Repeating Decimals to Fractions

Published by Matt in Java

math numbers strings

Performing division on a fraction often results in an infinitely repeating decimal.

```
1/3=.3333333...  1/7=.142857142857...
```

Create a function that takes a decimal in string form with the repeating part in parentheses and returns the equivalent fraction in string form and in lowest terms.

Examples

```
fractions("0.(6)") → "2/3"

fractions("1.(1)") → "10/9"

fractions("3.(142857)") → "22/7"

fractions("0.19(2367)") → "5343/27775"

fractions("0.1097(3)") → "823/7500"
```

Notes

N/A

LOGIC

Firstly I had little knowledge whatsoever on the process involving converting the decimal component into a fraction in a software programming context. The exception is if the fractional component is short such as 0.25.. But even then, the only tactic I know is if creating a mapping or alternatively try to perform ½, ¼..... until I each the answer.
This can become extensive notably if the fraction was 0.375 (3/8).
Unfortunately I had no idea if any simple techniques occurred, so I had to perform research on the internet for the mathematical principles.

I used this (Maths tutorial) as an example in order to derive some logic:

https://www.youtube.com/watch?v=a9WmkiamMdA
I think it’s a very good starting point in ascertaining the core logic...

APLLYING VIDEO PRINCIPLES FOR PSEUDO CODE ****
For number such as 0.75

Need to count number digits to right of decimal point

then multiply this by 1 = 100(multipliedBy)

Then place numerator over   numerator/multipliedBy

We know the top can be divided maximum by 100 (if it is a proper fraction).

for loop decrement variable multipliedBy (100=>1)

numerator / multipliedBy (100 => 1)

denominator/   multipliedBy 100 =>1

Stop when both render an integer..

//once that's done, need to go again this time divisor is

(numerator => 1)

The question is when to stop?

Can not perform a division of the numerator and denominator components

Now compare against original decimal...

Java might struggle to manually perform the division of the fraction depending on the decimal component length.

So can use a Big calculator online with high precision

And perform subtraction of decimal inputted in fraction method with the fraction...

****

Also I am aware that Pi is an irrational number since the decimal component is infinitely long. So this raises another question... We know that 22/7 is an approximation of Pi..... And in the test case, it has shown the recurring portion being 6 digits wide.
But I will need to write code once again (similar to addition challenge) to try and push the decimal part to its limit. This might not be an issue if the decimal part is derived and placed straight into long variable...
If there is a recursion technique, I will need to ensure I can prolong StackOverFlow.

Also testing will be fascinating since I am in full control of the test scenarios which operate on these boundaries:
As far as I can see there will be few scenarios:

-fractional digit component repeating endlessly from offset 10/9 = 1.1111111111

-finite fractional component before single repeat digit commences (more than once) = 823/7500 = 0.1097**(3)**

-finite fractional component before multiple repeat digit commences (more than once) = 5343 / 27775   = 0.19(2367)

-infinite fractional component such as pi or any irrational number. It can be seen that in his example, he has shown   3.(142857). I am extremely confused on this, so like the other scenarios, I have placed 22/7 in the calculator (scientific). I can see it does repeat every 6 digits... 3.142857142857142857142857142857... My query is how to produce the data for scenarios since most resources such as Windows calculator will round up at the least significant digit in precision.

But we know with this scenario, pi is irrational, so also will mean the fraction will be infinitely long to get exact answer... Hence the question has

provided the small reoccurring number portion..
It slightly creates a twist to the question, as to what will be the maximum length of the fractional component in decimal which is applicable to fulfil this challenge.

My understanding is:

Hypothetically, we know the entire number is confined to limit of String which is:
2,147,483,647 wide

So if the part prior to mantissa (and including) is three digit long, the remaining bit (in red) is 2,147,483,644 wide.. (for instance 62.XXXXXX
So in the worst case scenario, if there were two occurrences of lets say

123457989799 (imagine this last 9 digit stopped at 2,147,483,644 / 2 and the sequence started again .1234..... (up to index 2,147,483,647)
There is every possibility that the portion after decimal point could re-occur from this point onwards.... (1073741822 => up to index 2,147,483,647).
I am just finding this enormously unfathomable and can not think of any scenario to prove or disprove this, until I start experimenting...

