Problem 1238: Circular Permutation in Binary Representation

Problem Information

Difficulty: Medium
Acceptance Rate: 72.35%
Paid Only: No
Tags: Math, Backtracking, Bit Manipulation

Problem Description

Given 2 integers `n` and `start`. Your task is return **any** permutation `p` of `(0,1,2.....,2^n -1) `such that :

* `p[0] = start` * `p[i]` and `p[i+1]` differ by only one bit in their binary representation. * `p[0]` and `p[2^n -1]` must also differ by only one bit in their binary representation.

Example 1:

Input: n = 2, start = 3 **Output:** [3,2,0,1] **Explanation:** The binary representation of the permutation is (11,10,00,01). All the adjacent element differ by one bit. Another valid permutation is [3,1,0,2]

Example 2:

Input: n = 3, start = 2 **Output:** [2,6,7,5,4,0,1,3] **Explanation:** The binary representation of the permutation is (010,110,111,101,100,000,001,011).

Constraints:

* `1 <= n <= 16` * `0 <= start < 2 ^ n`

Code Snippets

C++:

```
class Solution {
public:
vector<int> circularPermutation(int n, int start) {

}
};
```

Java:

```
class Solution {
public List<Integer> circularPermutation(int n, int start) {

}
}
```

Python3:

```
class Solution:
def circularPermutation(self, n: int, start: int) -> List[int]:
```