Happy Number (/problems/happy-number/)

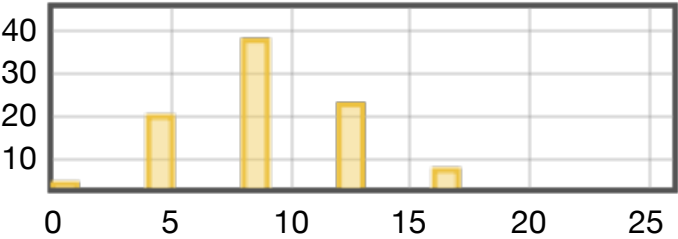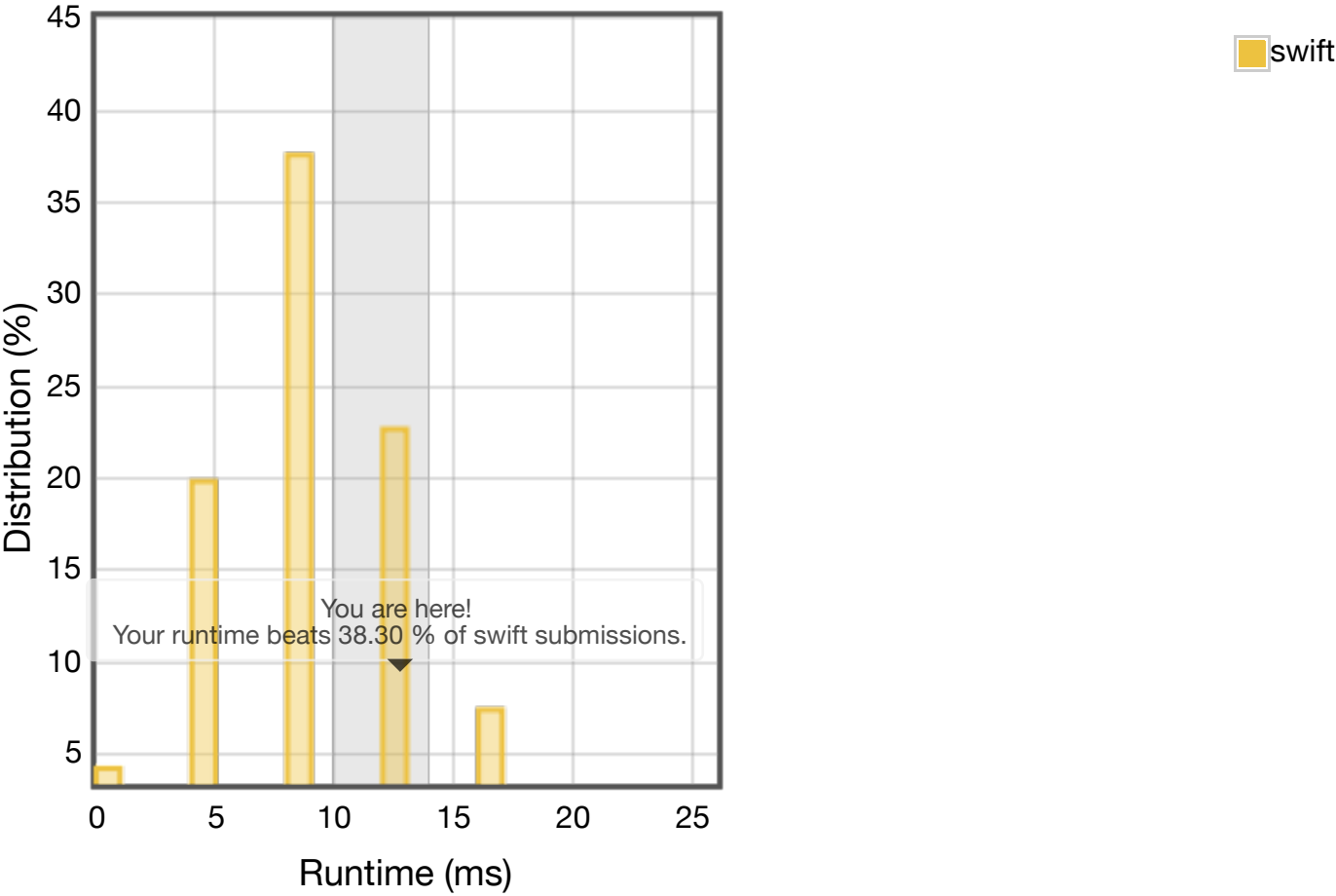# Submission Detail

401 / 401 test cases passed.

Runtime: **12 ms**
Memory Usage: **21 MB**

Status: Accepted

Submitted: **0 minutes ago**

Accepted Solutions Runtime Distribution

You are here!
Your runtime beats 38.30 % of swift submissions.

swift

Accepted Solutions Memory Distribution

Sorry. We do not have enough accepted submissions to show distribution chart.

Invite friends to challenge **Happy Number**

◀ **60**

Submitted Code: 0 minutes ago

Language: swift

[Edit Code]

```
1  1234567891011121314151617181920 21
   22232425262728293031323334353
   63738394041424344454647484950
   51525354555657585960616263646
   56667686970class Solution {
     func isHappy(_ n: Int) ->
   Bool {          //return
   isHappy00(n)          return
   isHappyProcedural(n)     }
       func isHappy00(_ n: Int)
   -> Bool {          guard n != 1
   else { return true }
         let generator =
   HappyNumberGenerator
   (initialValue: n)          var
   number: Int? = n          while
   number != nil {
   number = generator
   .nextUniqueNumber()
     //print(number ?? "nil")
       }          return
   generator.isHappy      }
    func isHappyProcedural(_ n:
   Int) -> Bool {
   precondition(n >= 0)
         var value = n
   //print(value)          var
   computedValues: Set<Int> = []
```

```swift
                while (value !
= 1 && !computedValues
.contains(value)) {
    computedValues.insert(value
)            value = value
.sumfOfDigitsSquared
    //print(value)          }
      return value == 1
}}class HappyNumberGenerator
{     private var
computedValues: Set<Int>
private var currentValue: Int
      init(initialValue: Int
) {         currentValue =
initialValue
computedValues = []      }
    func nextUniqueNumber() ->
Int? {         currentValue =
currentValue
.sumfOfDigitsSquared
if computedValues.contains
(currentValue) {
return nil         } else {
         computedValues
.insert(currentValue)
    return currentValue
  }    }          var isHappy:
Bool {        return
computedValues.contains(1)
 }}extension Int {     var
sumfOfDigitsSquared: Int {
    let numberAsString =
String(self)         var
result = 0        for char in
numberAsString {
result += Int(pow(Float
(String(char))!,2))        }
      return result
}}XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXX
```

Back to problem (/problems/happy-number/)