Highest Value Palindrome ☆

Palindromes are strings that read the same from the left or right, for example *madam* or *0110*.

You will be given a string representation of a number and a maximum number of changes you can make. Alter the string, one digit at a time, to create the string representation of the largest number possible given the limit to the number of changes. The length of the string may not be altered, so you must consider 0's left of all higher digits in your tests. For example 0110 is valid, 0011 is not.

Given a string representing the starting number and a maximum number of changes allowed, create the largest palindromic string of digits possible or the string -1 if it's impossible to create a palindrome under the contstraints.

Input Format

The first line contains two space-separated integers, n (the number of digits in the number) and k (the maximum number of changes allowed), respectively.
The second line contains an n-digit string of numbers that you are to attempt to make palindromic.

Constraints

- $0 < n \leq 10^5$
- $0 \leq k \leq 10^5$
- Each character i in the number is an integer where $0 \leq i \leq 9$.

Output Format

Print a single line with the largest number that can be made by changing no more than k digits; if this is not possible, print `-1`.

Sample Input 0

```
4 1
3943
```

Sample Output 0

```
3993
```

Sample Input 1

```
6 3
092282
```

Sample Output 1

```
992299
```

Sample Input 2

Author	nabila_ahmed
Difficulty	Medium
Max Score	30
Submitted By	17814

NEED HELP?

View discussions
View editorial
View top submissions

RATE THIS CHALLENGE

☆ ☆ ☆ ☆ ☆

MORE DETAILS

Download problem statement
Download sample test cases
Suggest Edits

```
4 1
0011
```

Sample Output 2

```
-1
```

Explanation

Sample 0

There are two ways to make 3943 a palindrome by changing no more than $k = 1$ digits:

1. $3943 \rightarrow 3443$
2. $3943 \rightarrow 3993$

$3993 > 3443$, so we print 3993.

Current Buffer (saved locally, editable) Java 7

```
1  import java.io.*;
2  import java.math.*;
3  import java.text.*;
4  import java.util.*;
5  import java.util.regex.*;
6
7  public class Solution {
8
9
10     /*
11      * Complete the highestValuePalindrome function below.
12      */
13     static String highestValuePalindrome(String s, int n, int k) {
14         /*
15          * Write your code here.
16          */
17
18     }
19
20
21     private static final Scanner scan = new Scanner(System.in);
22
23     public static void main(String[] args) throws IOException {
24         BufferedWriter bw = new BufferedWriter(new
   FileWriter(System.getenv("OUTPUT_PATH")));
25
26         String[] nk = scan.nextLine().split(" ");
27
28         int n = Integer.parseInt(nk[0].trim());
29
30         int k = Integer.parseInt(nk[1].trim());
31
32         String s = scan.nextLine();
33
34         String result = highestValuePalindrome(s, n, k);
35
36         bw.write(result);
37         bw.newLine();
38
39         bw.close();
40     }
41 }
```

Line: 1 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code