

Home	Algorithms	DS	GATE	Interview Corner	Q&A	C	C++	Java	Books	Contribute	Ask a Q	About
Array	Bit Magic	C/C++	Articles	GFactS	Linked List	MCQ	Misc	Output	String	Tree	Graph	

## Given a number, find the next smallest palindrome

Given a number, find the next smallest palindrome larger than this number. For example, if the input number is "2 3 5 4 5", the output should be "2 3 6 3 2". And if the input number is "9 9 9", the output should be "1 0 0 1".

The input is assumed to be an array. Every entry in array represents a digit in input number. Let the array be 'num[]' and size of array be 'n'

There can be three different types of inputs that need to be handled separately.

- 1) The input number is palindrome and has all 9s. For example "9 9 9". Output should be "1 0 0 1"
- 2) The input number is not palindrome. For example "1 2 3 4". Output should be "1 3 3 1"
- 3) The input number is palindrome and doesn't have all 9s. For example "1 2 2 1". Output should be "1 3 3 1".

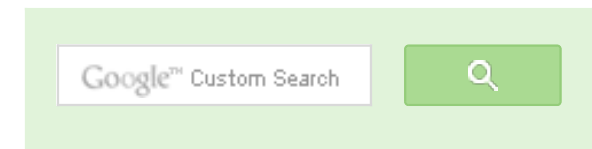
Solution for input type 1 is easy. The output contains  $n + 1$  digits where the corner digits are 1, and all digits between corner digits are 0.

Now let us first talk about input type 2 and 3. How to convert a given number to a greater palindrome? To understand the solution, let us first define the following two terms:

**Left Side:** The left half of given number. Left side of "1 2 3 4 5 6" is "1 2 3" and left side of "1 2 3 4 5" is "1 2"

**Right Side:** The right half of given number. Right side of "1 2 3 4 5 6" is "4 5 6" and right side of "1 2 3 4 5" is "4 5"

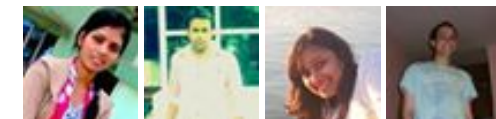
To convert to palindrome, we can either take the mirror of its left side or take mirror of its right side. However, if we take the mirror of the right side, then the palindrome so formed is not guaranteed to be next larger palindrome. So, we must take the mirror of left side and copy it to



GeeksforGeeks



53,525 people like [GeeksforGeeks](#).



Facebook

[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

right side. But there are some cases that must be handled in different ways. See the following steps.

We will start with two indices  $i$  and  $j$ .  $i$  pointing to the two middle elements (or pointing to two elements around the middle element in case of  $n$  being odd). We one by one move  $i$  and  $j$  away from each other.

**Step 1.** Initially, ignore the part of left side which is same as the corresponding part of right side. For example, if the number is “8 3 **4 2 2 4** 6 9”, we ignore the middle four elements.  $i$  now points to element 3 and  $j$  now points to element 6.

**Step 2.** After step 1, following cases arise:

**Case 1:** Indices  $i$  &  $j$  cross the boundary.

This case occurs when the input number is palindrome. In this case, we just add 1 to the middle digit (or digits in case  $n$  is even) propagate the carry towards MSB digit of left side and simultaneously copy mirror of the left side to the right side.

For example, if the given number is “1 2 9 2 1”, we increment 9 to 10 and propagate the carry. So the number becomes “1 3 0 3 1”

**Case 2:** There are digits left between left side and right side which are not same. So, we just mirror the left side to the right side & try to minimize the number formed to guarantee the next smallest palindrome.

In this case, there can be **two sub-cases**.

**2.1)** Copying the left side to the right side is sufficient, we don't need to increment any digits and the result is just mirror of left side. Following are some examples of this sub-case.

