Reverse String II

Given a string and an integer k, you need to reverse the first k characters for every 2k characters counting from the start of the string. If there are less than k characters left, reverse all of them. If there are less than 2k but greater than or equal to k characters, then reverse the first k characters and left the other as original.

Example:

```
Input: s = "abcdefg", k = 2
Output: "bacdfeg"
```

Restrictions:

1. The string consists of lower English letters only.
2. Length of the given string and k will in the range [1, 10000]

Solution 1

```
public class Solution {
    public String reverseStr(String s, int k) {
        char[] arr = s.toCharArray();
        int n = arr.length;
        int i = 0;
        while(i < n) {
            int j = Math.min(i + k - 1, n - 1);
            swap(arr, i, j);
            i += 2 * k;
        }
        return String.valueOf(arr);
    }
    private void swap(char[] arr, int l, int r) {
        while (l < r) {
            char temp = arr[l];
            arr[l++] = arr[r];
            arr[r--] = temp;
        }
    }
}
```

written by sheva original link here

Solution 2

For every block of 2k characters starting with position i, we want to replace S[i:i+k] with it's reverse.

```python
def reverseStr(self, s, k):
    s = list(s)
    for i in xrange(0, len(s), 2*k):
        s[i:i+k] = reversed(s[i:i+k])
    return "".join(s)
```

written by awice original link here

Solution 3

C++

```
class Solution {
public:
    /**
     * 0            k           2k          3k
     * |-----------|-----------|-----------|---
     * +--reverse--+           +--reverse--+
     */
    string reverseStr(string s, int k) {
        for (int left = 0; left < s.size(); left += 2 * k) {
            for (int i = left, j = min(left + k - 1, (int)s.size() - 1); i < j; i++
, j--) {
                swap(s[i], s[j]);
            }
        }
        return s;
    }
};
```

Java

```
public class Solution {
    public String reverseStr(String s, int k) {
        char[] ca = s.toCharArray();
        for (int left = 0; left < ca.length; left += 2 * k) {
            for (int i = left, j = Math.min(left + k - 1, ca.length - 1); i < j; i+
+, j--) {
                char tmp = ca[i];
                ca[i] = ca[j];
                ca[j] = tmp;
            }
        }
        return new String(ca);
    }
}
```

written by alexander original link here