Class Notes 9

▶ Python

▶ JavaScript

▼ JAVA

Question 1

The **Fibonacci numbers**, commonly denoted `F(n)` form a sequence, called the **Fibonacci sequence**, such that each number is the sum of the two preceding ones, starting from `0` and `1`. That is,

```
F(0) = 0, F(1) = 1
F(n) = F(n - 1) + F(n - 2), for n > 1.
```

Given `n`, calculate `F(n)`.

Explanation :

The Fibonacci numbers are the numbers in the following integer sequence.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ……..

In mathematical terms, the sequence Fn of Fibonacci numbers is defined by the recurrence relation

```
Fn = Fn-1 + Fn-2
```

with seed values

```
F0 = 0 and F1 = 1.
```

Time Complexity: Exponential,

as every function calls two other functions.

Auxiliary space complexity:

O(n), as the maximum depth of the recursion tree is n.

Solution :

```
class Solution {
    public int fib(int n) {
        if(n==0 || n==1){
            return n;
        }
        else
            return fib(n-1)+fib(n-2);
    }
}
```

Question 2

Given an integer `n`, return *`true` if it is a power of four. Otherwise, return `false`*.

An integer `n` is a power of four, if there exists an integer `x` such that `n == 4x`.

Example 1:

```
Input: n = 16
Output: true
```

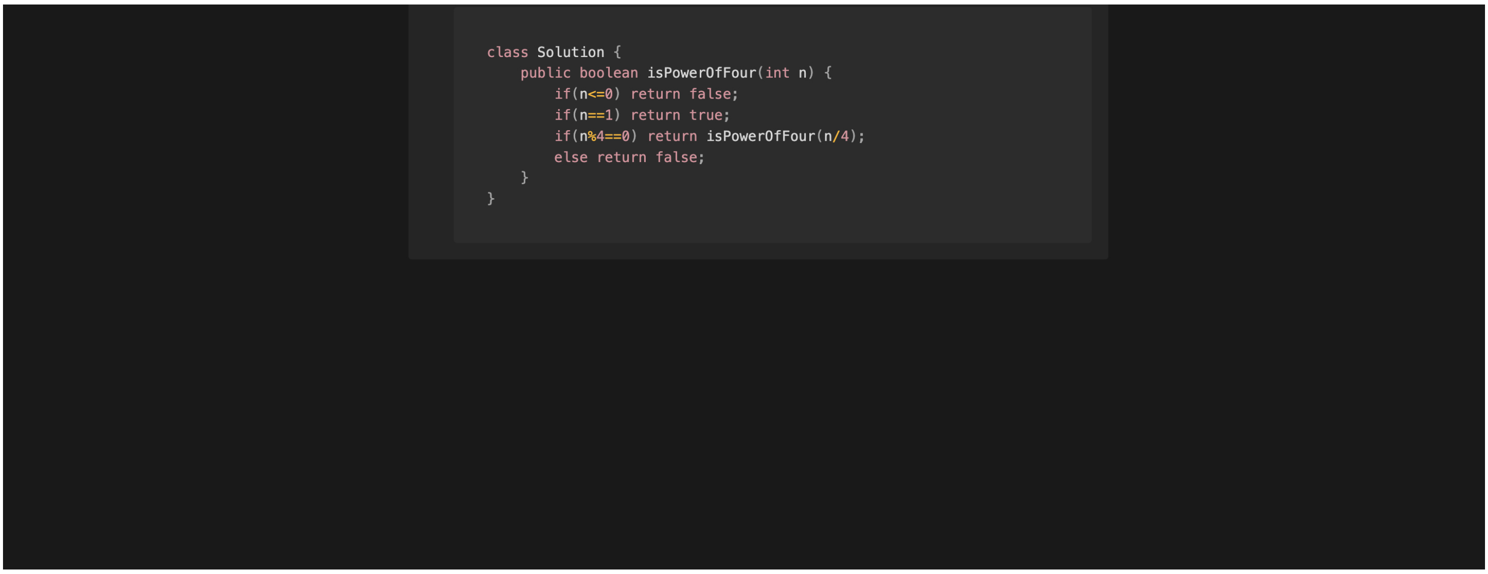
Example 2:

```
Input: n = 5
Output: false
```

Example 3:

```
Input: n = 1
Output: true
```

Solution :



```java
class Solution {
    public boolean isPowerOfFour(int n) {
        if(n<=0) return false;
        if(n==1) return true;
        if(n%4==0) return isPowerOfFour(n/4);
        else return false;
    }
}
```