# 202 Happy Number (link)

## Description

Write an algorithm to determine if a number `n` is happy.

A **happy number** is a number defined by the following process:

- Starting with any positive integer, replace the number by the sum of the squares of its digits.
- Repeat the process until the number equals 1 (where it will stay), or it **loops endlessly in a cycle** which does not include 1.
- Those numbers for which this process **ends in 1** are happy.

Return `true` *if* `n` *is a happy number, and* `false` *if not*.

### Example 1:

```
Input: n = 19
Output: true
Explanation:
1^2 + 9^2 = 82
8^2 + 2^2 = 68
6^2 + 8^2 = 100
1^2 + 0^2 + 0^2 = 1
```

### Example 2:

```
Input: n = 2
Output: false
```

### Constraints:

- $1 <= n <= 2^{31} - 1$

(scroll down for solution)

## Solution

*Language: cpp*

**Status: Accepted**

```cpp
#include <unordered_set>

class Solution {
public:
    bool isHappy(int n) {
        std::unordered_set<int> seen;

        while (n != 1 && seen.find(n) == seen.end()) {
            seen.insert(n);
            n = getNext(n);
        }

        return n == 1;
    }

private:
    int getNext(int n) {
        int sum = 0;
        while (n > 0) {
            int digit = n % 10;
            sum += digit * digit;
            n /= 10;
        }
        return sum;
    }
};
```