suma1.py - C:/Users/surya/suma1.py (3.12.2)

File Edit Format Run Options Window Help

```python
class Node:
    def __init__(self, state, parent=None, cost=0, heuristic=0):
        self.state = state
        self.parent = parent
        self.cost = cost
        self.heuristic = heuristic
        self.f = cost + heuristic
def is_goal(state, goal):
    return state == goal
def generate_successors(node, goal):
    successors = []
    if node.state < goal:
        successors.append(Node(node.state + 1, node, node.cost + 1, heuristic(node.state + 1, goal)))
    return successors
def heuristic(state, goal):
    return abs(goal - state)
def rbfs(node, f_limit, goal):
    if is_goal(node.state, goal):
        return node
    successors = generate_successors(node, goal)
    if not successors:
        return None
    while True:
        successors.sort(key=lambda x: x.f)
        best = successors[0]
        if best.f > f_limit:
            return None
        if len(successors) > 1:
            alternative = successors[1].f
        else:
            alternative = float('inf')
        result = rbfs(best, min(f_limit, alternative), goal)
        if result is not None:
            return result
initial_state = 0
goal_state = 5
initial_node = Node(initial_state, None, 0, heuristic(initial_state, goal_state))
```

suma1.py - C:/Users/surya/suma1.py (3.12.2)

File Edit Format Run Options Window Help

```
def generate_successors(node, goal):
    successors = []
    if node.state < goal:
        successors.append(Node(node.state + 1, node, node.cost + 1, heuristic(node.state + 1, goal)))
    return successors
def heuristic(state, goal):
    return abs(goal - state)
def rbfs(node, f_limit, goal):
    if is_goal(node.state, goal):
        return node
    successors = generate_successors(node, goal)
    if not successors:
        return None
    while True:
        successors.sort(key=lambda x: x.f)
        best = successors[0]
        if best.f > f_limit:
            return None
        if len(successors) > 1:
            alternative = successors[1].f
        else:
            alternative = float('inf')
        result = rbfs(best, min(f_limit, alternative), goal)
        if result is not None:
            return result
initial_state = 0
goal_state = 5
initial_node = Node(initial_state, None, 0, heuristic(initial_state, goal_state))
solution = rbfs(initial_node, float('inf'), goal_state)

if solution is not None:
    path = []
    while solution is not None:
        path.append(solution.state)
        solution = solution.parent
    path.reverse()
    print("RBFS Path:", path)
else:
```

IDLE Shell 3.12.2

File Edit Shell Debug Options Window Help

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win
32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/surya/suma1.py
RBFS Path: [0, 1, 2, 3, 4, 5]
>>>
```