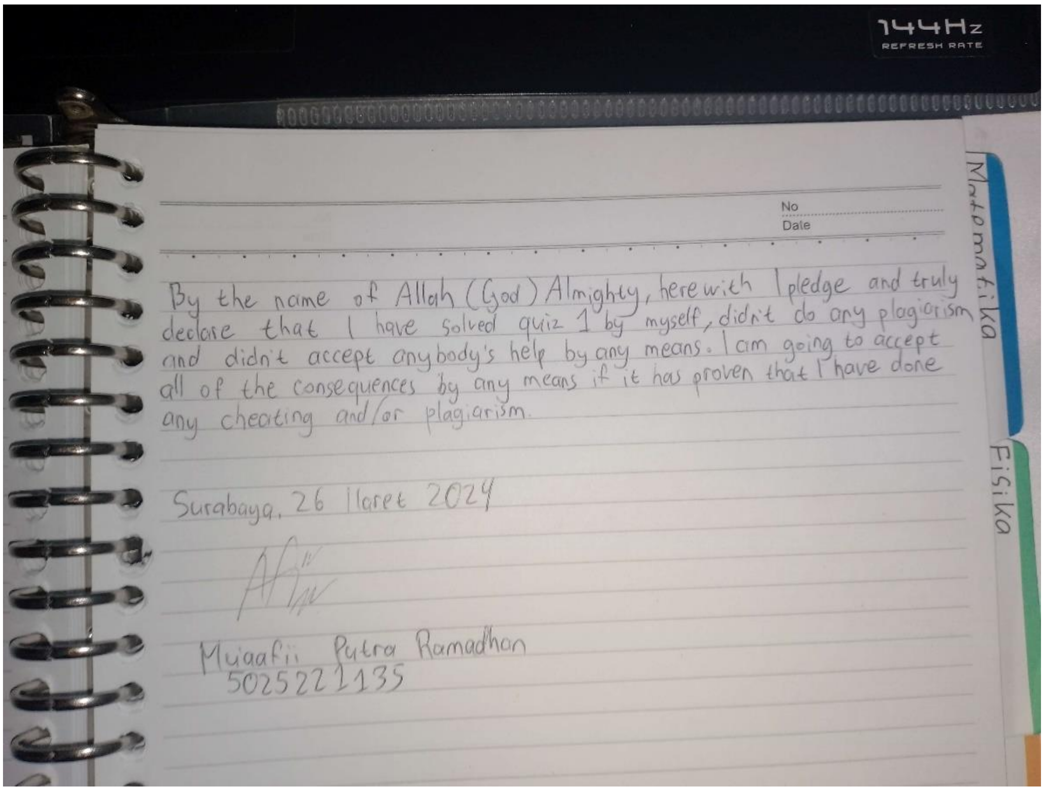
Name : Mu'aafii Putra Ramadhan
NRP : 5025221135
Class : Design & Analysis of Algorithm B

Quiz 1

SPOJ acc : mopipr_06
SPOJ pass : Hybrida12a
Signed Pledge :



1. SPOJ Classical 9034 – TOHU

# TOHU - Help Tohu

*no tags*

Tohu wants to enter Moscow State University, and he must pass the math exam. He does not know math, and asks you to help him. The problem is to find the sum $S_n = a_1 + a_2 + ... + a_n$ of the sequence $\{a_n\}$ on condition $\forall k \in \mathbb{N}: a_1 + 2a_2 + ... + ka_k = \frac{k+1}{k+2}$

## Input

First line contains single integer T <= 20000 - the number of test cases.

Each of the next T lines contains single integer 1 <= n <= 10^9.

## Output

For each n output the value $S_n$ with 11 digits after decimal point.

## Example

```
Input:
2
2
5

Output:
0.70833333333
0.73809523810
```

Problem Summary ➔ Help Tohu find the Sum $S_n = a_1 + a_2 + \cdots + a_n$ of a sequence $\{a_n\}$ with given terms $\forall k \in N: a_1 + 2a_2 + \cdots + ka_k = \frac{k+1}{k+2}$. Tohu need to fins the Sum Sn of the sequence$\{a_n\}$ for the given n

Solution ➔

- Observations :

The loop iterates through all numbers (i) from 2 to a certain maximum (max). This suggests a pattern in the sum of the values calculated within the loop.

Each number (i) contributes to the sum of divisors of its multiples (j * i) based on the value of i. In other words, how many times i appears as a divisor for other numbers affects its total contribution.

By understanding this pattern, we might be able to directly calculate the final sum without explicitly finding the number of divisors for every number.

- Concepts used :

Arithmetic Series: We can utilize the properties of arithmetic series to identify the pattern in the sum of divisors. By understanding how the number of terms, first term, and last term affect the sum, we might be able to develop a formula.

Recursive Relationship: The loop condition provides a clue about how each number contributes to the total based on its position in the sequence. This recursive relationship can be helpful in formulating a direct calculation method.

