```
Best accuracy: 10/20
```

Cursor filled in most of the basic functionality of the ReAct agent, with some guidance. After a few iterations, the harness was wired well enough to start iterating on the agent.

The first set of changes were to the system prompt and adding basic tools for finding files and running tests. The agent started replacing patches with descriptions of its changes, so in the next round the prompt and tools were updated to a stricter set of tools for generating and validating patches. Asking the agent to report its reasoning as it worked on the problem- hoping this would make it easier to diagnose flaws and improve its accuracy- was not effective. Replacing this guidance with instructions for tool use improved accuracy somewhat, but some tools had bugs that caused them to fail when the agent tried to use them. Adding unit tests for the tools avoided wasting cycles on broken implementations.

Tools added: 'check_syntax' (invalid patches), 'grep' (locating symbols), 'find_files' (locate test files, related code), 'show_diff' (examine changes before generating the patch), 'analyze_test_failure' (show the output of a failed test).

Adding unit tests including variants from failed tests was also useful in making the tools more tolerant of the agent's inputs (e.g., leading/trailing whitespace, path normalization). Many of the early iterations were getting the agent to stop quitting before it had made changes or generating broken code, but these tools started to feel redundant for the smaller model, I removed most of them.

A more robust file-edit mechanism helped (error checks for line bounds, rejecting too-large patches), as did agent code forcing it to make changes before calling `finish`, and having tested those changes before making more modifications. Adding a few examples to the system prompt was also recommended for `gpt-5-mini`, but adding them didn't seem to affect accuracy at all.

Guidance for smaller models also restricts context. To this end, adding a tool that shows snippets of a file, printing line numbers with `show_file`, and truncating text/bash output to 300 lines sometimes improved results a bit.

Changing the API from the chat API to the responses API recreated the "empty path" problem; improving its tolerance for failure restored performance without repeating the system prompt in every step (used the `previous_response_id` parameter to chain requests). The motivation was to see the effect on temperature, but `gpt-5-mini` doesn't support a temperature other than 1.

At this point, most runs solved 7 or 8 of the instances. One outlier run solved 10. Then I switched coding assistants first to Codex (no change) and then to Claude Code. Claude explored guidance to use simpler tools and shorter prompts given the smaller model, but after 5 iterations the accuracy was unchanged. Writing a script that summarized each instance's trajectory using a more sophisticated model, then using *that* report to plan modifications to the agent improved accuracy to 50% (see `ITERATION_RESULTS.txt`). It reproduced on a subsequent iteration.