Project 1 Part 3: Competitive Grid World Resource Gathering with Minimax and Alpha–Beta Pruning

Objective: Implement an computer search agent for the **two-player** grid world using **minimax** and **alpha–beta pruning**.

---

1. Problem Setup

Two players (Player A and Player B) compete in the same **5×5 grid world as in Part 2**, except:

- Both players start at **different base tiles**.
  - Player 1 starts at (0,0), Player 2 starts at (4,4).

2. Rules of Competition

- Players alternate turns (Player A moves, then Player B, then A again, etc.).
- Each move = one step (up, down, left, right) into an adjacent passable tile.
- **Conflict rule**: If both players attempt to move onto the same tile, the move resolves in turn order:
  - The player whose turn it is occupies the tile.
  - If it contained a resource, only they collect it.
- **Objective**: Deliver the most resources to your base before all resources are exhausted.
- **Game End**: When all resources have been delivered to bases.

3. Game Representation

To apply minimax and alpha–beta pruning, define the search as follows:

State (Node)

Each state will encode:

- Positions of Player A and Player B.
- Backpack contents of both players.
- Delivered resources of both players.
- Remaining resource tiles on the map.
- Whose turn it is.

Actions (Edges)

- On each turn, the active player chooses one move (up, down, left, right).

Start State

- Both players at their base positions.
- Empty backpacks.
- No resources delivered.

Terminal States (Game Over)

- All resources have been delivered to bases.

Utility Function

Zero-sum payoff:

Utility = (Player A's delivered total) – (Player B's delivered total)

- If positive → A wins.
- If negative → B wins.
- If zero → tie.

---

4. Search Algorithm Requirements

- Implement **minimax search** with a configurable search depth.
- Implement **alpha–beta pruning** to optimize minimax.
- Since the full game tree may be very large, your agent should use:
  - **Depth-limited search** (e.g., depth = 3–5).
  - A **heuristic evaluation function** for non-terminal nodes.

Heuristic Suggestions

- Remaining distance to nearest needed resource.
- Distance to base (if carrying resources).
- Opponent's potential access to resources.
- Weighted score: (your delivered + backpack) – (opponent delivered + backpack).

---

5. Deliverables

Each student must submit:

1. **Code Implementation**:
   - Minimax agent with alpha–beta pruning.
   - Ability to play against a random-move agent (for baseline).
   - Ability to play against another minimax agent.
2. **Report (2-3 pages)**:

- **Problem formulation** (state, actions, utility).
- **Heuristic design** (what features you used, why).
- **Results**:
  - Compare minimax vs. alpha–beta (nodes expanded, runtime).
  - Play games between different agents (e.g., minimax vs. random, minimax vs. minimax).
- **Discussion**:
  - Effectiveness of your heuristic.
  - Tradeoffs between depth and computation time.

---

6. Submission

Submit a .zip file containing:

- code/ directory with your implementation (include README with run instructions).
- report.pdf with your write-up.
- Any additional test cases or visualizations.