Source Code :

```c
#include <stdio.h>

int main() {
    int T;
    double sum;
    scanf("%d", &T);

    for (int i = 0; i < T; i++) {
        int n;
        scanf("%d", &n);

        sum = (3.0 / 4.0) + (1.0 / (2 * (n + 2))) - (1.0 / (2 * (n + 1)));

        printf("%.11f\n", sum);
    }

    return 0;
}
```

Proof of submission and accceptance

| | | | | | | |
|---|---|---|---|---|---|---|
| 32839493 | 2024-03-26 16:25:14 | Mu'aafii Putra Ramadhan_5025221135 | **accepted** edit ideone it | 0.01 | 5.2M | C |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 32839493 | ☐ | 2024-03-26 16:25:14 | Help Tohu | **accepted** edit ideone it | 0.01 | 5.2M | C |

2. SPOJ Classical 14138 – Amazing Factor Sequence

# AFS - Amazing Factor Sequence

*no tags*

Bhelu is the classmate of Bablu who made the Amazing Prime Sequence.

He felt jealous of his classmate and decides to make his own sequence. Since he was not very imaginative, he came up with almost the same definition just making a difference in f(n):

- a[0] = a[1] = 0.
- For n > 1, a[n] = a[n - 1] + f(n), where f(n) is the sum of positive integers in the following set S.
- S = {x | x < n and n % x = 0}.

Now, Bablu asked him to make a code to find f(n) as he already had the code of his sequence. So, Bhelu asks for your help since he doesn't know programming. Your task is very simple, just find a[n] for a given value of n (< 10^6).

## Input

Your code will be checked for multiple test cases.

First Line of Input contains T (<= 100), the number of test cases.

Next T lines contain a single positive integer N. (1 < N < 10^6).

## Output

Single line containing a[n] i.e. n-th number of the sequence for each test case.

## Example

```
Input:
3
3
4
5

Output:
2
5
6
```

**Explanation**

```
f(2) = 1 {1}
f(3) = 1 {1}
f(4) = 3 {1, 2}
f(5) = 1 {1}
```

Problem Summary ➔ Bhelu, inspired by his classmate Bablu's "Amazing Prime Sequence," creates his own sequence with a twist. Sequence Definition:

- The first two elements (a[0] and a[1]) are both 0.
- For any element a[n] where n is greater than 1, it's calculated by:
- Adding the previous element (a[n-1]) to
- A function f(n), which represents the sum of all positive divisors of n.

Task:

- Given a non-negative integer n less than 1,000,000 (10^6), you need to find the nth element (a[n]) in Bhelu's sequence.

Solution ➔Algorithm Strategy:

- Precomputation for Efficiency:

To avoid repetitive calculations for multiple test cases, the algorithm precomputes the entire sequence a[n] for all possible n values up to the given limit (1,000,000).

- f(n) Calculation:

The function f(n) represents the sum of all positive divisors of n excluding n itself. This is because the set S used in the definition of f(n) only considers divisors less than n.

The concept of "sum of proper divisors" is used to calculate f(n) for each n.

- Sequence Construction:

During precomputation, the recurrence relation a[n] = a[n-1] + f(n) is employed to iteratively build the sequence. Each element a[n] is calculated by adding the previous element a[n-1] to the corresponding f(n) value.

Concepts Used :

- Sum of Proper Divisors: This mathematical concept finds the sum of all divisors of a number excluding the number itself. It's crucial for calculating f(n), which represents the sum of divisors contributing to a[n].
- Precomputation: Similar to the Sieve of Eratosthenes algorithm, precomputing the entire sequence beforehand avoids redundant calculations for each test case, improving efficiency.
- Recurrence Relation: The provided formula a[n] = a[n-1] + f(n) defines how each element in the sequence depends on the previous element and f(n), allowing for iterative calculation of the sequence during precomputation.

Source Code :

```
#include <stdio.h>

#define max 1000000
long long int list[max];
long long int store[max];

void func() {
    long long int i, num, j;
    for (i = 2; i < max; i++)
        list[i] = 1;

    for (i = 2; i < max; i++) {
        j = 2;
        while ((num = j * i) < max) {
            list[num] += i;
            j++;
        }
    }
```

```c
    store[0] = store[1] = 0;
    for (i = 2; i < max; i++)
        store[i] = store[i - 1] + list[i];
}

int main() {
    long long int T, i, n, j;
    func();
    scanf("%lld", &T);

    for (i = 0; i < T; i++) {
        scanf("%lld", &n);
        printf("%lld\n\n", store[n]);
    }

    return 0;
}
```

Proof of submission and acceptance

| | | | | | | |
|---|---|---|---|---|---|---|
| 32839260 | 2024-03-26 15:46:59 | Mu'aafii Putra Ramadhan_5025221135 | **accepted** edit ideone it | 0.06 | 17M | C |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 32839260 | ☐ | 2024-03-26 15:46:59 | Amazing Factor Sequence | **accepted** edit ideone it | 0.06 | 17M | C |