# Project 3 —Reflection Form

**One page maximum. Each partner must submit a reflection form individually on Gradescope.**

**Student Name:** Edgar Sarceno **Partner Name:** Mitchell Sampson

**1) Task Division by Question (check one per row; keep notes to 1-2 sentences)**

| Question | Lead: Me | Lead: Partner | Lead: Both | Notes |
|---|---|---|---|---|
| Q1 | [x] | [ ] | [ ] | Implemented batch value iteration and helper methods |
| Q2 | [ ] | [x] | [ ] | Partner tuned noise parameter |
| Q3 | [ ] | [ ] | [x] | Both tested different parameter combinations |
| Q4 | [x] | [ ] | [ ] | Cyclic value iteration, built on Q1 structure |
| Q5 | [ ] | [x] | [ ] | Partner implemented priority queue approach |
| Q6 | [x] | [ ] | [ ] | Q-learning update and Q-value methods |
| Q7 | [ ] | [x] | [ ] | Partner added epsilon-greedy to Q6 |
| Q8 | [ ] | [ ] | [x] | Both tried many parameter combos, ended up NOT POSSIBLE |
| Q9 | [x] | [ ] | [ ] | Ran Pacman tests and verified win rate |
| Q10 | [ ] | [x] | [ ] | Partner implemented weight updates for features |

**2) Communication & Workflow (2–4 sentences)**

• Main tools (e.g., in-person, Zoom, Slack/Discord, GitHub, shared doc):Discord for daily communication, Imessage, GitHub for code sharing, Zoom for debugging sessions.

• Frequency (e.g., daily, 2–3×/week, weekly) and typical meeting length: 3-4 times per week, mostly after class, sessions typically 1-2 hours when working on implementation, shorter check-ins otherwise

• What worked well about your coordination:
We divided the work naturally between value iteration vs Q-learning, where each could focus and become expert in our sections. Regular updates on Discord kept us aligned without the need for regular meetings.

• One thing to improve next time:
Should have started Q8 earlier because of how much trial and error went into it, much more than expected. Ultimately, too much time was consumed at the end trying several epsilon/learning rate combinations.

**3) Biggest Challenge & Resolution (2–4 sentences)**

Describe one concrete blocker (bug/design conflict) and how you resolved it.

Our biggest blocker was Q5 (prioritized sweeping). At first, we used a list for predecessors instead of a set, and this caused duplicate states in the priority queue, and made it extremely slow. It took a couple hours of debugging before my partner realized that the problem was duplicates and changed to using sets. Immediately the performance problem was resolved. We also had to remember to use negative priorities since PriorityQueue is a min-heap but we wanted max-diff states first.

**4) Contributions & Accountability (2-4 sentences)**

I developed the value iteration base (Q1), from which both subsequent variants were built; developed the core Q-learning agent with update rules; and thoroughly tested the Pacman agent. My partner handled the more complex prioritized sweeping algorithm, and the approximate Q-learning with feature weights. For Q3 and Q8 we both experimented with parameters jointly until working combinations were found. We reviewed one another's code, so we each understand all implementations in full.

I can explain every part we submitted. Initial here: E.S.