Next palindrome for “7 **8** 3 3 2 2” is “7 8 3 3 8 7”

Next palindrome for “1 2 **5** 3 2 2” is “1 2 5 5 2 1”

Next palindrome for “1 4 **5** 8 7 6 7 8 3 2 2” is “1 4 5 8 7 6 7 8 5 4 1”

How do we check for this sub-case? All we need to check is the digit just after the ignored part in step 1. This digit is highlighted in above examples. If this digit is greater than the corresponding digit in right side digit, then copying the left side to the right side is sufficient and we don't need to do anything else.

**2.2)** Copying the left side to the right side is NOT sufficient. This happens when the above defined digit of left side is smaller. Following are some examples of this case.

## JDBC to Informix

 [progress.com/Informix](https://progress.com/Informix)

Supports Latest Data Connections  
J2EE Certified. Download Eval Now!



## Popular Posts

[All permutations of a given string](#)

[Memory Layout of C Programs](#)

[Understanding “extern” keyword in C](#)

[Median of two sorted arrays](#)

[Tree traversal without recursion and without stack!](#)

[Structure Member Alignment, Padding and](#)

[Data Packing](#)

[Intersection point of two Linked Lists](#)

[Lowest Common Ancestor in a BST.](#)

[Check if a binary tree is BST or not](#)

[Sorted Linked List to Balanced BST](#)

Next palindrome for "7 1 3 3 2 2" is "7 1 4 4 1 7"

Next palindrome for "1 2 3 4 6 2 8" is "1 2 3 5 3 2 1"

Next palindrome for "9 4 1 8 7 9 7 8 3 2 2" is "9 4 1 8 8 0 8 8 1 4 9"

We handle this subcase like Case 1. We just add 1 to the middle digit (or digits in case n is even) propagate the carry towards MSB digit of left side and simultaneously copy mirror of the left side to the right side.

```
#include <stdio.h>
```

```
// A utility function to print an array
```

```
void printArray (int arr[], int n);
```

```
// A utility function to check if num has all 9s
```

```
int AreAll9s (int num[], int n );
```

```
// Returns next palindrome of a given number num[].
```

```
// This function is for input type 2 and 3
```

```
void generateNextPalindromeUtil (int num[], int n )
```

```
{
```

```
    // find the index of mid digit
```

```
    int mid = n/2;
```

```
    // A bool variable to check if copy of left side to right is sufficient
```

```
    bool leftsmaller = false;
```

```
    // end of left side is always 'mid -1'
```

```
    int i = mid - 1;
```

```
    // Beginning of right side depends if n is odd or even
```

```
    int j = (n % 2)? mid + 1 : mid;
```

```
    // Initially, ignore the middle same digits
```

```
    while (i >= 0 && num[i] == num[j])
```

```
        i--, j++;
```

```
    // Find if the middle digit(s) need to be incremented or not (or carry
```

```
    // side is not sufficient)
```

```
    if ( i < 0 || num[i] < num[j])
```

```
        leftsmaller = true;
```

```
    // Copy the mirror of left to right
```

```
    while (i >= 0)
```

```
{
```

```
        num[j] = num[i];
```

```
        j++;
```



## Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 15 minutes ago

**Aman** Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 55 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 58 minutes ago

**Sanjay Agarwal** bool

tree::Root\_to\_leaf\_path\_given\_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

**GOPI GOPINATH** @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

**newCoder3006** If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

AdChoices

► [C++ Code](#)

► [Programming C++](#)

► [Numbers Number](#)

