1. Describe the pros and cons of using event-driven programming.
    - Pros
        - Only executes code when events occur, saving CPU cycles and power
        - Immediate reaction to network events, timers, and user commands
        - Natural handling of multiple simultaneous events without complex threading
    - Cons
        - Difficult to trace execution flow
        - Potential for events to interfere with each other if not carefully managed
2. Flooding includes a mechanism to prevent packets from circulating indefinitely, and the TTL field provides another mechanism. What is the benefit of having both? What would happen if we only had flooding checks? What would happen if we had only TTL checks?
    - Duplicate suppression stops loops immediately, and TTL guarantees packets eventually die, even if duplicate suppression fails
    - If we only had flooding checks, packets wouldn't circulate indefinitely, but if a node's "seen" cache overflows or loses track, the same packet might get reinjected and circulate again
    - If we had only TTL checks, packets would be guaranteed to die, but multiple copies of the same packet could still propagate, causing lots of redundant traffic
3. When using the flooding protocol, what would be the total number of packets sent/received by all the nodes in the best-case situation? Worse case situation? Explain the topology and the reasoning behind each case.
    - Best Case
        - Message reaches all nodes once per node, no redundancy
        - Tree topology
            - 1 receive per node (except source) $\rightarrow N - 1$
            - 1 send per node (except leaves) $\rightarrow N - 1$
            - Total Cost: $O(N)$
        - Best case $O(N)$ is acyclic tree topology with source at root
    - Worst Case
        - Message duplicated as much as possible
        - Fully connected graph (all nodes directly connected to all others)
            - Each node sends to neighbors
            - Each node receives and discards many duplicates
            - Source sends to $N - 1$ neighbors
            - Each node sends to $N - 2$ neighbors
            - Total Cost: $O(N^2)$
        - Worst case $O(N^2)$ is complete graph
4. Using the information gathered from neighbor discovery, what would be a better way of accomplishing multi-hop communication?
    - Link State Routing
        - Distribute complete topology information using neighbor discovery as foundation
        - Each node maintains full network map for optimal path calculation
        - Best paths computed locally using algorithms like Dijkstra's

5. Describe a design decision you could have made differently given that you can change the provided skeleton code and the pros and cons compared to the decision you made.
    - Current design is reactive neighbor discovery
        - Pros
            - Energy efficient
            - Reduced network traffic
            - Scales better with network size
        - Cons
            - Initial discovery delay
            - Potential stale information during rapid topology changes
    - Alternate design is proactive neighbor discovery
        - Pros
            - Always up-to-date neighbor information
            - Faster response to topology changes
            - No discovery delay
        - Cons
            - Higher energy consumption
            - Increased network overhead