Problem 3:

Understanding the problem

We're given an n*n grid with oil cells `'#'` and empty `'.'`. A scoop covers two **adjacent** (4-neighbour) oil cells and scoops can't overlap. We want the **maximum number of scoops**.

Key idea: model oil cells as vertices and "adjacent oil pairs" as edges. Picking the maximum set of disjoint pairs is **maximum bipartite matching** → solve via **max flow** (or Hopcroft–Karp). The grid is naturally bipartite by parity of (r+c)(r+c)(r+c).

Initial strategy

My first thought was: build a graph on all n2n^2n2 cells, connect adjacencies, then run max flow with capacity 1 on everything.

Two immediate issues:

1. I initially wired **every** cell to source/sink instead of **only** `'#'` **cells**, which lets matching "use" empty cells.
2. I started with an **adjacency matrix** implementation of Dinic; for n=300n=300n=300 the node count is ~90k, so an n*n matrix is infeasible (tens of GB).

Final design

- **Compress nodes**: assign ids only to oil cells `'#'`, so vertices are `0..count-1` where `count` is the number of oil cells.
- **Bipartition**: color cell (r,c)(r,c)(r,c) as **black** if (r+c)(r+c)(r+c) is odd, **white** otherwise.
- **Network**:
  - Source `S` → each **black oil** vertex (cap 1).
  - For each black oil cell, add edges to each **adjacent white oil** cell (cap 1).
  - Each **white oil** vertex → sink `T` (cap 1).

- Run **Dinic** on an **edge-list residual graph**. The max flow equals the maximum number of scoops.

What went wrong

- **Wrong vertex set**: I first connected all cells to `S/T`.

  → Fix: only map `'#'` cells to ids; skip `'.'` entirely.

- **Parity wiring bugs**: I hand-checked "if black, connect to S; else to T" with verbose conditions and made mistakes.

  → Fix: use the simple invariant `black = ((r+c)&1)`.

- **Adjacency matrix**: My matrix-based Dinic blew up for large grids.

  → Fix: switch to **edge-list residual graph** (forward+reverse edges), `O(V+E)` memory.

- **Double counting edges**: Initially I added both directions for neighbors; in bipartite flow we only add **black → white**.

  → Fix: generate neighbor edges **only from black** cells.