forthright48

learning never ends

Highly Composite Numbers

forthright48 on July 13, 2015

Definition

> **A Highly Composite Number (HCN) is a positive integer which has more divisors than any smaller positive integer (Wiki), i.e, if $NOD(N) > NOD(i)$, where $0 < i < N$, then N is HCN.**

For example, 6 is HCN cause $NOD(6) = 4$ which is bigger than $NOD1, 2, 3, 4, 5 = 1, 2, 2, 3, 2$.

Here are the first few HCN: $1, 2, 4, 6, 12, 24, 36, 48, 60, 120, 180, 240$ – A002182

Properties of Highly Composite Numbers

There are two properties of HCN and both of them are related to prime factorization.

Suppose we have a HCN with prime factorization $HCN = p_1^{a_1} \times p_2^{a_2} \ldots \times p_k^{a_k}$, then:

1. **First K Primes**: The prime factorization of HCN will contain the first K consecutive primes. If it doesn't, then we can replace the k^{th} prime in factorization with a smaller prime and still have the same NOD. For example, $2^2 \times 5 = 20$ cannot be an HCN, since we can replace

2
Shares

2. **Non-Increasing Power of Primes**: Power of prime factors of HCN, i.e, $a_1, a_2 \ldots a_k$ will form a non-increasing sequence. Why is that? If power $a_j > a_i$, where $i < j$, then we can simply swap them to get a smaller number with the same NOD. For example, $2 \times 3^2 = 18$ cannot be HCN cause we can swap power of prime 2 and 3 to get $2^2 \times 3 = 12$ which is smaller with same NOD.

Therefore we conclude that Highly Composite Numbers are product of Primorials (Primorials are same as Factorials, but instead of natural number we multiply primes. $p_3 = 2 \times 3 \times 5$ and $p_5 = 2 \times 3 \times 5 \times 7 \times 11$)

For example, 180 is HCN and it can be decomposed into $180 = p_3 \times p_2 = (2 \times 3 \times 5) \times (2 \times 3)$.

Generating Highly Composite Numbers

Problem: Given an integer N, find the largest HCN which is smaller than or equal to N.

This problem can be solved using backtrack. We just need to ensure that we satisfy the two properties of HCN.

Let's try to solve it for when $N = 10^9$. We know that $p_{10} = 6469693230$, so we will only have primes $2, 3, 5, 7, 11, 13, 17, 19, 23$ in HCN factorization.

Now, we will assign power to each of the primes in non-increasing order to generate numbers and corresponding NOD. Out of all the numbers generated, we will keep the one with the highest NOD and in case of a tie, the one with smallest value.

```
// prime[] is a list of prime.
int prime[] = {2, 3, 5, 7, 11, 13, 17, 19, 23 };

int resNum, resDiv, n;
void recur ( int pos, int limit, long long num, int div ) {
    if ( div > resDiv ) { // Get the number with highest NOD
        resNum = num;
        resDiv = div;
    }
    else if ( div == resDiv && num < resNum ) { //In case of tie, take sma
        resNum = num;
    }

    if ( pos == 9 ) return; //End of prime list

    long long p = prime[pos];

    for ( int i = 1; i <= limit; i++ ) {
```

```
19         if ( num * p > n ) break;
20         recur ( pos + 1, i, num * p, div * ( i + 1 ) );
21         p *= prime[pos];
22     }
23 }
```

Line 2 contains a prime list. It contains first 9 primes so it will work correctly for $N \leq 10^9$. Line 4 contains global variables to store the result. `resNum` to hold required value and `resDiv` to hold its NOD. Line 19 checks if multiplying $prime[pos]^i$ with `res` is becoming bigger than N or not.

In line 20, we call $recur()$ with $pos + 1$ as we want to work with the next prime in the list, i as the new limit since we don't want the next prime to have a power greater than the current one. The next two parameters are adjusted accordingly to maintain value and NOD.

In order to solve the above problem for $N = 10^9$, we have to initiate and call $recur()$ from $main()$ in following manner.

```
1 int main () {
2     n = 1000000000;
3     resNum = 0;
4     resDiv = 0;
5     recur ( 0, 30, 1, 1 );
6     printf ( "%d %d\n", resNum, resDiv );
7 }
```

In line 5, we call `recur(0,40,1,1)`. `pos` is assigned 0 since we want to start with the first prime in prime list. `limit` parameter is set to 30 since $2^{30} > N$.

Reference

1. Wiki – Highly Composite Numbers: https://en.wikipedia.org/wiki/Highly_composite_number

2. Primorials: https://en.wikipedia.org/wiki/Primorial

Share Tweet Share Share Email

Print Share

Category: CPPS, Number Theory

Previous:
Number of Divisors of an Integer

Next:
Upper Bound for Number Of Divisors

Guest

Your comment...

Sort by oldest

No comments to display

Archives

September 2018 (2)

February 2018 (1)

January 2018 (1)

November 2017 (2)

September 2015 (7)

August 2015 (13)

July 2015 (15)

Categories

CPPS (40)

- Combinatorics (3)
- Number Theory (33)

Meta (1)

Recent Comments

forthright48 on Number of Divisors of an Integer

Sohana Akter on Number of Divisors of an Integer

forthright48 on Number of Divisors of an Integer

forthright48 on Prufer Code: Linear Representation of a Labeled Tree

Mohammad Samiul Islam on SPOJ LCMSUM – LCM Sum

□

Privacy Policy