Problem A. Warm Up

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

When competing XCPC on site, contestants often test their machines in the warmup contest while Zayin has his own method to test – a program which can show the size of stack of the judge machine, how fast the judge runs, and whether __int128 is supported.

For simplicity, consider the following program in C++ implement:

```
#include<bits/stdc++.h>
using namespace std;

__int128 recursion(int n)   {
    return n?recursion(n-1)*rand():1;
}

int main()  {
    srand(time(0));
    int n;
    cin>>n;
    cout<<!recursion(n)<<endl;
    return 0;
}
```

But Zayin found an interesting phenomenon – Granted the exceptions that Runtime Error would be caused by overlarge recursions flooding the stack, the program always ouputs *1* when n is large enough!

Can you help Zayin figure out the result of his program? That is, calculate the probability that the program outputs *1*, or claim that it will cause a Runtime Error.

Input

The input has multiple test cases.

The first line contains an integer T $(1 \le T \le 10^4)$, indicating the number of test cases.

For each test cases, there contains an integer n $(0 \le n \le 10^9)$, indicating the input of program.

Output

For each testcases, if Zayin's program would introduce a Runtime Error, output 'RE'; otherwise, output the probability that the output is *1*.

Note that the answer can be represented as a fractional $\frac{p}{q}$, where p, q are coprime, and you should output $p \times q^{-1}$ modulo 998244353, where q^{-1} is a number satisfying $q \times q^{-1} \equiv 1$ when modulo 998244353.

Examples

standard input	standard output
3 0 1 1000000000	0 232013824 RE

Note

In this problem, we assume $rand()$ is a function that randomly return a number from $[0, 2^{32})$ with equal probability.

And it's guaranteed that the n is sure to or not to introduce a runtime error, in other word, if n_0 is a number which may introduce a 'RE', n is much smaller than n_0 or much bigger than n_0 ($|n - n_0| \geq 10^5$).

You can assume that n_0 is a number which can be obtained by the real online judge(pta).

Problem B. Number Theory

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

Suppose an integer number x's prime factorization is $x = \prod_i p_i^{e_i}$, let's define a number theory function $\lambda(x) = (-1)^{\sum_i e_i}$.

For example, $\lambda(1) = 1,\ \lambda(2) = -1,\ \lambda(4) = 1$.

Your task is to calculate:

$$\sum_{k=1}^{n} \sum_{i|k} \sum_{j|i} \lambda(i)\lambda(j)$$

Input

The only line of input contains an integer n ($1 \le n \le 10^{12}$).

Output

The only line of output contains an integer m, denoting the value modulo 998244353.

Examples

standard input	standard output
`10`	`13`
`100000`	`164038`
`10000000000`	`477285001`

Problem C. Count

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

Zayin and Taotao are poor in mathematics. In order to improve their math skill, they count numbers together today. They write a number every second in the ascending order, such as $0, 1, 2, 3$, etc. However, they can't recognize all the Arabic letters. As a result, they both ignore numbers containing the Arabic letters they don't know. For example, if Zayin only knows 0 and 1, he will write down $0, 1, 10, 11$, etc.

For each Arabic letter, you know whether Zayin and Taotao can recognize it. Both Zayin and Taotao can recognize at least two Arabic letters. Besides, you know the number Zayin will write in x-th second. Can you write a program to calculate the number that Taotao will write at this time?

Input

The first line of input contains an integer T ($1 \leq T \leq 10^4$), denoting the number of test cases.

For each test case, you will get a boolean array A which contains exactly 10 booleans in the first line. If $A_i = 1$ (the index starts at 0), Zayin can recognize Arabic letter i. Otherwise, Zayin can't recognize Arabic letter i.

Similarly, you will get a boolean array B which contains exactly 10 booleans in the second line. If $B_i = 1$ (the index starts at 0), Taotao can recognize Arabic letter i. Otherwise, Taotao can't recognize Arabic letter i.

In the third line, you will get the number Zayin will write in x-th ($x < 2^{64}$) second.

Output

For each test case, you should write down the number that Taotao will write in x-th second in a single line.

Examples

standard input	standard output

`1`
`1 0 1 0 0 0 0 0 0 0`
`1 1 0 0 0 0 0 0 0 0`
`20`
	`10`

Problem D. Stack Sort II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

There are 3 stacks numbered from 1 to 3. Initially, there are n ($n \le 10000$) integers in the first stack, while the second stack and the third stack are empty. Each time you can choose one of the following four operations:

1. Pop an integer from the top of the first stack, and then push it on the top of the second stack.
2. Pop an integer from the top of the first stack, and then push it on the top of the third stack.
3. Swap the first stack and the second stack.
4. Swap the first stack and the third stack.

Because the operations 3 and 4 cost too much energy, the sum of times that you choose 3 and 4 can not exceed 20. That is to say, if you use 3 for T_3 times, and use 4 for T_4 times, you must guarantee that $T_3 + T_4 \le 20$.

Now your task is to sort the n numbers. That is to say, after all operations, all the n numbers are in the first stack in non-decreasing order from top to bottom.

Input

The first line of input contains an integer n ($1 \le n \le 10000$), denoting the number of integers.

The second line contains n integers $A_1, A_2, \ldots, A_n$ ($1 \le A_i \le 20000$), denoting the numbers in the first stack from bottom to top. (A_1 is the bottom and A_n is the top)

Output

The first line of output contains an integer m, denoting the number of operations.

The second line contains m integers, denoting your operations. Pay attention that you can not choose 1 or 2 when the first stack is empty. And the total times that you choose 3 and 4 can not exceed 20.

If there are multiple solutions, you can print any one of them.

Examples

standard input	standard output
2 100 200	3 1 1 3
3 200 300 100	6 1 2 2 3 2 4