Planning Program Block

class BlockWorld:

def \_\_init\_\_(self, blocks):

self.blocks = blocks # Store block positions

self.actions = [] # List of actions taken

def move(self, block, from\_stack, to\_stack):

"""Move block from one stack to another."""

if from\_stack == 'table':

self.blocks[block] = to\_stack

self.actions.append(f"Move {block} from table to {to\_stack}")

else:

self.blocks[block] = to\_stack

self.actions.append(f"Move {block} from {from\_stack} to {to\_stack}")

def stack(self, block1, block2):

"""Stack block1 on top of block2."""

self.blocks[block1] = block2

self.actions.append(f"Stack {block1} on {block2}")

def unstack(self, block):

"""Unstack block from the stack it's on."""

if self.blocks[block] != 'table':

self.blocks[block] = 'table'

self.actions.append(f"Unstack {block} from {self.blocks[block]}")

def print\_state(self):

"""Print the current state of the blocks."""

for block, position in self.blocks.items():

print(f"{block} is on {position}")

def goal\_reached(self, goal\_state):

"""Check if the current state matches the goal state."""

return self.blocks == goal\_state

def get\_input():

"""Interactive input for block world."""

# Get the initial configuration of blocks from the user

blocks = {}

n = int(input("Enter number of blocks: "))

for \_ in range(n):

block = input("Enter block name: ")

position = input(f"Where is {block} (on table or on another block): ")

blocks[block] = position

# Get the goal configuration from the user

goal\_state = {}

print("\nEnter goal state:")

for block in blocks:

goal\_position = input(f"Where do you want {block} (on table or on another block): ")

goal\_state[block] = goal\_position

# Create BlockWorld object

block\_world = BlockWorld(blocks)

# Get the sequence of actions to reach the goal state

while True:

print("\nCurrent State:")

block\_world.print\_state()

action = input("\nEnter action (move, stack, unstack, done): ").lower()

if action == "move":

block = input("Enter block to move: ")

from\_stack = input(f"Where is {block} currently (table or block it is on): ")

to\_stack = input(f"Where do you want to move {block} (table or on top of which block): ")

block\_world.move(block, from\_stack, to\_stack)

elif action == "stack":

block1 = input("Enter block to stack: ")

block2 = input(f"Enter block to stack {block1} on: ")

block\_world.stack(block1, block2)

elif action == "unstack":

block = input("Enter block to unstack: ")

block\_world.unstack(block)

elif action == "done":

if block\_world.goal\_reached(goal\_state):

print("\nGoal reached!")

else:

print("\nGoal not reached.")

break

else:

print("Invalid action. Please enter 'move', 'stack', 'unstack', or 'done'.")

# Output the actions and the final state

print("\nActions taken:")

for action in block\_world.actions:

print(action)

print("\nFinal State:")

block\_world.print\_state()

# Main driver

get\_input()

OUTPUT:

Enter number of blocks: 3

Enter block name: A

Where is A (on table or on another block): table

Enter block name: B

Where is B (on table or on another block): table

Enter block name: C

Where is C (on table or on another block): table

Enter goal state:

Where do you want A (on table or on another block): B

Where do you want B (on table or on another block): C

Where do you want C (on table or on another block): table

Current State:

A is on table

B is on table

C is on table

Enter action (move, stack, unstack, done): move

Enter block to move: A

Where is A currently (table or block it is on): table

Where do you want to move A (table or on top of which block): B

Current State:

A is on B

B is on table

C is on table

Enter action (move, stack, unstack, done): stack

Enter block to stack: B

Enter block to stack B on: C

Current State:

A is on B

B is on C

C is on table

Enter action (move, stack, unstack, done): done

Goal reached!

Actions taken:

Move A from table to B

Stack B on C

Final State:

A is on B

B is on C

C is on table