Summation of all proper divisors

asked **8 months ago**
viewed **1317 times**
active **8 months ago**

0

I have solved a question that says:
Given a natural number n (1 <= n <= 500000), please output the summation of all its proper divisors.

Definition: A proper divisor of a natural number is the divisor that is strictly less than the number.

e.g. number 20 has 5 proper divisors: 1, 2, 4, 5, 10, and the divisor summation is: 1 + 2 + 4 + 5 + 10 = 22.

Input

An integer stating the number of test cases (equal to about 200000), and that many lines follow, each containing one integer between 1 and 500000 inclusive.

Output

One integer each line: the divisor summation of the integer given respectively.

Example

Sample Input:

```
3
2
10
20
```

Sample Output:

```
1
8
22
```

My code is as follows:

```
/* @BEGIN_OF_SOURCE_CODE */

#include <stdio.h>
#include <stdlib.h>

    int main(int argc, const char * argv[])
    {
        int sum = 0,
        cases = 0,
        i, j, buff;

        scanf("%d", &cases); //Number of tests


        int *n;
        n = (int*) malloc(cases * sizeof(int)); //Defining array for numbers to be

        for (i = 0; i < cases; i++) {
            scanf("%d", &n[i]);
        }
        for (i = 0; i < cases; i++ ) {
```

```
            buff = n[i] / 2;
            if (n[i] == 1) {
                sum = -1;
            }


            if (!(n[i] & 1)) {
                for (j = 2; j < buff; j++) {
                    if (n[i] % j == 0) {
                        sum += n[i] / j + j;
                        buff /= j;
                    }
                }
```

but it is not fast enough. Any suggestions?

c performance

share | improve this question

edited Sep 18 '13 at 22:04

jxh
30.9k ●6 ●22 ●56

asked Sep 16 '13 at 19:59

Orkhan Hasanli
14 ●5

I think this code is from *Chef Code*. – haccks Sep 16 '13 at 20:04

why are you mallocing? – Bartlomiej Lewandowski Sep 16 '13 at 20:08

1 And what are you expecting to do for double-factors (e.g. 4 has a factor of 2 twice)? – Zac Howland Sep 16 '13 at 20:09

2 Also, you're not freeing the memory you allocated with `malloc`. – Freddie Sep 16 '13 at 20:09

3 This question appears to be off-topic because it belongs to codereview SE site. – BartoszKP Sep 16 '13 at 20:11

show **6** more comments

5 Answers

active oldest votes

You don't have to allocate space. Just do line by line. For each line, there is an O(n ^ 1/2) algorithm.

4 Algorithm to find all the exact divisors of a given integer

127 A fast method to round a double to a 32-bit int explained

0 Summation of nth prime

0 Getting result 1 less or 1 more, divisor counting Program - C

28 What is the fastest way to return the positions of all set bits in a 64-bit integer?

0 Finding Divisors using loops

Hot Network Questions

Case-insensitive substring search in a shell script

Builder Pattern: When to fail?

Hypernym for reading a book and listening to an audiobook

Finding substring with regex for contents of <tr> tags

Is it ethical to submit a paper with the name of a co-author who cannot be contacted?

How can I break a macro / how to realize a switch or array-

2

```
#include <iostream>
using std::cout; using std::endl; using std::cin;

int main() {
   int count, number;
   cin >> count;
   for (int i = 0; i < count; ++i) {
      cin >> number;
      int sum = 1;
      for ( int j = 2; j * j <= number; ++j ) {
         if ( number % j == 0 ) {
            sum += j;
            sum += number / j;
         }
         if ( j * j == number ) sum -= j; // recalculate twice
      }
      cout << sum << endl;
   }
}
```

This is the runtime for 200,000 test case

```
real    0m55.420s
user    0m0.016s
sys     0m16.124s
```

share | improve this answer

edited Sep 16 '13 at 20:35

answered Sep 16 '13 at 20:18

BigTailWolf
389 ●1 ●5

It is not optimized well. Cause for odd numbers you are checking whether it is divisable by even number or not. My code works fine. I dont have any error. Just need to make it faster. – Orkhan Hasanli Sep 16 '13 at 20:32

Just for your sake, I submitted your code. It does not work and it is slower than my code. – Orkhan Hasanli Sep 16 '13 at 20:36

btw, How do you check the runtime? – Orkhan Hasanli Sep 16 '13 at 20:37

using command "time" – BigTailWolf Sep 16 '13 at 20:46

structure

Breaking the equation inside a fraction

Why does the percent sign in a URL cause an HTTP 400 Bad Request error?

