1. You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have security systems connected and it will automatically contact the police if two adjacent houses were broken into on the same night. Given an integer array nums representing the amount of money of each house, return the maximum amount of money you can rob tonight without alerting the police
Input: nums = [1,2,3,1]
Output: 4
Explanation: Rob house 1 (money = 1) and then rob house 3 (money = 3).
Total amount you can rob = 1 + 3 = 4.

Input: nums = [2,7,9,3,1]
Output: 12
Explanation: Rob house 1 (money = 2), rob house 3 (money = 9) and rob house 5 (money = 1).
Total amount you can rob = 2 + 9 + 1 = 12.

2. You are given an integer array prices where prices[i] is the price of a given stock on the ith day, and an integer k.
Find the maximum profit you can achieve. You may complete at most k transactions.
Note: You may not engage in multiple transactions simultaneously (i.e., you must sell the stock before you buy again).
Input: k = 2, prices = [2,4,1]
Output: 2
Explanation: Buy on day 1 (price = 2) and sell on day 2 (price = 4), profit = 4-2 = 2.

Input: k = 2, prices = [3,2,6,5,0,3]
Output: 7
Explanation: Buy on day 2 (price = 2) and sell on day 3 (price = 6), profit = 6-2 = 4. Then buy on day 5 (price = 0) and sell on day 6 (price = 3), profit = 3-0 = 3.

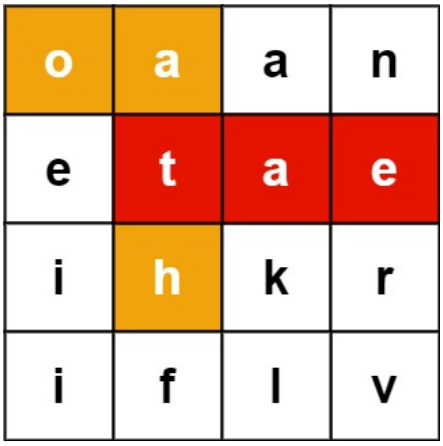
3. You are given a string `s`. You can convert `s` to a palindrome by adding characters in front of it. Return *the shortest palindrome you can find by performing this transformation*.

Input: `s = "aacecaaa"`
Output: `"aaacecaaa"`

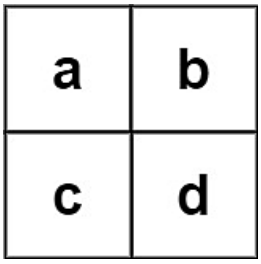
Input: `s = "abcd"`
Output: `"dcbabcd"`

4. Given an m x nboard of characters and a list of strings words, return all words on the board. Each word must be constructed from letters of sequentially adjacent cells, where adjacent cells

are horizontally or vertically neighboring. The same letter cell may not be used more than once in a word.



Input: board = [["o","a","a","n"],["e","t","a","e"],["i","h","k","r"],["i","f","l","v"]], words = ["oath","pea","eat","rain"]
Output: ["eat","oath"]



Input: board = [["a","b"],["c","d"]], words = ["abcb"]
Output: []

5. Convert a non-negative integer `num` to its English words representation.

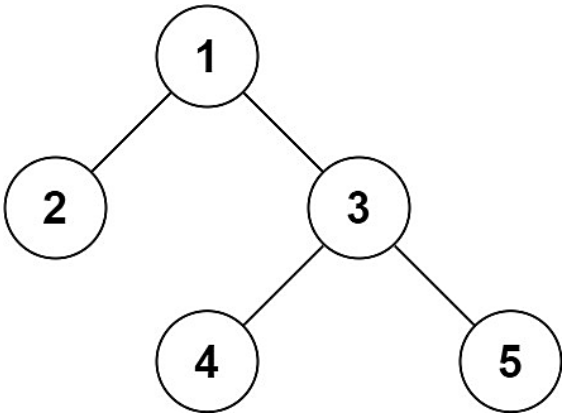
```
Input: num = 123
Output: "One Hundred Twenty Three"

Input: num = 12345
Output: "Twelve Thousand Three Hundred Forty Five"

Input: num = 1234567
Output: "One Million Two Hundred Thirty Four Thousand Five Hundred Sixty Seven"
```

6. Serialization is the process of converting a data structure or object into a sequence of bits so that it can be stored in a file or memory buffer, or transmitted across a network connection link to be reconstructed later in the same or another computer environment.
Design an algorithm to serialize and deserialize a binary tree. There is no restriction on how your serialization/deserialization algorithm should work. You just need to ensure that a binary tree can

be serialized to a string and this string can be deserialized to the original tree structure.
Clarification: The input/output format is the same as how serializes a binary tree. You do not necessarily need to follow this format, so please be creative and come up with different approaches yourself.
Example 1:



Input: root = [1,2,3,null,null,4,5]
Output: [1,2,3,null,null,4,5]

Input: root = []
Output: []

7. You are given `n` balloons, indexed from `0` to `n - 1`. Each balloon is painted with a number on it represented by an array `nums`. You are asked to burst all the balloons. If you burst the `i`th balloon, you will get `nums[i - 1] * nums[i] * nums[i + 1]` coins. If `i - 1` or `i + 1` goes out of bounds of the array, then treat it as if there is a balloon with a `1` painted on it.

Return *the maximum coins you can collect by bursting the balloons wisely*.

Input: `nums = [3,1,5,8]`
Output: `167`
Explanation:
```
nums = [3,1,5,8] --> [3,5,8] --> [3,8] --> [8] --> []
coins =  3*1*5    +   3*5*8   +  1*3*8  + 1*8*1 = 167
```

Input: `nums = [1,5]`
Output: `10`