TUGAS KELAS

KELAS PAA B
05111840000094
Rafi Nizar Abiyyi

Departemen Teknik Informatika
Fakultas Teknik ELektro dan Informatika Cerdas
Institut Teknologi Surabaya
2020

Soal SPOJ – STUDCHAN

STUDCHAN - Student Chains

no tags

Problem Statement
There are n students, standing in a line, adjacent to each other, holding hands and forming a chain. You have to remove k students from the line, so that you can form a chain of students of any length between 1 and n. Find the smallest k with which this can be done.

Note:

1. The first and last students in the line are not holding their hands. Thus the chain is not circular.

2. Any one student, can link two chains of arbitrary length.

Input
The first line of input contains an integer T, denoting the no of test cases. Next T lines contain an integer n, denoting the number of students in the chain.

Output
For each test case, print a line denoting the required output.

Constraints
1 <= T <= 50
1 <= n <= 10^10

Sample Input
1
6

Sample Output
1

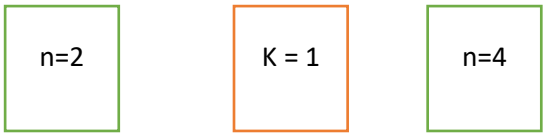
Abstraksi

Terdapat satu baris siswa yang saling berdampingan sebanyak n siswa mementuk rantai siswa. Sebanyak k siswa harus di ambil untuk membentuk rantai siswa yang dapat membentuk rantai sepanjang 1 sampai n siswa. Dengan tambahan ;

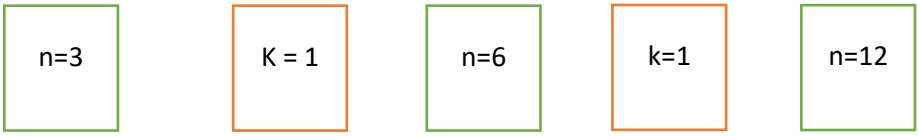
- Siswa pertama dan siswa terakhir tidak membentuk rantai sehingga rantai tidak membentuk lingkaran.
- Setiap satu siswa dapat menghubungkan dua rantai siswa.

Soal ini dikerjakan dengan mencari kontribusi k terhadap n. Setiap nilai k memiliki maksimal n siswa yang dapat di bentuk. Optimal nya, setiap satu siswa yang di ambil harus menggandeng dua rantai.

Untuk K=1, dapat membentuk maksimal rantai sebanyak 7 siswa



Untuk K=2, dapat membentuk maksimal rantai sebanyak 23 siswa



Dan dapat di simpulkan

nilai k	n maksimal
1	7
2	23
3	63

Dari tabel di atas, nilai k membentuk sebuah deret $(k+1) 2^{(K+1)} - 1$

nilai k	n maksimal	$(k+1) 2^{(K+1)} - 1$
1	7	$(2) 2^{(2)} - 1$
2	23	$(3) 2^{(3)} - 1$
3	63	$(4) 2^{(4)} - 1$

Setelah mendapatkan bahwa k memiliki satu nilai maksimum, hasil deret di hitung (pre computation).

Implementasi

```c
#include<stdio.h>

int main() {

	int t;
	long long int a[33], n;

	scanf("%d", &t);

	for(int i=2; i<=32; i++){
		a[i-1] = (long long)i * (long long)(1<<i) - (long long)1;
	}

	while(t--) {
		scanf("%lld", &n);
		for(int i=1; i<=30; i++){
			if(n <= a[i]){
				printf("%d\n", i);
				i = 40;
			}
		}

	}

}
```

Soal SPOJ – MONONUM

MONONUM - Monotonous numbers

#dynamic-programming

Some integers possess interesting quality: each of their digits is not greater than the digit to the right. Let us define such integers as increasing integers. And let's call integers for which each digit is not lesser than the digit to the right decreasing integers. For example 24558 is increasing, 888410 is decreasing and 5 - is both increasing and decreasing. Given **n** calculate the ratio of the decreasing **n**-digit integers to the increasing **n**-digit integers. We consider only positive integers. Leading zeros are not allowed.

Input

The first line of the input contains number **t** – the amount of tests. Then **t** test descriptions follow. Each test consists of the single integer **n**

Constraints

1 <= **t** <= 10000
1 <= **n** <= 10^6

Output

For each test print the needed ratio with six digits in the fractional part.

Abstraksi

Beberapa integer memiliki properti yang unik, setiap digit nya tidak lebih besar dari digit di kanannya atau tidak lebih kecil dari digit di kanannya. Integer yang tidak lebih besar dari digit kanannya kita sebut increasing integers dan integer yang tidak lebih kecil dari digit di kanannya kita sebut decreasing integer.

Contohnya 245558 adalah increasing integer dan 888410 adalah decreasing integer dan 5 adalah increasing dan decreasing.

Untuk *n* adalah rasio dari n-digit decreasing integer ke n-digit increasing integer. Setiap integer bernilai positif, dan kepala 0 tidak diperbolehkan.

Bilangan ascending dan decreasing tersebut dapat dicari menggunakan kombinatorika kombinasi dengan repitisi :

$$C (n + r -1, n)$$

Sebelum mencari rasio *n*, perlu menghitung terlebih dahulu kombinasi untuk bilangan decreasing dan increasing.

Pada bilangan increasing, terdapat 9 angka yang mungkin muncul yaitu 1-9 karena tidak diperbolehkan bilangan dengan kepala 0 dan tidak bernilai negatif.

$$x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 = n,$$

$$x1, x2, x3, x4, x5, x6, x7, x8, x9 >= 0$$

dimana, Xi = Banyak digit angka ke-i, dan

n = Jumlah digit bilangan increasing

Menggunakan rumus kombinasi di dapatkan **C (9+n-1, n)** untuk bilangan icreasing.

Pada bilangan decreasing terdapat 10 angka yang mungkin muncul yaitu (9-0) karena angka nol tidak akan muncul sebagai kepala bilangan pada urutan turun.

$$x9 + x8 + x7 + x6 + x5 + x4 + x3 + x2 + x1 + x0 = n,$$

$$x9, x8, x7, x6, x5, x4, x3, x2, x1, x0 >= 0$$

dimana, Xi = Banyak digit angka ke-i, dan

n = Jumlah digit bilangan increasing

Dalam bilangan decreasing terdapat kombinasi yang tidak mungkin muncul yaitu kombinasi yang hanya berisi x0 (00, 000, 0000, dst). Maka hasil akhir kombinasi bilangan decreasing adalah **C (10+n-1, n) - 1**.

Dari kedua kombinasi diatas, dapat di cari rasio n dengan membandingkan kombinasi decreasing terhadap kombinasi increasing.

$$\frac{C\ (n+9, n) - 1}{C\ (n+8, n)}$$

Rumus tersebut dapat di sederhanakan menjadi :

$$\frac{n+9}{9} - \frac{8!}{\prod_{i=n+1}^{n+8} i}$$

Penyelesaian (pseudocode)

get jumlah digit

inisiasi variabel pi = 1.0

for i dimulai dari n+1 sampai n+8

pi adalah pi dikali dengan i

hasil adalah n+9/9 – 8!/pi

Implementasi :

```c
#include<stdio.h>

int main() {

	int tc, n;
	scanf("%d", &tc);

	double res;
	while(tc--){
		scanf("%d", &n);
		double pi=1.0;
		int i;
		for(i=n+1; i<=n+8; i++){
			pi=pi*(double)i;
		}
		res = (double)(n+9)/9 - 40320.0/pi;
		printf("%.6lf\n", res);
	}

	return 0;
}
```