Exercise 10: Implementation of Block World Problem

Block World Problem:

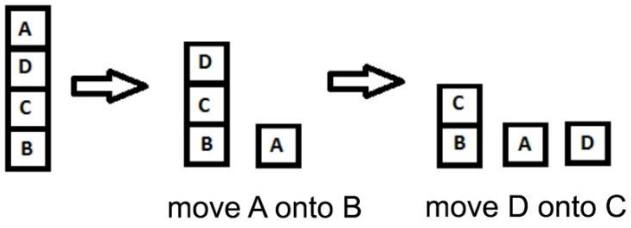
Aim:

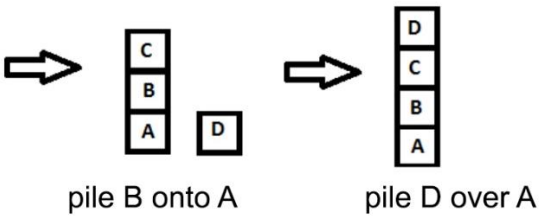
To Solve Block World problem using Goal Stack Algorithm.

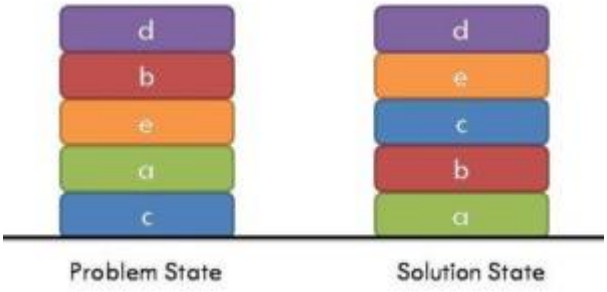
Algorithm:

In **Block World** problem we will model a simple block world under certain rules and constraints. That is, we will ``program'' a robotic arm to respond to a limited set of commands.

- Push the Goal state in to the Stack
- Push the individual Predicates of the Goal State into the Stack
- Loop till the Stack is empty
    - Pop an element E from the stack
        - IF E is a Predicate
            - IF E is True then
                - Do Nothing
- ELSE
    - Push the relevant action into the Stack
    - Push the individual predicates of the Precondition of the action into the Stack
- Else IF E is an Action
    - Apply the action to the current State.
    - Add the action 'a' to the plan
    -







Program:

```
tab = []
result = []
problem = ["c", "a", "e", "d", "b"]
goalList = ["a", "b", "c", "d", "e"]

def parSolution(N):
    for i in range(N):
        if goalList[i] != result[i]:
            return False
    return True


def Onblock(index, count):

    if count == len(goalList)+1:
        return True

    block = tab[index]

    result.append(block)
    print(result)
    if parSolution(count):
        print("Valid Step - Pushing Result Soln ")
        tab.remove(block)
```

```
        Onblock(0, count + 1)
    else:
        print("Invalid Step - Back To Tab")
        result.pop()
        Onblock(index+1, count)


def Ontab(problem):

    if len(problem) != 0:
        tab.append(problem.pop())
        Ontab(problem)
    else:
        return True


def goal_stack_planing(problem):
    Ontab(problem)
    if Onblock(0, 1):
        print(result)

if __name__ == "__main__":

    print("Initial -> Goal")
    for k, j in zip(goalList, problem):
        print(j+"         "+k)
    goal_stack_planing(problem)
    print("\nFinal Result Solution:")
    print(result)
```

OUTPUT:

```
Initial -> Goal
c          a
a          b
e          c
d          d
b          e
['b']
Invalid Step - Back To Tab
['d']
Invalid Step - Back To Tab
['e']
Invalid Step - Back To Tab
['a']
Valid Step - Pushing Result Soln
['a', 'b']
Valid Step - Pushing Result Soln
['a', 'b', 'd']
Invalid Step - Back To Tab
['a', 'b', 'e']
Invalid Step - Back To Tab
['a', 'b', 'c']
Valid Step - Pushing Result Soln
['a', 'b', 'c', 'd']
Valid Step - Pushing Result Soln
['a', 'b', 'c', 'd', 'e']
Valid Step - Pushing Result Soln

Final Result Solution:
['a', 'b', 'c', 'd', 'e']


...Program finished with exit code 0
Press ENTER to exit console.
```

Result:

The given program was successfully created and was successfully executed in an Online C++ compiler.