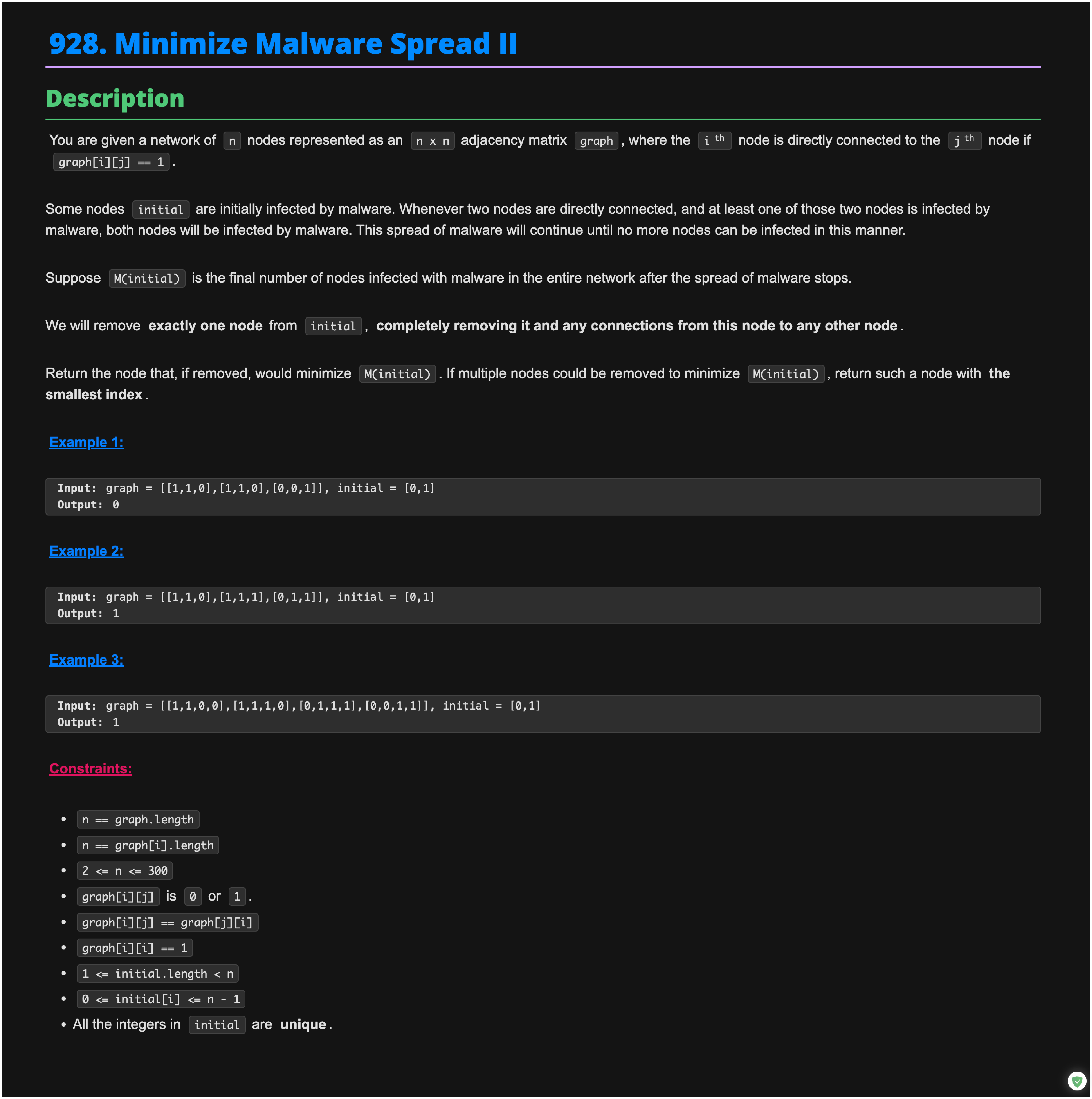

928. Minimize Malware Spread II

Description

You are given a network of `n` nodes represented as an `n x n` adjacency matrix `graph`, where the `i`th node is directly connected to the `j`th node if `graph[i][j] == 1`.

Some nodes `initial` are initially infected by malware. Whenever two nodes are directly connected, and at least one of those two nodes is infected by malware, both nodes will be infected by malware. This spread of malware will continue until no more nodes can be infected in this manner.

Suppose `M(initial)` is the final number of nodes infected with malware in the entire network after the spread of malware stops.

We will remove **exactly one node** from `initial`, **completely removing it and any connections from this node to any other node**.

Return the node that, if removed, would minimize `M(initial)`. If multiple nodes could be removed to minimize `M(initial)`, return such a node with **the smallest index**.

Example 1:

```
Input: graph = [[1,1,0],[1,1,0],[0,0,1]], initial = [0,1]
Output: 0
```

Example 2:

```
Input: graph = [[1,1,0],[1,1,1],[0,1,1]], initial = [0,1]
Output: 1
```

Example 3:

```
Input: graph = [[1,1,0,0],[1,1,1,0],[0,1,1,1],[0,0,1,1]], initial = [0,1]
Output: 1
```

Constraints:

- `n == graph.length`
- `n == graph[i].length`
- `2 <= n <= 300`
- `graph[i][j]` is `0` or `1`.
- `graph[i][j] == graph[j][i]`
- `graph[i][i] == 1`
- `1 <= initial.length < n`
- `0 <= initial[i] <= n - 1`
- All the integers in `initial` are **unique**.