A. Lucky Division

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

Petya calls a number *almost lucky* if it could be evenly divided by some lucky number. Help him find out if the given number n is almost lucky.

Input

The single line contains an integer n $(1 \le n \le 1000)$ — the number that needs to be checked.

Output

In the only line print "YES" (without the quotes), if number n is almost lucky. Otherwise, print "NO" (without the quotes).

Sample test(s)

input
```
47
```
output
```
YES
```

input
```
16
```
output
```
YES
```

input
```
78
```
output
```
NO
```

Note

Note that all lucky numbers are almost lucky as any number is evenly divisible by itself.

In the first sample 47 is a lucky number. In the second sample 16 is divisible by 4.

B. Lucky Substring

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

One day Petya was delivered a string s, containing only digits. He needs to find a string that

- represents a lucky number without leading zeroes,
- is not empty,
- is contained in s as a substring the maximum number of times.

Among all the strings for which the three conditions given above are fulfilled, Petya only needs the lexicographically minimum one. Find this string for Petya.

Input
The single line contains a non-empty string s whose length can range from 1 to 50, inclusive. The string only contains digits. The string can contain leading zeroes.

Output
In the only line print the answer to Petya's problem. If the sought string does not exist, print "-1" (without quotes).

Sample test(s)

input
```
047
```
output
```
4
```

input
```
16
```
output
```
-1
```

input
```
472747
```
output
```
7
```

Note
The lexicographical comparison of strings is performed by the < operator in the modern programming languages. String x is lexicographically less than string y either if x is a prefix of y, or exists such i $(1 \le i \le min(|x|, |y|))$, that $x_i < y_i$ and for any j $(1 \le j < i)$ $x_j = y_j$. Here $|a|$ denotes the length of string a.

In the first sample three conditions are fulfilled for strings "4", "7" and "47". The lexicographically minimum one is "4".

In the second sample s has no substrings which are lucky numbers.

In the third sample the three conditions are only fulfilled for string "7".

C. Lucky Sum

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

Let $next(x)$ be the minimum lucky number which is larger than or equals x. Petya is interested what is the value of the expression $next(l) + next(l + 1) + \ldots + next(r - 1) + next(r)$. Help him solve this problem.

Input

The single line contains two integers l and r ($1 \le l \le r \le 10^9$) — the left and right interval limits.

Output

In the single line print the only number — the sum $next(l) + next(l + 1) + \ldots + next(r - 1) + next(r)$.

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

Sample test(s)

input
```
2 7
```
output
```
33
```

input
```
7 7
```
output
```
7
```

Note

In the first sample: $next(2) + next(3) + next(4) + next(5) + next(6) + next(7) = 4 + 4 + 4 + 7 + 7 + 7 = 33$

In the second sample: $next(7) = 7$

D. Lucky Transformation

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

Petya has a number consisting of n digits without leading zeroes. He represented it as an array of digits without leading zeroes. Let's call it d. The numeration starts with 1, starting from the most significant digit. Petya wants to perform the following *operation* k times: find the minimum x $(1 \le x < n)$ such that $d_x = 4$ and $d_{x+1} = 7$, if x is odd, then to assign $d_x = d_{x+1} = 4$, otherwise to assign $d_x = d_{x+1} = 7$. Note that if no x was found, then the operation counts as completed and the array doesn't change at all.

You are given the initial number as an array of digits and the number k. Help Petya find the result of completing k operations.

Input
The first line contains two integers n and k $(1 \le n \le 10^5, 0 \le k \le 10^9)$ — the number of digits in the number and the number of completed operations. The second line contains n digits without spaces representing the array of digits d, starting with d_1. It is guaranteed that the first digit of the number does not equal zero.

Output
In the single line print the result without spaces — the number after the k operations are fulfilled.

Sample test(s)

input
```
7 4
4727447
```
output
```
4427477
```

input
```
4 2
4478
```
output
```
4478
```

Note
In the first sample the number changes in the following sequence: $4727447 \to 4427447 \to 4427477 \to 4427447 \to 4427477$.

In the second sample: $4478 \to 4778 \to 4478$.

E. Lucky Permutation

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits ***4*** *and* ***7****. For example, numbers* ***47****,* ***744****,* ***4*** *are lucky and* ***5****,* ***17****,* ***467*** *are not.*

One day Petya dreamt of a lexicographically k-th permutation of integers from 1 to n. Determine how many lucky numbers in the permutation are located on the positions whose indexes are also lucky numbers.

Input

The first line contains two integers n and k $(1 \le n, k \le 10^9)$ — the number of elements in the permutation and the lexicographical number of the permutation.

Output

If the k-th permutation of numbers from 1 to n does not exist, print the single number "-1" (without the quotes). Otherwise, print the answer to the problem: the number of such indexes i, that i and a_i are both lucky numbers.

Sample test(s)

input

```
7 4
```

output

```
1
```

input

```
4 7
```

output

```
1
```

Note

A permutation is an ordered set of n elements, where each integer from 1 to n occurs exactly once. The element of permutation in position with index i is denoted as a_i $(1 \le i \le n)$. Permutation a is lexicographically smaller that permutation b if there is such a i $(1 \le i \le n)$, that $a_i < b_i$, and for any j $(1 \le j < i)$ $a_j = b_j$. Let's make a list of all possible permutations of n elements and sort it in the order of lexicographical increasing. Then the lexicographically k-th permutation is the k-th element of this list of permutations.

In the first sample the permutation looks like that:

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

The only suitable position is 4.

In the second sample the permutation looks like that:

```
2 1 3 4
```

The only suitable position is 4.