PRINCIPLES OF ARTIFICIAL INTELLIGENCE LABORATORY PROGRAMS

Ao* SEARCH ALGORITHM PYTHON PROGRAM

SOURCE CODE:

```
import heapq

class Node:
    def __init__(self, state, g_value, h_value, parent=None):
        self.state = state
        self.g_value = g_value
        self.h_value = h_value
        self.parent = parent

    def f_value(self):
        return self.g_value + self.h_value

def ao_star_search(initial_state, is_goal, successors, heuristic):
    open_list = [Node(initial_state, 0, heuristic(initial_state), None)]
    closed_set = set()

    while open_list:
        open_list.sort(key=lambda node: node.f_value())
        current_node = open_list.pop(0)

        if is_goal(current_node.state):
            path = []
            while current_node:
                path.append(current_node.state)
                current_node = current_node.parent
            return list(reversed(path))

        closed_set.add(current_node.state)

        for child_state in successors(current_node.state):
            if child_state in closed_set:
```

```
            continue
        g_value = current_node.g_value + 1
        h_value = heuristic(child_state)
        child_node = Node(child_state, g_value, h_value, current_node)
        in_open_list = any(node.state == child_state for node in open_list)
        if not in_open_list:
            open_list.append(child_node)
        else:
            existing_node = next(node for node in open_list if node.state == child_state)
            if existing_node.g_value > g_value:
                existing_node.g_value = g_value
                existing_node.parent = current_node

    return None

def is_goal(state):
    return state == (4, 4)

def successors(state):
    x, y = state
    return [(x + 1, y), (x, y + 1)]

def heuristic(state):
    x, y = state
    return abs(4 - x) + abs(4 - y)

initial_state = (0, 0)
path = ao_star_search(initial_state, is_goal, successors, heuristic)
if path:
    print("Path found:", path)
else:
    print("No path found")
```

OUTPUT:

03-04-2024 Ao Algorithm.py - C:\Users\ur mom\Documents\PRINCIPLES OF AI\SANTHOSHKUMAR S 231501147\03-04-2024 Ao Algorithm.py (3.9.10)

File Edit Format Run Options Window Help

```python
import heapq

class Node:
    def __init__(self, state, g_value, h_value, parent=None):
        self.state = state
        self.g_value = g_value
        self.h_value = h_value
        self.parent = parent

    def f_value(self):
        return self.g_value + self.h_value

def ao_star_search(initial_state, is_goal, successors, heuristic):
    open_list = [Node(initial_state, 0, heuristic(initial_state), None)]
    closed_set = set()

    while open_list:
        open_list.sort(key=lambda node: node.f_value())
        current_node = open_list.pop(0)

        if is_goal(current_node.state):
            path = []
            while current_node:
                path.append(current_node.state)
                current_node = current_node.parent
            return list(reversed(path))

        closed_set.add(current_node.state)
```

IDLE Shell 3.9.10

File Edit Shell Debug Options Window Help

```
Python 3.9.10 (tags/v3.9.10:f2f3f53, Jan 17 2022, 15:14:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ur mom\Documents\PRINCIPLES OF AI\SANTHOSHKUMAR S 231501147\03-04-2
024 Ao Algorithm.py
Path found: [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (4, 1), (4, 2), (4, 3), (4, 4)]
>>>
```