AdChoices

```

    i--;
}

// Handle the case where middle digit(s) must be incremented.
// This part of code is for CASE 1 and CASE 2.2
if (leftsmaller == true)
{
    int carry = 1;
    i = mid - 1;

    // If there are odd digits, then increment
    // the middle digit and store the carry
    if (n%2 == 1)
    {
        num[mid] += carry;
        carry = num[mid] / 10;
        num[mid] %= 10;
        j = mid + 1;
    }
    else
        j = mid;

    // Add 1 to the rightmost digit of the left side, propagate the
    // towards MSB digit and simultaneously copying mirror of the
    // to the right side.
    while (i >= 0)
    {
        num[i] += carry;
        carry = num[i] / 10;
        num[i] %= 10;
        num[j++] = num[i--]; // copy mirror to right
    }
}

// The function that prints next palindrome of a given number num[]
// with n digits.
void generateNextPalindrome( int num[], int n )
{
    int i;

    printf("\nNext palindrome is:\n");

    // Input type 1: All the digits are 9, simply o/p 1
    // followed by n-1 0's followed by 1.
    if( AreAll9s( num, n ) )
    {

```

```

        printf( "1 " );
        for( i = 1; i < n; i++ )
            printf( "0 " );
        printf( "1" );
    }

    // Input type 2 and 3
    else
    {
        generateNextPalindromeUtil ( num, n );

        // print the result
        printArray (num, n);
    }
}

```

// A utility function to check if num has all 9s

```

int AreAll9s( int* num, int n )
{
    int i;
    for( i = 0; i < n; ++i )
        if( num[i] != 9 )
            return 0;
    return 1;
}

```

/\* Utility that prints out an array on a line \*/

```

void printArray(int arr[], int n)
{
    int i;
    for (i=0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
}

```

// Driver Program to test above function

```

int main()
{
    int num[] = {9, 4, 1, 8, 7, 9, 7, 8, 3, 2, 2};

    int n = sizeof (num) / sizeof(num[0]);

    generateNextPalindrome( num, n );

    return 0;
}

```

[► Code Number](#)

[► Even Number](#)

[► Print Number](#)

AdChoices 

[► Print a Number](#)

[► Copy Number](#)

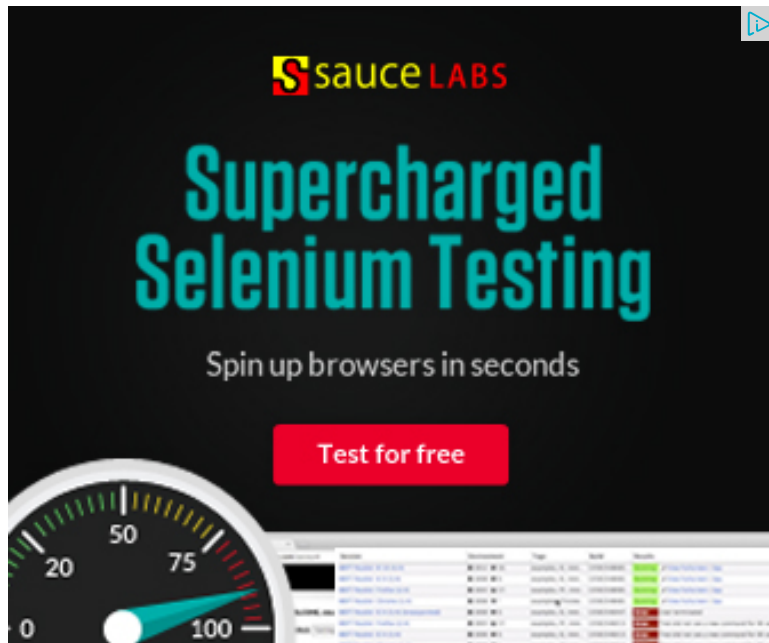
[► Java Array](#)

Output:

Next palindrome is:

9 4 1 8 8 0 8 8 1 4 9

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.



## Related Topics:

- [Backtracking | Set 8 \(Solving Cryptarithmic Puzzles\)](#)
- [Tail Recursion](#)
- [Find if two rectangles overlap](#)
- [Analysis of Algorithm | Set 4 \(Solving Recurrences\)](#)
- [Print all possible paths from top left to bottom right of a mXn matrix](#)
- [Generate all unique partitions of an integer](#)
- [Russian Peasant Multiplication](#)
- [Closest Pair of Points | O\(nlogn\) Implementation](#)



15



Tweet

1



2

Writing code in comment? Please use [ideone.com](https://ideone.com) and share the link here.

