用栈操作构建数组

给你一个目标数组 `target` 和一个整数 `n`。每次迭代，需要从 `list = {1,2,3..., n}` 中依序读取一个数字。

请使用下述操作来构建目标数组 `target` ：

- **Push**：从 `list` 中读取一个新元素， 并将其推入数组中。
- **Pop**：删除数组中的最后一个元素。
- 如果目标数组构建完成，就停止读取更多元素。

题目数据保证目标数组严格递增，并且只包含 `1` 到 `n` 之间的数字。

请返回构建目标数组所用的操作序列。

题目数据保证答案是唯一的。

示例 1：

输入：`target = [1,3], n = 3`

输出：`["Push","Push","Pop","Push"]`

解释：

读取 `1` 并自动推入数组 `-> [1]`

读取 `2` 并自动推入数组，然后删除它 `-> [1]`

读取 `3` 并自动推入数组 `-> [1,3]`

示例 2：

输入：`target = [1,2,3], n = 3`

输出：`["Push","Push","Push"]`

示例 3：

输入：`target = [1,2], n = 4`

输出：`["Push","Push"]`

解释：只需要读取前 `2` 个数字就可以停止。

示例 4：

输入：`target = [2,3,4], n = 4`

输出：["Push","Pop","Push","Push","Push"]

```
class Solution {
public:
    vector<string> buildArray(vector<int>& target, int n) {
        vector<string> res;
        unordered_map<int,int> tmp;
        for(int i=0;i<target.size();i++)
        {
            tmp[target[i]]++;
        }
        for(int i=1;i<=target[target.size()-1];i++)
        {
            if(tmp[i])
            {
                res.push_back("Push");
            }
            else
            {
                res.push_back("Push");
                res.push_back("Pop");
            }
        }
        return res;
    }
};
```