1[20%].
This problem is similar to question 2.10 in the text book, except the environment configuration is slightly different.

Consider the following vacuum-cleaning world which has 3 squares:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
</table>

We assume the following:

- The vacuum-cleaning agent knows the configuration of the 3 squares, but does not know which square it is in initially. The initial dirt distribution is not known either.
- A clean square remains clean and a dirty square remains dirty unless the agent cleans it up.
- The agent has two sensors: \textit{Loc}(x) and \textit{Status}(x). \textit{Loc}(x) returns the identity ("A" or "B" or "C") of the square \( x \) that the agent is located, and \textit{Status}(x) returns either "Dirty" or "Clean" for the square \( x \). Assume that the sensors are 100% reliable.
- The actions available for the agent are: \textit{Left}, \textit{Right}, \textit{Suck}, \textit{NoOp}. Each action takes place in one time "step".
- The actions are perfectly reliable - perform "Suck" in dirty square will change it to become clean, and moving left when in square C will get the agent to square B.
- The agent will be penalized by \(-1\) point for each movement \textit{Left}, \textit{Right}, \textit{Suck}. The agent gets +10 points reward for each dirty square cleaned.
- For a time horizon \( T \) (say \( T = 100 \)) steps, the agent is penalized by \(-2\) points per dirty square per time step. (If a square is dirty at the "start" of time step \( t \) and the agent performs "Suck" at this time step, the square becomes "clean" at the "end" of step \( t \). Thus for time step \( t \) for this particular square, no penalty should be applied to the agent).
- The agent’s performance is measured by the total number of points (positive or negative) over \( T = 100 \) time steps.

(a) Design a simple reflex agent (without states) for this problem - provide pseudo-code for the agent and define the "condition-action" rules. Can such an agent be perfectly rational (i.e., maximizing the expected performance measure) for any initial agent location and dirt distribution? Explain your answer.

(b) What about a reflex agent with states (i.e., an agent which has an internal state representation of the world environment)? Can such an agent be perfectly rational under the current assumptions? Design such an agent (pseudo-code, rules, or look-up table, etc.).

(c) Now assume that the agent’s dirt sensor is more powerful which provides complete information about dirty/clean status of every square in the environment. Can a simple reflex agent be perfectly rational now?
2[50%].

(a) Questions 3.6(a) and 3.6(d) in the text book.
(b) Question 3.9(a) in the text book.
(c) Questions 3.14(a), 3.14(c) and 3.14(e) in the text book.
(d) Question 3.21 in the text book.
(e) Give an example heuristic function $h$ (on an example state-space graph) such that $h$ is admissible but not consistent. Show how this heuristic function could lead to a sub-optimal solution on your example graph with $A^*$ graph search.

3[30%].
Consider again the vacuum-cleaning problem defined in Question 1 of this homework. We make the following modifications to the assumptions in Question 1:

- We assume a 3 by 3 grid world known to the agent. The environment is fully observable: the percepts give complete information about the dirty/clean status of every square and the agent’s location.
- The agent’s actions now include "Up", "Down" in addition to the ones assumed in Question 1. Each of these new actions will also incur a penalty of $-1$ point.
- The agent will NOT get a reward of 10 points for each dirty square cleaned - so that the agent will only receive negative points for dirty squares and for its actions. (However because the agent will get a penalty of -2 points per dirty square per time step, cleaning up a dirty square still benefits the agent’s performance).
- The agent’s performance is measured by the sum of negative points received on reaching a state with all squares in the environment being "clean".

(a) Define the vacuum-cleaning problem as a state-space search problem. Specify the states, actions, the transition model, and the goal state(s). The initial state could be any of the states in the state space.

You should also specify a cost function $C(n, a, n') > 0$, which defines the cost of one step transition from node $n$ to node $n'$ by applying action $a$, such that the cost of a path in the search tree should be exactly the absolute value of the sum of penalty points due to dirty square(s) and moves.

(b) Which of the algorithms defined in Chapter 3 would be appropriate for this problem? Should the algorithm use tree search or graph search?

(c) Write a program to implement the $A^*$ algorithm for the search problem and define a suitable admissible heuristic function $h$. Run the program for the 3 by 3 grid world with top 3 squares dirty and agent in the center square as the initial state. Print out the sequence of actions and the associated sequence of states in one optimal path returned by the program.

4[Bonus:20%]

Question 3.32 in the text book.