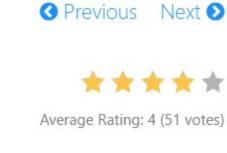
166. Fraction to Recurring Decimal

March 10, 2016 | 37.3K views



format. If the fractional part is repeating, enclose the repeating part in parentheses.

Given two integers representing the numerator and denominator of a fraction, return the fraction in string

Example 1:

```
Input: numerator = 1, denominator = 2
 Output: "0.5"
Example 2:
```

```
Input: numerator = 2, denominator = 1
 Output: "2"
Example 3:
```

Input: numerator = 2, denominator = 3

```
Output: "0.(6)"
```

Summary

Hints

This is a straight forward coding problem but with a fair amount of details to get right.

2. Try a long division on $\frac{4}{9}$, the repeating part is obvious. Now try $\frac{4}{333}$. Do you see a pattern?

Solution

3. Be wary of edge cases! List out as many test cases as you can think of and test your code thoroughly.

1. No scary math, just apply elementary math knowledge. Still remember how to perform a long division?

Approach 1: Long Division Intuition

The key insight here is to notice that once the remainder starts repeating, so does the divided result. 0.16

6)1.00

 $\Leftarrow remainder = 1$, mark 1 as seen at position = 0.

```
$$
                 \Leftarrow remainder = 4, mark 4 as seen at position = 1.
                 \Leftarrow remainder = 4 was seen before at position = 1,
                    so the fractional part starts repeating at position = 1 \Rightarrow 1(6).
Algorithm
You will need a hash table that maps from the remainder to its position of the fractional part. Once you
```

The remainder could be zero while doing the division. That means there is no repeating fractional part and

 $\frac{-1}{4}$ or $\frac{1}{-4}$

4

Comments: 19

the position from the table.

Here are some good test cases:

if (numerator == 0) { return "0";

Numerator is zero.

you should stop right away. Just like the question Divide Two Integers, be wary of edge cases such as negative fractions and nasty extreme case such as $\frac{-2147483648}{-1}$.

found a repeating remainder, you may enclose the reoccurring fractional part with parentheses by consulting

Explanation Test case

Divisor is 0, should handle it by throwing an exception but here we ignore for simplicity $\frac{1}{0}$ sake. Answer is a whole integer, should not contain the fractional part. Answer is 0.5, no recurring decimal.

Both numerator and denominator are negative, should result in positive fraction. $\frac{-2147483648}{-1}$ Beware of overflow if you cast to positive. **С**ору Java

public String fractionToDecimal(int numerator, int denominator) {

StringBuilder fraction = new StringBuilder();

Type comment here... (Markdown is supported)

// If either one is negative (not both)

One of the numerator or denominator is negative, fraction is negative.

```
if (numerator < 0 ^ denominator < 0) {
              fraction.append("-");
  9
          // Convert to Long or else abs(-2147483648) overflows
  10
          long dividend = Math.abs(Long.valueOf(numerator));
  11
          long divisor = Math.abs(Long.valueOf(denominator));
  12
          fraction.append(String.valueOf(dividend / divisor));
  13
          long remainder = dividend % divisor;
  14
          if (remainder == 0) {
  15
              return fraction.toString();
  16
  17
          fraction.append(".");
  18
          Map<Long, Integer> map = new HashMap<>();
  19
          while (remainder != 0) {
  20
              if (map.containsKey(remainder)) {
  21
                  fraction.insert(map.get(remainder), "(");
  22
  23
                  fraction.append(")");
                  break;
  24
  25
              map.put(remainder, fraction.length());
  26
              remainder *= 10;
  27
Rate this article: * * * *
 Previous
                                                                                                          Next ()
```

Sort By ▼

```
Preview
                                                                                           Post
ChangRutgers ★ 21 ② December 10, 2017 6:52 AM
what's the space complexity?
21 A Y Share Share Reply
SHOW 3 REPLIES
jabberwok ★4 ② September 7, 2017 6:19 PM
@Geminiiiiii
I can't read the original article because my subscription ran out, so I can't explain in details what is
wrong with it, the only thing I remember that it used HashMap which was a huge overkill performance-
wise. My idea was that all fractions are either finite like 0.123 or cyclic like 0.123333...(3). Cyclic fractions
                                           Read More
6 \Lambda 🗸 🗗 Share 🥱 Reply
amaroraiko ★ 64 ② October 1, 2019 12:53 AM
The unit tests around integer min and max values are brutal.
SHOW 1 REPLY
ueabu ★ 3 ② November 14, 2019 11:50 AM
probably a dumb question but why do you multiply the remainder by 10 each time? I revised long
division and I can't see anywhere that happens.
SHOW 1 REPLY
calvinchankf * 2925 • June 20, 2019 9:25 AM
Here is my approach which beats 90.96% in python
math + hashtable

    do long division

                                           Read More
1 A V C Share   Reply
oyugioho ★ 1 ② September 6, 2017 12:04 PM
```

HT_Wang ★ 487 ② July 24, 2019 8:11 AM Awesome coding question 1 A V C Share Reply

1 A V C Share Reply

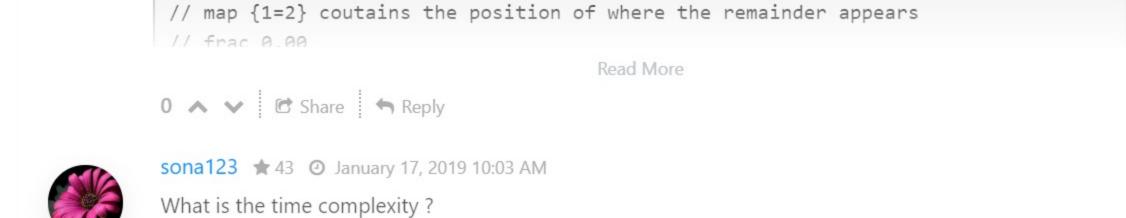
SHOW 1 REPLY

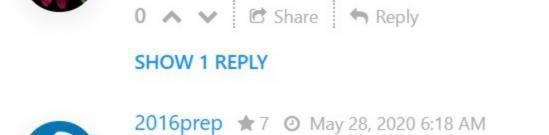
magusliuj ★ 4 ② February 15, 2019 4:00 AM // for test case 1/333 // frac 0.0

I think he is referring to the technique of loop detection that we all know how to do for the question

for one and one at a time for another. When they are equal, we detect a loop, etc.

"Find the node where the loop starts for a singly linked list". Basically, try to get reminders two at a time





(1 2)

37/39 test cases passed during mock session, only missed doing the math with longs