Is there a logic gate which directly outputs (A'B)' (complement of, A complement product B)?

Gödels (first) incompleteness Theorem and the Halting Problem - How limiting is it?

Get distinct combinations of numbers

Why is the first hotcake always the worst?

"Give him a box that everyone knows what it contains."

Why "A=10 echo $A" does not print 10?

How do I write a solid plus sign?

Declare Bankruptcy or Just Wait Seven Years?

Three consecutive integers which are power of prime but not prime

What word would be used for someone who shoots video

using command "time" BigTailWolf Sep 16 '13 at 20:45

What do you mean by "not work"? Doesn't complied? Run time error? – BigTailWolf Sep 16 '13 at 20:51

show **4** more comments

2

I have updated the code below to terminate sooner. Running it for all integers from 1 to 500,000 takes under half a second on a MacBookPro6,1 (2.66 GHz Intel Core i7), compiled with Apple GCC 4.2.1 with -O3.

It uses the formula for $\sigma_x(n)$ in the Properties section of the Wikipedia page for the divisor function. It could be made faster with a list of precalculated primes. (126 are needed to support inputs up to 500,000, and this reduces the time to less than a quarter of a second.) There are also some divisions that can be eliminated, at the expense of cluttering the code slightly.

```
//  Return the least power of a that does not divide x.
static unsigned int LeastPower(unsigned int a, unsigned int x)
{
    unsigned int b = a;
    while (x % b == 0)
        b *= a;
    return b;
}


//  Return the sum of the proper divisors of x.
static unsigned int SumDivisors(unsigned int x)
{
    unsigned int t = x;
    unsigned int result = 1;

    //  Handle two specially.
    {
```

What can I use instead of commercially prepared wet cat food?

Climbing Mount Fuji

Is there an expression for someone who often takes things too literally?

If a character does something evil, but they can justify it as good are they evil?

An idiom for a stupid action which is beneficial in the end, or for the person who performed said action

Is it possible to change a hypothesis to match observed data (aka fishing expedition) and avoid an increase in Type I errors?

```
        unsigned int p = LeastPower(2, t);
        result *= p-1;
        t /= p/2;
    }

    //  Handle odd factors.
    for (unsigned int i = 3; i*i <= t; i += 2)
    {
        unsigned int p = LeastPower(i, t);
        result *= (p-1) / (i-1);
        t /= p/i;
    }

    //  At this point, t must be one or prime.
    if (1 < t)
        result *= 1+t;
```

share | improve this answer

edited Sep 18 '13 at 22:48

answered Sep 16 '13 at 20:44

Eric Postpischil

35.9k 3 36 73

I see, you take a short cut to computing the sum by computing (pow(p, k+1)-1)/(p - 1). – jxh Sep 17 '13 at 2:33

I think if you modify the `LeastPower()` function to divide out the prime powers as you go, you will get a speed up because the `%` operator result will get reused by the subsequent divide, so the divide is free. – jxh Sep 18 '13 at 22:50

@jxh: Thanks, that is useful. I had already reduced the time to .061 seconds with some other modifications, including memoization. Your change brings it to .027 seconds (for finding the sums of divisors of all numbers from 1 to 500,000). – Eric Postpischil Sep 18 '13 at 23:39

Wow, memoization is a massive win. – jxh Sep 19 '13 at 2:16

add comment

0

I would start by NOT storing the numbers in an array at all. You don't need to - just read the value, process it, and output the result. The compiler may well not realize that `n[i]` is the same value throughout the loop, and that nothing else modifies it.

The logic doesn't seem very clear to me. And `if (n[i] == 1) { sum = 1} else ...` would make more sense than setting `sum = -1`.

You could perhaps also, keep a list of "common factors" (http://en.wikipedia.org/wiki/Memoization), so that you don't have to recalculate the same thing many times over. [If you know that somehing has the factor 24, then it also has 2, 3, 4, 6 and 8, for example.

share | improve this answer

answered Sep 16 '13 at 20:16

Mats Petersson
63k ●6 ●29 ●79

First parameter is "cases", let say it is 2. Then I have to input 2 numbers each in new line. Let say first one is 15. When I submit the input, it returns me a value before second input. So, I have to save them first, then do my dirty stuff. Moreover, I'm trying to return the value at the end, so assignment sum = -1 is just the way i use. I do not need stylish things, guys, all I need a suggestion for faster code. – Orkhan Hasanli Sep 16 '13 at 20:21

