

How to find missing number in a string of numbers with no separator in python?

Asked today Active today Viewed 59 times



4



Given a string consisting of some numbers, not separated by any separator. The numbers are positive integers and the sequence increases by one at each number except the missing number. The task is to find the missing number. The numbers will have no more than six digits. Print -1 if input sequence is not valid.



Input : 89101113

Output : 12

Input : 9899101102

Output : 100

Input : 596597598600601602:

Output : 599

Input : 909192939495969798100101

Output : 99

Input : 11111211311411511

Output : -1

As far as I am aware, looping considers each character at a time. Is this possible to achieve in python ?

python

python-3.x

edited 2 hours ago



wjandrea

12.5k 5 24 47

asked 2 hours ago



Fresher

75 4

3 The way you're presenting is not a sequence of *numbers*, it's a sequence of *digits* or a sequence of numbers that have at least one digit but possibly more - if more than how many digits?. If a number in a string has more than a single digit then you have to decide where one number stops and the other begins - otherwise there's no way to know if the string is one number or a sequence of integers. - if you assume 9899 has 98 and 99, you have to decide how you'll know it's not 989 and 9 or 9899 as single number, so you have to limit your operations somehow.

– dmityro 2 hours ago

1 Great discussion @dmityro, that was going to be along the lines of my thinking out loud: you have to determine what the first number is somehow and eliminate all possibilities. – Charlie G 2 hours ago

3 Answers

Active

Oldest

Votes

You can't simply loop over set sizes, you have parse, using a changing size.

3

Take `n` characters from the start of the string, convert them to `int`, add one, and check if it matches the new start of the string. If not, record that, add the characters back to the start of the string and try again. Repeat. If the number of non-matches gets greater than 1, stop, add one to `n`, and try the whole process again. If you get through a whole loop at `n` and the number of non-matches is 1, then you've found the correct answer.

```
def split(sequence, x):
    return sequence[:x], sequence[x:]

def parse(digits):
    """
    Try to parse "digits" into numbers, and find the missing one.

    The numbers will have no more than six digits.
    Return -1 if "digits" isn't parseable or isn't missing one.

    >>> parse("89101113") # 8, 9, 10, (12), 13
    12
    >>> parse("9899101102") # 98, 99, (100), 101, 102
    100
    >>> parse("596597598600601602") # 596, 597, 598, (599), 600, 601, 602
    599
    >>> parse("909192939495969798100101") # 90, ...
    99
    >>> parse("11111211311411511") # Looks like "111, ..." but isn't
    -1
    """
    for n in range(1, 7):
        expected, remainder = split(digits, n)
        failures = []
        while len(failures) <= 1 and remainder:
            expected = str(int(expected) + 1)
            actual, remainder = split(remainder, len(expected))
            if actual != expected:
                failures.append(expected)
                remainder = actual + remainder # Re-parse
        if len(failures) == 1:
            return int(failures[0])
    return -1
```


answered 1 hour ago



wjandrea

12.5k ● 5 ● 24 ● 47

I did similar solution but my response failed when the sequence starts from `910111213` or `99100101102103` or `999100010011002`. Working thru that before I post. – [Joe Ferndz](#) 1 hour ago



I don't think this is the most readable, but seems to be working. My basic logic was to use `str.index()` and a continually shortened string to check if the next expected number (for a given starting digit size) came at position `0`. Do this until the original string is matched (in which case there are no missing numbers), but allow one chance for the expected number to be incremented (and then if the starting string gets matched, return that skipped number). If more than one number needs to be skipped, move on to increasing the starting digit size. Once the starting digit length gets too long (more than half of the input digits), call it quits and return `-1`. Some more comments below, happy to try and clean up or explain more:

```
def missing_digits(number):
    s = str(number)
    l = 1
    while True:
        if l > len(s)/2:
            # LOOP to increase starting number size
            # return negative one if starting number gets
            # too large
            return -1
        nextnum = int(s[:l])
        s_test = s[len(str(nextnum)):]
        skips = 0
        lastskip = None
        if no skips)
            while True:
                if len(s_test) == 0:
                    # LOOP to check each sequential number
                    # return either None or the last skipped number
                    if end of input reached
                        return lastskip
                    if skips > 1:
                        # stop if there is more than one number skipped
                        break
                    nextnum = nextnum + 1
                    try:
                        if s_test.index(str(nextnum)) == 0:
                            # use .index() to see if
                            # nextnum occurs first
                            s_test = s_test[len(str(nextnum)):]
                            # continue if so and shorten
                            # s_test
                            continue
                        except:
                            # if this fails or .index()
                            # of nextnum isn't first
                            pass
                            lastskip = nextnum
                            # record the skip
                            skips += 1
                    l += 1
```

Testing:

```
for i in [89101113,
          9899101102,
          596597598600601602,
          909192939495969798100101,
          11111211311411511,
          123456]: # extra with no missing digits
    print(missing_digits(i))
```

Output:

CARRY_ARRAY = {-8, -89, -899, -8999, -89999, -899999, -7, -88, -898, -8998, -89998, -899998}

```
def check_lack(string):
    for b in range(1, 7):
        start, index = int(string[:b]), b
        while index < len(string):
            num = string[index:index + b]
            difference = int(num) - start
            if difference in CARRY_ARRAY:
                b += 1
                continue
            if difference == 1:
                start, index = start + 1, index + b
            elif difference == 2:
                return int(num) - 1
            else:
                break
    return -1
```

answered 11 mins ago



scp10011

1



New contributor