A. Lucky Sum of Digits

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Petya loves lucky numbers. We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

Petya wonders eagerly what minimum lucky number has the sum of digits equal to n. Help him cope with the task.

Input

The single line contains an integer n $(1 \le n \le 10^6)$ — the sum of digits of the required lucky number.

Output

Print on the single line the result — the minimum lucky number, whose sum of digits equals n. If such number does not exist, print -1.

Sample test(s)

input

```
11
```

output

```
47
```

input

```
10
```

output

```
-1
```

B. Lucky Probability

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Petya loves lucky numbers. We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

Petya and his friend Vasya play an interesting game. Petya randomly chooses an integer p from the interval $[p_l, p_r]$ and Vasya chooses an integer v from the interval $[v_l, v_r]$ (also randomly). Both players choose their integers equiprobably. Find the probability that the interval $[min(v, p), max(v, p)]$ contains exactly k lucky numbers.

Input

The single line contains five integers p_l, p_r, v_l, v_r and k ($1 \le p_l \le p_r \le 10^9$, $1 \le v_l \le v_r \le 10^9$, $1 \le k \le 1000$).

Output

On the single line print the result with an absolute error of no more than 10^{-9}.

Sample test(s)

```
input
1 10 1 10 2
output
0.320000000000
```

```
input
5 6 8 10 1
output
1.000000000000
```

Note

Consider that $[a, b]$ denotes an interval of integers; this interval **includes** the boundaries. That is, $[a, b] \overset{\text{def}}{=} \{x \in \mathbb{R} : a \leqslant x \leqslant b\}$

In first case there are 32 suitable pairs: $(1, 7), (1, 8), (1, 9), (1, 10), (2, 7), (2, 8), (2, 9), (2, 10), (3, 7), (3, 8), (3, 9), (3, 10), (4, 7), (4, 8), (4, 9), (4, 10), (7, 1), (7, 2), (7, 3), (7, 4), (8, 1), (8,$
. Total number of possible pairs is $10 \cdot 10 = 100$, so answer is $32 / 100$.

In second case Petya always get number less than Vasya and the only lucky 7 is between this numbers, so there will be always 1 lucky number.

C. Lucky Tree

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Petya loves lucky numbers. We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

One day Petya encountered a tree with n vertexes. Besides, the tree was weighted, i. e. each edge of the tree has weight (a positive integer). An edge is lucky if its weight is a lucky number. Note that a *tree with n vertexes* is an undirected connected graph that has exactly $n - 1$ edges.

Petya wondered how many vertex triples (i, j, k) exists that on the way from i to j, as well as on the way from i to k there must be at least one lucky edge (all three vertexes are pairwise distinct). The order of numbers in the triple matters, that is, the triple $(1, 2, 3)$ is not equal to the triple $(2, 1, 3)$ and is not equal to the triple $(1, 3, 2)$.

Find how many such triples of vertexes exist.

Input

The first line contains the single integer n $(1 \le n \le 10^5)$ — the number of tree vertexes. Next $n - 1$ lines contain three integers each: u_i v_i w_i $(1 \le u_i, v_i \le n, 1 \le w_i \le 10^9)$ — the pair of vertexes connected by the edge and the edge's weight.

Output

On the single line print the single number — the answer.

Please do not use the %lld specificator to read or write 64-bit numbers in C++. It is recommended to use the cin, cout streams or the %I64d specificator.

Sample test(s)

input

```
4
1 2 4
3 1 2
1 4 7
```

output

```
16
```

input

```
4
1 2 4
1 3 47
1 4 7447
```

output

```
24
```

Note

The 16 triples of vertexes from the first sample are: $(1, 2, 4), (1, 4, 2), (2, 1, 3), (2, 1, 4), (2, 3, 1), (2, 3, 4), (2, 4, 1), (2, 4, 3), (3, 2, 4), (3, 4, 2), (4, 1, 2), (4, 1, 3), (4, 2, 1), (4, 2, 3), (4, 3, 1), (4, 3, 2)$.

In the second sample all the triples should be counted: $4 \cdot 3 \cdot 2 = 24$.

D. Lucky Sorting

time limit per test: 3 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Petya loves lucky numbers. We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

Petya got an array consisting of n numbers, it is the gift for his birthday. Now he wants to sort it in the non-decreasing order. However, a usual sorting is boring to perform, that's why Petya invented the following limitation: one can swap any two numbers but only if at least one of them is lucky. Your task is to sort the array according to the specified limitation. Find any possible sequence of the swaps (the number of operations in the sequence should not exceed $2n$).

Input

The first line contains an integer n $(1 \le n \le 10^5)$ — the number of elements in the array. The second line contains n positive integers, not exceeding 10^9 — the array that needs to be sorted in the non-decreasing order.

Output

On the first line print number k $(0 \le k \le 2n)$ — the number of the swaps in the sorting. On the following k lines print one pair of **distinct** numbers (a pair per line) — the indexes of elements to swap. The numbers in the array are numbered starting from 1. If it is impossible to sort the given sequence, print the single number -1.

If there are several solutions, output any. Note that you don't have to minimize k. Any sorting with no more than $2n$ swaps is accepted.

Sample test(s)

input

```
2
4 7
```

output

```
0
```

input

```
3
4 2 1
```

output

```
1
1 3
```

input

```
7
77 66 55 44 33 22 11
```

output

```
7
1 7
7 2
2 6
6 7
3 4
5 3
4 5
```

E. Lucky Interval

time limit per test: 4 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Petya loves lucky numbers. We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

One day Petya came across an interval of numbers $[a, a + l - 1]$. Let $F(x)$ be the number of lucky digits of number x. Find the minimum b $(a < b)$ such, that $F(a) = F(b)$, $F(a + 1) = F(b + 1)$, ..., $F(a + l - 1) = F(b + l - 1)$.

Input

The single line contains two integers a and l $(1 \le a, l \le 10^9)$ — the interval's first number and the interval's length correspondingly.

Output

On the single line print number b — the answer to the problem.

Sample test(s)

input

```
7 4
```

output

```
17
```

input

```
4 7
```

output

```
14
```

Note

Consider that $[a, b]$ denotes an interval of integers; this interval **includes** the boundaries. That is, $[a,b] \overset{\text{def}}{=} \{x \in \mathbb{R} : a \leqslant x \leqslant b\}$