Jack and Daniel are friends. They want to encrypt their conversations so that they can save themselves from interception by a detective agency so they invent a new cipher.

Every message is encoded to its binary representation. Then it is written down k times, shifted by $0, 1, \cdots, k-1$ bits. Each of the columns is XORed together to get the final encoded string.

If $b = 1001011$ and $k = 4$ it looks like so:

```
1001011     shift 0
01001011    shift 1
001001011   shift 2
0001001011  shift 3
----------
1110101001  <- XORed/encoded string s
```

Now we have to decode the message. We know that $k = 4$. The first digit in $s = 1$ so our output string is going to start with 1. The next two digits are also 1, so they must have been XORed with 0. We know the first digit of our 4^{th} shifted string is a 1 as well. Since the 4^{th} digit of s is 0, we XOR that with our 1 and now know there is a 1 in the 4^{th} position of the original string. Continue with that logic until the end.

Then the encoded message s and the key k are sent to Daniel.

Jack is using this encoding algorithm and asks Daniel to implement a decoding algorithm. Can you help Daniel implement this?

Function Description

Complete the *cipher* function in the editor below. It should return the decoded string.

cipher has the following parameter(s):

- *k*: an integer that represents the number of times the string is shifted
- *s*: an encoded string of binary digits

Input Format

The first line contains two integers n and k, the length of the original decoded string and the number of shifts.
The second line contains the encoded string s consisting of $n + k - 1$ ones and zeros.

Constraints

$1 \leq n \leq 10^6$
$1 \leq k \leq 10^6$
$|s| = n + k - 1$
It is guaranteed that s is valid.

Output Format

Return the decoded message of length n, consisting of ones and zeros.

Sample Input 0

```
7 4
1110100110
```

Sample Output 0

```
1001010
```

Explanation 0

```
1001010
 1001010
  1001010
   1001010
```

```
----------
1110100110
```

Sample Input 1

```
6 2
1110001
```

Sample Output 1

```
101111
```

Explanation 1

```
101111
 101111
-------
1110001
```

Sample Input 2

```
10 3
1110011011
```

Sample Output 2

```
10000101
```

Explanation 2

10000101 010000101

0010000101

1110011011