# SPOJ PALIN Python Solution

By:
Adam Gincel
Matthew Gomez
Alex Massenzio

# Description of Problem

- Link: http://www.spoj.com/problems/PALIN/
- Given a number find the next highest palindrome
- A palindrome is a number which is the same when read left to right and right to left

### Input

The first line contains integer *t*, the number of test cases. Integers *K* are given in the next *t* lines.

### Output

For each *K*, output the smallest palindrome larger than *K*.

# Example Input / Output

## Example

**Input:**

```
2
808
2133
```

**Output:**

```
818
2222
```

**Warning: large Input/Output data, be careful with certain languages**

# First Idea - The Inefficient Solution

- Our first approach to this problem was very simple:
  - When given a number, check if it’s a palindrome.
  - If not, increment by one, check if that’s a palindrome.
  - Repeat until palindrome is found.
- This was very inefficient, and was not fast enough to pass SPOJ’s tests.

# The Better Solution - Overview

- Even
  - 4321
  - 1234
  - All 9’s
- Odd
  - 54321
  - 12345
  - All 9’s
  - 12945

# The Better Solution - Implementation

```
7  def findNextPalindrome(number):
8      size = len(number) #number is a string
9      if size % 2 == 1: #if our number is odd, get the center digit
10         center = number[int(size / 2)]
11     else:
12         center = ''
13     left = number[0:int(size / 2)]
14     right = ''.join(reversed(left))
15     reversedNumber = left + center + right
16     if reversedNumber > number: #if it's greater
17         print(reversedNumber)
18     else:
19         if center:
20             if center < '9': #increment center by 1
21                 center = str(int(center) + 1)
22                 print(left + center + right)
23                 return
24             else:
25                 center = '0'
26         if left == len(left) * '9': #if all 9s
27             print('1' + (len(number) - 1) * '0' + '1') #print in the form 10001
28         else:
29             left = increment(left) #otherwise increment left half then print
30             print(left + center + ''.join(reversed(left)))
31
32 def increment(l):
33     left = list(l)
34     last = len(left) - 1
35     while left[last] == '9': #can't be all 9s, so this will not go out of range
36         left[last] = '0'
37         last -= 1
38     left[last] = str(int(left[last]) + 1) #increment
39     return "".join(left)
40
41 numCases = int(input())
42 while numCases > 0:
43     numCases -= 1
44     findNextPalindrome(input())
```

# Explanation - Even Cases

- 4321
  - Easiest Case
  - Flip left half and append onto the right half.
  - 4334
- 1234
  - 4321 case does not work here (1234 > 1221)
  - Need to increment the number closest to the center on the left before flipping.
  - 1331
- All 9's (9999)
  - Second easiest case
  - Palindrome will 1 ((The number of 9's - 1) 0's) 1
  - 10001

# Explanation - Odd Cases

- 54321
  - Same as the even 4321 case (except don't touch the center number)
  - 54345
  - This number is greater than the starting number, so it is our answer.
- 12345
  - Flip produces 12421 which is less than our starting number, so it isn't our answer
  - Increment the middle number, then flip.
  - 12421
- All 9's
  - Same as before; 1 ((The number of 9s - 1) 0s) 1 (ie 100001)
- 12945
  - Similar to 12345 case, except we have to set the center digit to 0
  - Once we've done that, increment the left half and flip
  - 13031