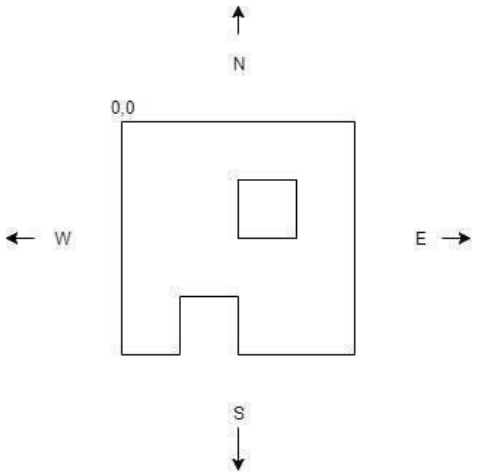
Description

Company has decided to launch a new automated cleaning robot to the market. The robot shall be able to clean all surfaces in a room, automatically, without manual intervention.

To map the operating space, the robot will receive information about the room as a set of cells. Each cell represents:

- A cleanable space of 1 by 1 that can be occupied and cleaned (S).
- A column of 1 by 1 which can't be occupied or cleaned (C)
- A wall represented by an empty cell (null) or by being outside the matrix

The map is provided as a matrix of m by n in which each element of the matrix is one of those items or else it is empty (null). For example, the 4x4 map (top left is 0,0):



[

["S", "S", "S", "S"]

["S", "S", "C", "S"]

["S", "S", "S", "S"]

["S", *null*, "S", "S"]

]

The robot also recognizes a set of basic commands. Each command drains the battery of the robot by a certain amount.

- **Turn Left** (TL). Instructs the robot to turn 90 degrees to the left. Consumes 1 unit of battery.
- **Turn Right** (TR). Instructs the robot to turn 90 degrees to the right. Consumes 1 unit of battery.
- **Advance** (A). Instructs the robot to advance one cell forward into the next cell. Consumes 2 unit of battery.
- **Back** (B). Instructs the robot to move back one cell without changing direction. Consumes 3 units of battery.
- **Clean** (C). Instructs the robot to clean the current cell. Consumes 5 units of battery.

A sequence of valid commands may look like:
[C, TR, A, C, A, C, A, C, TL, B, C, A, A, C, A, A, C, TL, C].

The robot will carry out the commands in the command set unless it hits an obstacle (a column or a wall) or runs out of battery. If a command results in hitting an obstacle, it will consume the battery, but the robot will not move. Instead, it will initiate a back off strategy:

- Perform [TR, A, TL]. If an obstacle is hit, drop the rest of the sequence and
- perform [TR, A, TR]. If an obstacle is hit, drop the rest of the sequence and
- perform [TR, A, TR]. If an obstacle is hit, drop the rest of the sequence and
- perform [TR, B, TR, A]. If an obstacle is hit, drop the rest of the sequence and
- perform [TL, TL, A]. If an obstacle is hit,
- the robot is considered stuck. Skip all the remaining commands and finish the program.

The robot will execute each command in order until no more commands are left, the battery is spent, or all the back off sequences hit an obstacle. For example, if the robot has 4 units of battery and a “clean” command (requiring 5 units of battery) is received, the robot will stop and finish the program with 4 units of battery left. If the battery low condition is hit during a back off sequence, the robot does not continue with the next back off strategy, but also finishes the program immediately.

To provide the information to the robot, a JSON file is provided with the following format:

```
{
  "map": [
    ["S", "S", "S", "S"],
    ["S", "S", "C", "S"],
    ["S", "S", "S", "S"],
    ["S", null, "S", "S"]
  ],
  "start": {"X": 3, "Y": 0, "facing": "N"},
  "commands": [ "TL","A","C","A","C","TR","A","C"],
  "battery": 80
}
```

In which “map” contains the map; “start” contains the starting point of the robot as X, Y with X being the column and Y being the row, for example, the column obstacle in the example is on position (2,1) and the direction it is facing which can be North (N), East (E), South (S) or West (W); “commands” contains the ordered list of commands to execute,
“battery” the initial battery level.

Upon execution, the robot must produce a result json which describes the results of the cleaning containing:

- All cells visited.
- All cells cleaned.
- Final position of the robot.
- Final battery left.

```
{
 “visited” : [ {“X”: 1, “Y”:0}, {“X”: 2, “Y” : 0}, {“X”:3, “Y”: 0}],
 “cleaned”: [{“X”: 1, “Y”:0}, {“X”: 2, “Y” : 1}],
 “final”: { “X” : 2, “Y” : 0, “facing”: “E”},
 “battery” : 54
}
```

Task

Create a .NET Core Microservice that will receive a json file as an input with the parameter specified and will run the robot simulation and produce the output as specified:

- Visited will contain all the cells visited by the robot.
- Cleaned will contain all the cells cleaned by the robot.
- Final will contain the final position and facing direction of the robot.
- Battery will contain the final battery left on the robot.
- The program shall be invoked by REST API POST method call

Guidance.

- The code and the whole solution should be production quality. Follow good design patterns, keep things simple and elegant.
- In the provided ZIP file, together with the description of this problem, there are two JSON examples for both a sample input and expected output.