Euler Problem 1 Writeup

Leo Steinberg

October 9, 2022

1 Problem Statement:

If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23.
Find the sum of all the multiples of 3 or 5 below 1000.

1.1 Coding adapation

Haker Rank provides the following code adapation:

If we list all the natural numbers below 10 that are multiples of 3 or 5 , We get $3, 5, 6, 9$. The summ of theese multiples is 23. Find the sum of all the multiples of 3 or 5 below N.

Input Format

First line contains T that denotes the number of test cases. This is followed by T lines, each containing an integer, N.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq N \leq 10^9$

Output Format

For each test case, print an integer that denotes the sum of all the multiples of 3 or 5 below N.

1.2 Solution

1.2.1 bash

My first atempt was simply to bash this. There are quicker ways and it could all be done in main, but I found this to be the most "human readable" way of doing it.

```c
#include <math.h>
#include <stdio.h>

long SumUnderN(long);

int main(){
	int i; //counter
	int T; // # of testcases
	long N; // place to store each instance of N.

	scanf("%d",&T);

	for (i = 0;i < T;i++)
	{
		scanf("%ld",&N);
		printf("%ld\n",SumUnderN(N));
	}
	return 0;
}
long SumUnderN(long N)
{
	long i; //counter
	long sum = 0; // sum
	for(i = 1; i< N;i++)
	{
		sum += i * ((i % 3 == 0) || (i % 5 == 0));
	}
	return sum;
}
```

It should be noted the long data type needs to be used as N can be up to 10^9.

1.2.2 Optimization / Math

The problem with this method is it's inefficient. For most problems like this with modern compuational power this is not a problem, however for the sake of good practice we should do better. As such Hackerank fails two of the automated testcases as it is also testing for this.
The multiples of 3 and 5 are by definition at regular intervals. We could edit our for loops to ittrate twice through the given number N, in steps of 3 then 5, but this still isn't optimal. Some basic Number Theory and inclusion/exclusion tell us this can be calulated for a given number with division. If we add up all the mutiples of 3, then the mutplies of 5 then subtract the multiples of 15 we get our answer Thus we get:

$$\text{answer} = \left(\sum_{k=1}^{\lfloor \frac{N}{3} \rfloor} (3 \cdot k)\right) + \left(\sum_{k=1}^{\lfloor \frac{N}{5} \rfloor} (5 \cdot k)\right) - \left(\sum_{k=1}^{\lfloor \frac{N}{15} \rfloor} (15 \cdot k)\right)$$

$$= 3 \cdot \frac{\lfloor \frac{N}{3} \rfloor \left(\lfloor \frac{N}{3} \rfloor + 1\right)}{2} + 5 \cdot \frac{\lfloor \frac{N}{5} \rfloor \left(\lfloor \frac{N}{5} \rfloor + 1\right)}{2} - 15 \cdot \frac{\lfloor \frac{N}{15} \rfloor \left(\lfloor \frac{N}{15} \rfloor + 1\right)}{2}$$

We subtract the sum of multiples up to $\lfloor \frac{N}{15} \rfloor$ to remove the double counting of numbers that are both multiples of 3 and 5. We can implement this in code by changing our SumUnderN function using some custom functions from the math.h header file.

```
long SumUnderN(long N)
{
        long M3 = floor(((N-1)/3)); // multiples of 3
        long M5 = floor(((N-1)/5));// multiples of 5
        long M15 = floor(((N-1)/15));// multiples of 15
        long sum = (3*(M3*(M3+1))/2) +
        (5*(M5*(M5+1))/2) -
        (15*(M15*(M15+1))/2);
        return sum;
}
```