(a) Initial Plan (Extracted)

```python
def solution(agent, start_from=1):
```
General plan:
```python
# General plan: I need to get a list of receptacles to find the
# lettuce, clean it, and put it in a diningtable.
```

Sub-plan:
```python
# [Step 1] Get a list of receptacles where the lettuce is likely
# to appear.
    if start_from <= 1:
```

In-plan refine:
```python
        recep_to_go= literal_eval(ask_LLM(f'Sort
{agent.receptacles} in descending order based on the
likelihood of finding a lettuce.'))
        assert recep_to_go, f'Error in [Step 1]: recep_to_go
should not be empty. {agent.report()}'

# [Step 2] Go to each receptacle in the list until seeing a
# lettuce
    if start_from <= 2:
        for receptacle in recep_to_go:
            observation = agent.goto(receptacle)
            if 'lettuce' in observation:
                break
```

Predict via assert:
```python
        assert 'lettuce' in observation, f'Error in [Step 2]: There
is no lettuce in/on {recep_to_go}. {agent.report()}'

# [Step 3] Identify the lettuce I just found and clean it
    if start_from <= 3:
```

In-plan refine:
```python
        found_lettuce = ' lettuce' + ask_LLM(f'From the
observation, get the identifier of the lettuce: {observation}.')
        observation = agent.take(found_lettuce, receptacle)
        observation = agent.clean(found_lettuce, 'sinkbasin 1')
        assert 'clean' in observation, f'Error in [Step 3]: I cannot
clean {found_lettuce} using the sinkbasin 1. {agent.report()}'

# [Step 4] Go to a diningtable and put the lettuce on it.
    ...
```

Environment

Execution → Halt at [Step 3]

AdaPlanner

Out-of-plan refine

(b) Out-of-plan Feedback (Extracted)

Error in [Step 3]: I cannot clean lettuce 2 using the sinkbasin 1. I am at toilet 1 and holding None. The last three interactions before the error were:
Act: go to sinkbasin 2
Obs: On the sinkbasin 2, you see nothing.
...

(c) Refined Plan (Extracted, revisions highlighted)

```python
def solution(agent, start_from=3):
# General plan: I need to get a list of receptacles to find the lettuce,
# take the lettuce to the sinkbasin, clean it, and put it in a diningtable.

# [Step 1] Get a list of receptacles where the lettuce is likely to appear.
    ...
# [Step 2] Go to each receptacle in the list until seeing a lettuce
    ...
# [Step 3] Identify the lettuce I just found and take it
    if start_from <= 3:
        found_lettuce = 'lettuce' + ask_LLM(f'From the observation, get
the identifier of the lettuce: {observation}. ')
        observation = agent.take(found_lettuce, receptacle)
        assert agent.holding == found_lettuce, f'Error in [Step 3]: I cannot
take {found_lettuce} from the {receptacle}. {agent.report()}'

# [Step 4] Go to a sinkbasin to clean the lettuce.
    if start_from <= 4:
        observation = agent.goto('sinkbasin 1')
        observation = agent.clean(found_lettuce, 'sinkbasin 1')
        assert 'clean' in observation, f'Error in [Step 4]: I cannot clean
{found_lettuce} using the sinkbasin 1. {agent.report()}'

# [Step 5] Go to a diningtable and put the lettuce on it.
    ...
```