Word Search

Core idea:

- Do a DFS from every cell that matches the first character.
- At each step try 4 directions (up / down / left / right).
- Mark the current cell as visited so you don't reuse it in the same path.
- If you match all characters of word, return true; if all starts fail, return false.

Implementation steps:

1. Pre-check pruning (big win):
   - Count letters on the board and in word. If any word letter count exceeds the board's count → immediately return false.
   - Optional micro-heuristic: if the first char of word is more frequent than the last, reverse the word so you start from the rarer character. This reduces DFS branching.
2. DFS signature:
   - Parameters: (r, c, idx) where idx is the index in word you're matching now.
   - Base: if idx == word.size(), you matched everything → true.
   - Bounds & char check: if out of bounds or board[r][c] != word[idx] → false.
3. Visited handling:
   - In place mark the cell as visited by temporarily changing the char (e.g. to '#'), then restore it after exploring neighbors. This avoids an extra visited matrix and is cache-friendly.
   - Explore neighbors with a small direction array like dr = {-1, 1, 0, 0}, dc = {0, 0, 1, -1}.
   - Short-circuit: return true as soon as any neighbor path returns true.
4. Outer loop:
   - Scan all cells; when board[r][c] == word[0] (or word.back() if you reversed), start DFS.
   - If any start returns true → overall true; else false.

Pruning ideas for the follow-up:

- Letter feasibility check (described above) is the biggest constant-factor win.
- Word direction choice: reverse the word to start from the rarer endpoint.
- Early mismatch bailout: do the equality check before marking visited.
- Stop exploring when remaining length > cells available in a small region (often implicit due to bounds / visited).
- Short-circuit returns on first success at each level (don't explore all 4 neighbors after you found a match).

Complexity:

- Worst case is exponential, but with a 6×6 board and word length ≤ 15, plus pruning, it's fine.
- With the frequency check and word reversal heuristic, the search tree is dramatically smaller on typical cases.