE
        self.total_frames = TOTAL_FRAMES
        self.sent_frames = []
        self.acknowledged_frames = []

    def send(self):
        base = 0
        next_frame = 0

        while base < self.total_frames:
            while next_frame < base + self.window_size and next_frame < self.total_frames:
                if next_frame not in self.acknowledged_frames and next_frame not in self.sent_frames:
                    print(f"Sender: Sending frame {next_frame}")
                    self.sent_frames.append(next_frame)
                    if random.random() < LOSS_PROBABILITY:
                        print(f"Sender: Frame {next_frame} lost!")
                    else:
                        self.receive(next_frame)
                next_frame += 1

            time.sleep(1)
```

```
        if self.mode == "GBN":
            if base not in self.acknowledged_frames:
                print(f"Sender: Timeout! Resending from frame {base}")
                next_frame = base
                self.sent_frames = [frame for frame in self.sent_frames if frame >= base]
            else:
                print(f"Sender: Frame {base} acknowledged. Sliding window.")
                base += 1

        elif self.mode == "SR":
            while base in self.acknowledged_frames:
                print(f"Sender: Frame {base} acknowledged. Sliding window.")
                base += 1

    def receive(self, frame):
        if self.mode == "GBN":
            if frame == len(self.acknowledged_frames):
                print(f"Receiver: Acknowledging frame {frame}")
                self.acknowledged_frames.append(frame)
        elif self.mode == "SR":
            print(f"Receiver: Acknowledging frame {frame}")
            if frame not in self.acknowledged_frames:
                self.acknowledged_frames.append(frame)

    def simulate(self):
        print(f"Starting simulation in {self.mode} mode...\n")
        self.send()
        print(f"\nAll frames sent and acknowledged in {self.mode} mode.")
        print(f"Acknowledged frames: {sorted(self.acknowledged_frames)}\n")
```

```
if __name__ == "__main__":
    random.seed(42)

    gbn_peer = Peer(mode="GBN")
    gbn_peer.simulate()

    print("\n" + "="*50 + "\n")

    sr_peer = Peer(mode="SR")
    sr_peer.simulate()
```

OUTPUT :

```
Starting simulation in GBN mode...

Sender: Sending frame 0
Receiver: Acknowledging frame 0
Sender: Sending frame 1
Frame 1 lost!
Sender: Sending frame 2
Receiver: Acknowledging frame 2
Timeout! Resending from frame 1
Sender: Resending frame 1
Receiver: Acknowledging frame 1
Sender: Sending frame 3
Receiver: Acknowledging frame 3

All frames sent and acknowledged in GBN mode.
Acknowledged frames: [0, 1, 2, 3]
```

==========

Starting simulation in SR mode...

Sender: Sending frame 0

Receiver: Acknowledging frame 0

Sender: Sending frame 1

Receiver: Acknowledging frame 1

Sender: Sending frame 2

Frame 2 lost!

Sender: Sending frame 3

Receiver: Acknowledging frame 3

Timeout! Resending frame 2

Sender: Resending frame 2

Receiver: Acknowledging frame 2

All frames sent and acknowledged in SR mode.

Acknowledged frames: [0, 1, 2, 3]