PROGRAM-13

HILL CLIMBING PROBLEM

AIM:-

To write and execute the python program for the Hill climbing program.

PROCEDURE:-

- **Input**:
  - initial_state: The starting state of the search.
  - goal_state: The target state to achieve.
  - evaluate: A function that evaluates the "goodness" of a state. .
- **Algorithm**:
  - Start with the current_state as the initial_state.
  - While current_state is not equal to the goal_state:
    - Generate neighboring states by randomly changing either the x-coordinate or y-coordinate of the current state.
    - Choose the neighbor with the highest evaluation score.

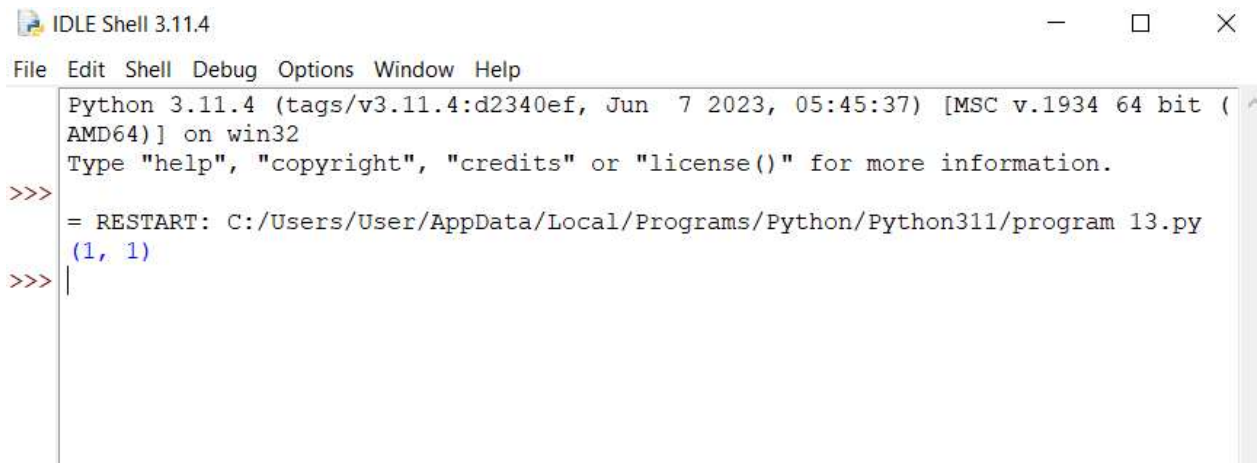
- **Example evaluation function**:
  - It calculates the negative Manhattan distance from a given state to the goal state. Negative distance is used because the hill climbing algorithm seeks to maximize this value.
- **Initialization**:
  - Define initial_state and goal_state.
  - Define an evaluation function using a lambda expression.
- **Print Result**:
  - Print the result of running hill_climbing with the provided parameters.

CODING:-

```
import random

def hill_climbing(initial_state, goal_state, evaluate):

    current_state = initial_state
```

```
    while current_state != goal_state:

        neighbors = [(current_state[0] + random.choice([-1, 1]), current_state[1]),
(current_state[0], current_state[1] + random.choice([-1, 1]))]

        next_state = max(neighbors, key=evaluate)

        if evaluate(next_state) <= evaluate(current_state):

            break

        current_state = next_state

    return current_state

initial_state = (0, 0)

goal_state = (5, 5)

evaluate = lambda state: -(abs(state[0] - goal_state[0]) + abs(state[1] - goal_state[1]))

print(hill_climbing(initial_state, goal_state, evaluate))
```

OUTPUT:-



RESULT:-

Hence the program has been successfully executed and verified.