第202题. 快乐数

力扣题目链接(opens new window)

编写一个算法来判断一个数 n 是不是快乐数。

「快乐数」定义为：对于一个正整数，每一次将该数替换为它每个位置上的数字的平方和，然后重复这个过程直到这个数变为 1，也可能是 无限循环 但始终变不到 1。如果 可以变为 1，那么这个数就是快乐数。

如果 n 是快乐数就返回 True ；不是，则返回 False 。

示例 1：

> **输入：** n = 19
> **输出：** true
> **解释：**
> $1^2 + 9^2 = 82$
> $8^2 + 2^2 = 68$
> $6^2 + 8^2 = 100$
> $1^2 + 0^2 + 0^2 = 1$

示例 2：

> **输入：** n = 2
> **输出：** false

思路

这道题目看上去貌似一道数学问题，其实并不是！

题目中说了会 **无限循环**，那么也就是说**求和的过程中，sum会重复出现，这对解题很重要！**

正如：关于哈希表，你该了解这些！(opens new window)中所说，**当我们遇到了要快速判断一个元素是否出现集合里的时候，就要考虑哈希法了。**

所以这道题目使用哈希法，来判断这个sum是否重复出现，如果重复了就是return false， 否则一直找到sum为1为止。

判断sum是否重复出现就可以使用unordered_set。

还有一个难点就是求和的过程，如果对取数值各个位上的单数操作不熟悉的话，做这道题也会比较艰难。

(版本一)使用集合:

```
class Solution:
    def isHappy(self, n: int) -> bool:
        record = set()
        while True:
            n = self.get_sum(n)
            if n == 1:
                return True

            if n in record:
```

```
                return False
            else:
                record.add(n)


    def get_sum(self,n: int ) -> int:
        new_num = 0
        while n:
            n, r = divmod(n, 10)
            new_num += r ** 2
        return new_num
```

(版本二)使用集合

```
class Solution:
    def isHappy(self, n: int) -> bool:
        record = set()
        while n not in record:
            record.add(n)
            new_num = 0
            n_str = str(n)
            for i in n_str:
                new_num+=int(i)**2
            if new_num==1: return True
            else: n = new_num
        return False
```

(版本三)使用数组

```
class Solution:
    def isHappy(self, n: int) -> bool:
        record = []
        while n not in record:
            record.append(n)
            new_num = 0
            n_str = str(n)
            for i in n_str:
                new_num+=int(i)**2
            if new_num==1: return True
            else: n = new_num
        return False
```

(版本四)使用快慢指针

```
class Solution:
    def isHappy(self, n: int) -> bool:
        slow = n
        fast = n
        while self.get_sum(fast) != 1 and self.get_sum(self.get_sum(fast)):
            slow = self.get_sum(slow)
            fast = self.get_sum(self.get_sum(fast))
            if slow == fast:
```

```
                return False

        return True

    def get_sum(self,n: int) -> int:
        new_sum = 0
        while n:
            n,r =divmod(n,10)
            new_sum += r**2

        return new_sum
```

快慢指针的核心思想：

- **慢指针（slow）**：每次走 1 步 → `slow = f(slow)`
- **快指针（fast）**：每次走 2 步 → `fast = f(f(fast))`
- 如果**存在环**，快指针最终会"追上"慢指针（`slow == fast`）
- 如果**到达 1**，说明无环，是快乐数

(版本五)使用集合+精简

```
class Solution:
    def isHappy(self, n: int) -> bool:
        seen = set()
        while n != 1:
            n = sum(int(i) ** 2 for i in str(n))
            if n in seen:
                return False
            seen.add(n)
        return True
```

(版本六)使用数组+精简

```
class Solution:
    def isHappy(self, n: int) -> bool:
        seen = []
        while n != 1:
            n = sum(int(i) ** 2 for i in str(n))
            if n in seen:
                return False
            seen.append(n)
        return True
```