Faculty of Technology and Engineering

Chandubhai S. Patel Institute of Technology

Date: / /

Practical Performa

Academic Year	:	2025-26	Semester	:	7 th
Course code	:	OCCSE4001	Course name	:	Reinforcement Learning

Practical- No. 1

Aim: Introduction to RL with Grid World Code:

```
import numpy as np
import time
from IPython.display import clear_output

# -- Environment Setup --
class GridWorld:
    def __init__(self, size=5):
        self.size = size
        self.start_pos = (0, 0)
        self.goal_pos = (size - 1, size - 1)
        self.agent_pos = self.start_pos

        # 0: up, 1: down, 2: left, 3: right
        self.action_space = [0, 1, 2, 3]

    def reset(self):
        """Resets the agent to the starting position."""
```

```
        self.agent_pos = self.start_pos
        return self.agent_pos

    def step(self, action):
        """Moves the agent and returns the new state,
reward, and done status."""
        x, y = self.agent_pos
        if action == 0:  # Up
            x = max(0, x - 1)
        elif action == 1:  # Down
            x = min(self.size - 1, x + 1)
        elif action == 2:  # Left
            y = max(0, y - 1)
        elif action == 3:  # Right
            y = min(self.size - 1, y + 1)
        self.agent_pos = (x, y)
        done = self.agent_pos == self.goal_pos
        reward = 1 if done else -0.1
        return self.agent_pos, reward, done

    def render(self):
        """Prints the grid to the console."""
        grid = np.full((self.size, self.size), '_')
        grid[self.start_pos] = 'S'
        grid[self.goal_pos] = 'G'
        grid[self.agent_pos] = 'A'
        print(grid)
        print("-" * 10)

# -- Agent and Policies --
def random_policy(env):
    """A policy that chooses actions randomly."""
    return np.random.choice(env.action_space)

def deterministic_policy(state, env):
```

```
    """A simple policy: move right, then down."""
    x, y = state
    if y < env.size - 1:
        return 3  # Move Right
    else:
        return 1  # Move Down

# -- Simulation --
env = GridWorld(size=4)
print("--- Running with Random Policy ---")
state = env.reset()
done = False
while not done:
    env.render()
    action = random_policy(env)
    state, reward, done = env.step(action)
    time.sleep(0.5)
    clear_output(wait=True)
env.render()
print("Goal Reached!")

print("\n--- Running with Deterministic Policy ---")
state = env.reset()
done = False
while not done:
    env.render()
    action = deterministic_policy(state, env)
    state, reward, done = env.step(action)
    time.sleep(0.5)
    clear_output(wait=True)
env.render()
print("Goal Reached!")
```

Output:

```
[['S' '_' '_' '_']
 ['_' '_' '_' '_']
 ['_' '_' '_' '_']
 ['_' '_' '_' 'A']]
----------
Goal Reached!
```

Grade/Marks

(_____ / 10)

Sign of Lab Teacher with Date