To be honest, I did not even know how to perform a google search to ask correct question. And this is first time I am asking Google to support a deep part of my logic.

I placed a google search as follows which in itself is so ambiguous.
“longest fraction with recurring twice”

And very fortunately I found something in forum which answered the query, but its something I will just have to accept being a non-mathematician.

https://math.stackexchange.com/questions/4200959/is-there-a-fraction-in-which-the-decimal-representation-never-terminates-nor-rep

If you divide by a number, which is not a multiple of 2 or 5 you always get a periodic number. If you divide 1 by 65537 there are only 65536 possible remainders , so once you hit the same remainder the second time, the digits repeat. To convince you try

it out with 1/7 or 1/17 the maximum period is 6 or 16

I also found another site:
https://math.stackexchange.com/questions/287805/the-longest-repeating-decimal-that-can-be-created-from-a-simple-fraction/3529512#3529512?newreg=a09764eaad8a43808bd94de02c59b496

So, after reading this, my first port of call is understanding the correct terminology to try and break this solution down. This will help me create the correct variables in my code

repetend = this appears to be the repeating part of the fraction
period = . length of the repetend before repetition occurs again in the part after decimal point. For instance, in perspective of the comment in forum (see blue), we expect the repetend to be 6 or 16 for 1/7 or 1/17 respectively.

periodic number = this is a number which has a level recurring in the decimal but it can be expressed as a fraction.. For instance, 0.8888888888(infinitely) can be written as 8/9

But it is stated in the articles that repetend is unbound.. We know real decimal portion of pi is not repetend (but it is shown in the example question).

My main aim is to understand this logic:
If you divide by a number, which is not a multiple of 2 or 5 you always get a periodic number. If you divide 1 by 65537 there are only 65536 possible remainders , so once you hit the same remainder the second time, the digits repeat. To convince you try it out with 1/7 or 1/17 the maximum period is 6 or 16

I will choose 3/57 =
0.052631578947368421**05263157894737** = 14 digit wide repetend

I will choose 1/23 = 0.0434782608695652173913**0434782609** = 22 digit wide repetend

I am not entirely sure of the initial question nor how this logic in blue will assist.
In layman terms, I need to program this code so that period of repetend is less than 2,147,483,647 – (characters in length of the whole number + decimal point) / 2
So I need to restrict any inputted samples with repetend which has a period no greater than 1073741822

I found this useful information in my research which seems to be level of logic to implement my code...

https://thestarman.pcministry.com/math/rec/RepeatDec.htm

Observations on Repeating Decimals

- **If** a fraction is composed of repeating decimal digits, the number of digits that repeat will ***never be greater than*** **one subtracted from the fraction's denominator**: Either **k = d - 1**, or **k < d - 1** (where **k** is the *periodic length* of the repeating decimal digits).

It means that I should ensure that when I choose sample denominator in order to generate a decimal, it should not be any greater than 1073741821

I know I will not be able to perform a calculation in Java, hence this is why the decimal part is presented as a String. We can from this resource:

https://engert.us/erwin/miscellaneous/Reciprocals%20of%20prime%20numbers.pdf where they are finding recripical of a prime number..
1/4961311, 826885 (this is the period of the repetend)..
So strictly speaking, if I was able to find a copy of this value from a source, I could input this a String in Java and can reach these limits and more....

Here are other sites that I used for research:
https://pballew.blogspot.com/2022/08/repeating-decimal-periods-and-patterns.html

This will generate decimal number with precision of 9,999 digits...
I will use this to create a suitable sample, examine the repetend.....
https://www.calculator.net/big-number-calculator.html?cx=1&cy=65537&cp=9999&co=divide
I have tried one sample so far in conjunction with 1/Prime number

Reciprocals of prime numbers, number of repeating digits

This is part two of the other file engert.us/erwin/miscellaneous/Repeating digits list for prime reciprocals.pdf. This file is for the prime numbers between 3 through 5,000,000.

1/3, 1	1/277, 69	1/641, 32	1/1031, 103	1/1451, 290

I am expecting period of repetend to be 69 digits wide..
So I will set precision to 69 in calculator.
And then set the precision to 128 and see if its exactly recurring portion twice in succession.

