Problem 41 - Pandigital Prime

Gautam Manohar

24 June 2018

This document originally appeared as a blog post on my website. Find it at `gautammanohar.com/euler/41`.

1 Problem Statement

We shall say that an n-digit number is pandigital if it makes use of all the digits 1 to n exactly once. For example, 2143 is a 4-digit pandigital and is also prime. What is the largest pandigital prime less than or equal to N? If there is none, return -1.

2 My Algorithm

The sum of digits of an n-digit number is the sum of the first n positive integers, or $\frac{n(n+1)}{2}$. Consider the value of this expression modulo 3. The multiplicative modular inverse of 2 modulo 3 is 2, because $2 \cdot 2 = 4 \equiv 1 \pmod 3$. And so

$$\frac{n(n+1)}{2} \equiv 2n(n+1) \pmod 3. \tag{1}$$

If and only if 3 divides one of n and $n+1$, then this expression is congruent to 0 modulo 3; thus it is not prime. This means that $n \equiv 1 \pmod 3$; and so if an n-digit pandigital prime exists, we must have $n \in \{4, 7\}$.

This observation reduces the search space of this problem by a lot. Our strategy is to generate pandigital numbers and check if they are prime. The maximum possible value of a seven-digit number is $10^8 - 1$; to test the primality of all such numbers, we need a list of primes up to $\lfloor \sqrt{10^7 - 1} \rfloor = 3162$.

To generate the pandigital numbers of length 4 and 7, we will take the first 4 or 7 characters of the string `123456789` and use `itertools.permutations`. Then, we will check whether each pandigital number p is prime; if so, it is either part of our list of primes or it is not divisible by any prime less than or equal to $\sqrt{p}$.

Finally, we can answer each query with a binary search on our list of pandigital primes.

As shown above, we generate the primes up to 3162 with a Sieve of Eratosthenes. We then generate the $4! + 7! = 5064$ pandigital candidates. Then, for each of them, we test at most about $\frac{3162}{\log 3162}$ primes. Finally, we perform a binary search on our list of pandigital primes with at most $\log 3162$ operations.

All in all, the time complexity for this solution does not depend on the size of N, and so it is $O(1)$; however, the constant term is quite large.