This is a [reverse challenge](keyword://reverse-challenge), enjoy!

**Input/Output**

* **[execution time limit] 3 seconds (cs)**
* **[input] integer n**

*Guaranteed constraints:*  
0 ≤ n ≤ 2000.

* **[output] integer**

**[C#] Syntax Tips**

// Prints help message to the console

// Returns a string

**string** **helloWorld**(**string** name) {

Console.Write("This prints to the console when you Run Tests");

**return** "Hello, " + name;

}

<https://codefights.com/challenge/r8bT4u7QQWpSB84v5/solutions>

<https://www.geeksforgeeks.org/generate-palindromic-numbers-less-n/>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp36

{

class Program

{

// A utility for creating palindrome

static int createPalindrome(int input, int b, bool isOdd)

{

int n = input;

int palin = input;

// checks if number of digits is odd or even

// if odd then neglect the last digit of input in

// finding reverse as in case of odd number of

// digits middle element occur once

if (isOdd)

n /= b;

// Creates palindrome by just appending reverse

// of number to itself

while (n > 0)

{

palin = palin \* b + (n % b);

n /= b;

}

return palin;

}

// Fruition to print decimal palindromic number

static int nPal(int n)

{

List<int> lista = new List<int>();

int number=0;

// Run two times for odd and even length palindromes

for (int j = 0; j < 2; j++)

{

// Creates palindrome numbers with first half as i.

// Value of j decided whether we need an odd length

// of even length palindrome.

int cont = 0;

int i = 1;

while ( cont < n)

{

number = createPalindrome(i, 10, j % 2 == 0);

//cout << number << " ";

//Console.WriteLine(number);

lista.Add(number);

i++;

cont++;

}

}

lista.Sort();

if (n == 0) return 0;

return lista[n-1];

}

static void Main(string[] args)

{

Console.WriteLine( nPal(1)); //37 272

Console.ReadLine();

}

}

}

int nPal(int n)

{

Dictionary<int, int> d = new Dictionary<int, int>();

int i = 0;

while (d.Count <= n)

{

if (p(i.ToString()))

d[d.Count] = i;

i++;

}

return d[n];

}

bool p(string n)

{

if (n == new string(n.Reverse().ToArray()))

return true;

return false;

}