Big Number Calculator

The calculator below can compute very large numbers. Acceptable formats include: integers, decimal, or the E-notation form of scientific notation, i.e. 23E18, 3.5e19, etc.

Result

0.00361010830324909747292418772563176895306859205776173285198555956678 7

X = 1

Y = 277

Precision: 69 digits after the decimal place in the result

Click the buttons below to calculate

X + Y | X − Y | X × Y | X / Y | X^Y | √X | X ^ 2 | X! | MOD | GCD | LCM

Big Number Calculator

The calculator below can compute very large numbers. Acceptable formats include: integers, decimal, or the E-notation form of scientific notation, i.e. 23E18, 3.5e19, etc.

Result

0.00361010830324909747292418772563176895306859205776173285198555956678700
3610108303249097472924187725631768953068592057761732851985559566787

X = 1

Y = 277

Precision: 138 digits after the decimal place in the result

Click the buttons below to calculate

X + Y | X − Y | X × Y | X / Y | X^Y | √X | X ^ 2 | X! | MOD | GCD | LCM

Unfortunately I could not find a calculator online which can generate precision greater than 9999 digits.

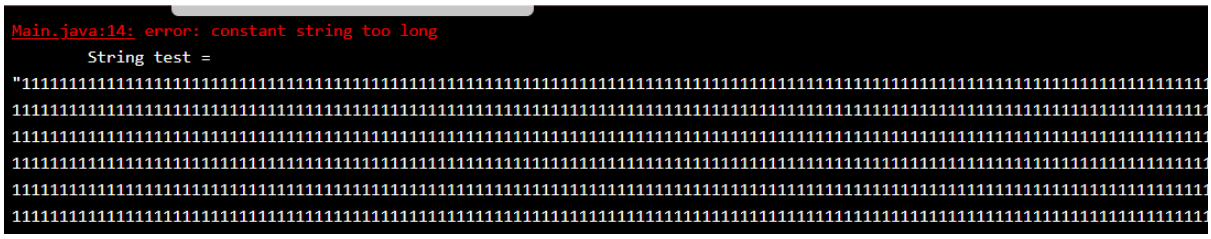
I found this research extremely beneficial...
I can understand the limitations from this highly test driven exercise and understand that my inputs are limited with heterogenuous numbers in period of repetend.

But, I now understand what a periodic number is (can be represented as a fraction) irrespective of having recurring fractional component. So there is no reason why I could not create a test sample consisting of 1.111111111111111111111 (with over million digits in the mantissa portion)...

I will first attempt in Java language to try and generate subsection of Strings at intervals and then concatenate them to form 1,000,000.

Alternatively I found this resource online,
https://www.unit-conversion.info/texttools/random-string-generator/
but it has only been able to generate a number 100,000 digits wide.. (which is still a considerable amount).. And it will allow me to test several scenarios in the following documentation
https://engert.us/erwin/miscellaneous/Reciprocals%20of%20prime%20numbers.pdf).

Having tried on Java, it rendered following output:



I conducted further research online and it informed me that limitation of a String is
2 power 16 = 65,536

This is something I was unaware of at all....
So this effectively means that I need to state my limitations again for the period of repetend:

In layman terms, I need to program this code so that period of repetend is less than
65,536 – (characters in length of the whole number + decimal point) / 2
So I need to restrict any inputted samples with repetend which has a period no greater than (65536 – (1+1)) /2 = 32 767 in which fraction(**1.X**XXXXXX).
It means that I should ensure that when I choose sample denominator in order to generate a decimal (using online calculator), it should not be any greater than 32 767.
Otherwise Java will not be able to identify the period of repetend for two successive repetends....

I have created a String of length 32760 and Java compiler is content..

My aim initially will be to understand this. We know the repetend has to be non-repeating, otherwise it would not be terminating. This is foundation of my first set of test cases in which I can create a fraction and then use the decimal conversion back into my code.

Key points about terminating fractions:

Denominator key:
To check if a fraction will terminate, look at its denominator in simplest form. If the denominator only has prime factors of 2 and/or 5, the fraction will terminate