## 13 Comments GeeksforGeeks

Sort by Newest ▼



Join the discussion...



**kinshuk chandra** · 8 days ago

Here is my basic algo:

1. Break the number into 2 halves "2133" = "21", "33"
  2. Now if first half is greater than 2nd half AND first half is same size as 2nd half AND reverse(first) is greater than 2nd half, then nextPalindrome = first + reverse(first), otherwise go to step 3.  
Example 783322 = 783 + 387 = 783387. However 2133 wont hold. Nor will 1001, as first = 10, and last = 01 = 1
  3. Increment the first half by 1,  
first = first+1  
nextPalindrome = first + reverse(first)  
Example 2133, first = 21, first+1 = 22, Now concat first and reverse first = 222
- Here is my post -  
<http://k2code.blogspot.in/2014...>

^ | v ·



**sammyblr** · 23 days ago

Check out this code for finding next smallest palindrome.

<http://wp.me/3bJAF>

^ | v ·



**rahul** · 7 months ago

@GEEKSforGeeks

We can have one check before the code mirror left to right

i.e.

```
if(!leftsmaller)
```

```
mirror left to right;
```

```
else
```

```
increment middle case
```

This would save lot of time for the cases in which we are increment middle digit unnecessary steps.

thnx

^ | v .

**Preeti** · 7 months ago

Next Palindrome for 9 4 1 8 7 9 7 8 3 2 2 should be 9 4 1 8 7 9 7 8 1 4 9. Right

^ | v .

**Rahul** → Preeti · 4 months ago

No its 9 4 1 8 8 0 8 8 1 4 9

^ | v .

**Thunderbird** · 8 months ago

<http://www.codechef.com/viewso...>

^ | v .

**Akash Verma** · 9 months ago

wonder if the so-long code effects the time complexity!

^ | v .

**Sri Hari** → Akash Verma · 8 months ago





Yeah totally!!

Not the execution time complexity but the implementation time comple:

5 ^ | v .



**Abhinav** · 11 months ago

Hi GeeksforGeeks,

In the below code we can optimize a little bit by putting a check in the while loc

```
while (i >= 0)
{
    num[i] += carry;
    carry = num[i] / 10;
    num[i] %= 10;
    num[j++] = num[i--]; // copy mirror to right
}
```

Instead of while (i >= 0), we can put

while(i >= 0 && carry == 1)

~~Abhinav

1 ^ | v .



**Prashant** · 2 years ago

i got it :)


thanx

^ | v .



**Rishabh** · 2 years ago

#include



```
using namespace std;
int main()
{
    int a[20],n,i,j,k,b[20],mid;
    cout<<n;
    cout<<"Enter the array\n";
    for(k=0;k<n;k++)
    {
        cin>>a[k];
    }
    mid=n/2;
    if(n%2==0)
    {
        i=mid-1;
        j=mid;
    }
    else
    {
        i=mid-1;
```

[see more](#)

^ | v .



**Kailash** · 2 years ago

Steps:

1. Add 1 to the given no.(as we want next palindrome, greater then current number)
2. keep i and j pointers on leftmost and rightmost digit respectively.
3. now move both i and j inside i.e. towards middle digit.
4. make digit at [j] equal to digit at [i] by adding something. if there is any carry towards MSB digit.
5. now move j to next digit from right and follow step 4.
6. repeat step 4,5 till i,j are on the middle digit/s.

In this algorithm there is no need to handle any special cases.

^ | v .



**Prashant** ↗ Kailash · 2 years ago

@kailash: plzz explain how your mentioned algorithm will work for a pa

^ | v .



Subscribe



Add Disqus to your site

@geeksforgeeks, **Some rights reserved**

**Contact Us!**

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team