My `if (n[i] == 1)` makes it faster by not doing the whole divide stuff - albeit it obviously doesn't matter. Why can't you output the value as you calculate it? Surely the code runs as `myprog < input.txt > result.txt`, so outputting the data as you calculate would work just fine... – Mats Petersson Sep 16 '13 at 20:23

Also, not using an array will (most likely) make it faster. – Mats Petersson Sep 16 '13 at 20:33

Believe in me, I DID ALL THINGS YOU SAID! None works. The error I get from submission is Time limit exceeded. – Orkhan Hasanli Sep 16 '13 at 20:33

Then you are using the wrong method to solve the problem - I suspect jxh's solution is the one that is the right one. – Mats Petersson Sep 16 '13 at 20:41

add comment

I replied to a similar question on stackoverflow

0

There is a faster performing algorithm which is based on a formula for the sum of divisor using the decomposition in prime factors.

First you construct a primetable such that the last prime squared is smaller than the upper bound for your number. Then you apply the formula to each entry. If a number is written as

```
n = a1^p1 * a1^p2 *... *an^pn
```

the complexity of finding the sum for a given number `n` will be

```
p1+p2+...+pn = roughtly log(n)
```

which is better than the complexity `O(sqrt(n))` of the first optimization which stop the loop early

share | improve this answer

edited Sep 16 '13 at 20:44

answered Sep 16 '13 at 20:38

UmNyobe

8,513 ●2 ●14 ●43

Will try the factorization. Hope it will work. – Orkhan Hasanli Sep 16 '13 at 20:44

hint: you can compute the primes with a program, print them, store them in a statis array (primetable={2,3,5,7...) and use them in the program to submit to the online judge. Which means zero time spent by the online judge in the prime factorization... – UmNyobe Sep 16 '13 at 20:47

Given a natural number n (1 <= n <= 500000), please output the summation of all its proper divisors. An integer stating the number of test cases (equal to about 200000) Are you sure? There is time, memory limitations. – Orkhan Hasanli Sep 16 '13 at 20:59

My bad I meant prime generation. – UmNyobe Sep 16 '13 at 21:31

add comment

0

Let's suppose you have a way to compute primes relatively quickly. This could be a one time upfront activity, bounded by the square root of the largest input value. In this case, you already know the bound of the largest input value (500000), so you can simply hard code a table of primes into the program.

```
static unsigned P[] = {
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151,
157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233,
239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317,
331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419,
421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503,
509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607,
613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701
```

```
};

static int P_COUNT = sizeof(P)/sizeof(*P);
```

Now, from the primes, for each input value, you can:

- Compute the prime factorization
- Compute the product of the sums of the powers of each prime factor.

This will result in the sum of the divisors. Subtract the input value from the sum to obtain the sum of proper divisors. These two steps can be combined into a single loop.

This algorithm works because multiplying polynomials naturally results in sums of all combinations of the polynomial terms multiplied together. In the case where each polynomial term consists of powers of primes that divide the input, the combinations of the terms multiplied together make up the divisors. The algorithm is fast, and should be able to process 500000 numbers in the interval [1, 500000] in less than a second on a Core i3 or better processor.

The following function implements the method described above.

```
unsigned compute (unsigned n) {
    unsigned sum = 1;
    unsigned x = n;
    for (int i = 0; i < P_COUNT; ++i) {
        if (P[i] > x / P[i]) break;    /* remaining primes won't divide x */
        if (x % P[i] == 0) {           /* P[i] is a divisor of n */
            unsigned sub = P[i] + 1;   /* add in power of P[i] */
            x /= P[i];                 /* reduce x by P[i] */
            while (x % P[i] == 0) {    /* while P[i] still divides x */
                x /= P[i];             /* reduce x */
                sub = sub * P[i] + 1;  /* add by another power of P[i] */
            }
            sum *= sub;                /* product of sums */
        }
    }
    if (x > 1) sum *= x + 1;           /* if x > 1, then x is prime */
    return sum - n;
}
```

share | improve this answer

answered Sep 19 '13 at 1:58

jxh
30.9k 6 22 56

A version of the above program running on all numbers from 1 to 500000 (in random order) on IDEONE. – jxh Sep 19 '13 at 4:39

add comment

Your Answer

Sign up or log in

Sign up using Google

Sign up using Facebook

Sign up using Stack Exchange

Post as a guest

Name

Email

Post Your Answer

By posting your answer, you agree to the privacy policy and terms of service.

Not the answer you're looking for? Browse other questions tagged c performance or ask your own question.

question feed