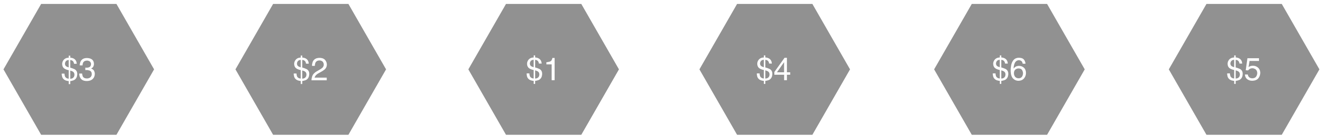
Amazon Online Assessment Coding Questions

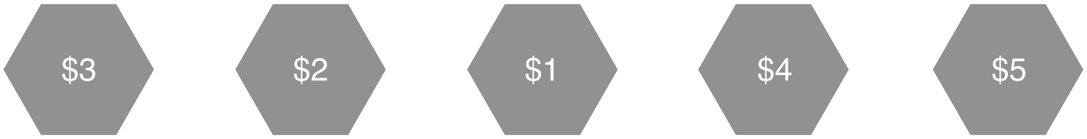
Question 1: New Year's Day is around the corner and Amazon is having a sale. They have a list of items they are considering but they may need to remove some of them. Determine the minimum number of items to remove from an array of prices so that the sum of prices of any k items does not exceed a threshold.

Note: If the number of items in the list is less than K, then there is no need to remove any more items.

Example:
prices = [3, 2, 1, 4, 6, 5]
k = 3
threshold = 14



The sum of prices for every k= 3 items must not be more than threshold = 14. The sum of the prices of the last three items is 6 + 5 + 4 = 15. The item priced $6 can be removed leaving:



No 3 item’s prices sum is greater than 14. Only 1 item needs to be removed.

Function Description:
Complete the function reduceGifts in the editor below.

reduceGifts has the following parameters:
int prices[n]: the prices of each item
int k: the number of items to sum
int threshold: the maximum price of k items

Returns
int: the maximum number of items to remove from the list

Constrains
1 <= k <= n <= 10^5
1 <= threshold <= 10^9
1 <= prices[i] <= 10^9

Sample Cases

```
STDIN       FUNCTION


6            —> prices[ ] size n = 6
9            —> prices = [9, 6, 7, 2, 7, 2]
6
7
2
7
2
2           —> k = 2
13          —> threshold =13
```

Sample Output

```
2
```

Explanation

Items with the price 9 and 7 have a sum larger than 13. After removing these two items, prices' = [6, 2, 7, 2]. No k items have a sum of prices greater than the given threshold.

Question 2:

Amazons database doesn't support very large numbers, so numbers are stored as a string of binary characters, '0' and '1'. Accidentally, a !' was entered at some positions and it is unknown whether they should be '0' or '1'.

The string of incorrect data is made up of the characters '0', '1' and !' where !'is the character that got entered incorrectly. !' can be replaced with either '0' or '1. Due to some internal faults, some errors are generated every time '0' and 1' occur together as '01' or '10 in any subsequence of the string. It is observed that the number of errors a subsequence '01' generates is x, while a subsequence '10' generates y errors.

Determine the minimum total errors generated. Since the answer can be very large, return it modulo 10^9+7.

For example, given string errorString = "101!1", x = 2, y = 3,

- If the !' at index 3 is replaced with 'O, the string is "10101". The number of times the subsequence 01 occurs is 3 at indices (1, 2), (1, 4), and (3, 4). The number of times the subsequence 10 occurs is also
3, indices (0, 1), (0, 3) and (2, 3). The number of errors is 3 * x+3* y= 6 + 9 = 15.
- If the "! is replaced with '1', the string is "10111". The subsequence 01 occurs 3 times and10 occurs 1 time. The number of errors is 3 * x + y= 9.

The minimum number of errors is min(9, 15) modulo (10^9 + 7) = 9.

Note: A subsequence of a string is obtained by omitting zero or more characters from the original string without changing their order.

Hint: It can be proved that (a + b) % c = (a % c) + (b% c)) % c where a, b, and c are integers and % represents the modulo operation.

Function Description
Complete the function getMinErrors in the editor below.

getMinErrors has the following parameter(s):
string errorString: a string of characters '0', '1'; and '!'
int x: the number of errors generated for every occurrence of subsequence 01
int y: the number of errors generated for every occurrence of subsequence 10

Returns
int: the minimum number of errors possible, modulo 10°+7

Constraints
- 1 ≤ len(errorString) ≤ 10^5
- 0 ≤ x, y≤ 10^5
- s consists only of characters '0, '1', and

Input Format For Custom Testing

The first line contains a string, errorString.

The next line contains an integer, x.
The next line contains an integer, y.

Sample Case 0:

Sample Input For Custom Testing

```
STDIN      FUNCTION

01!0        —> errorString = “01!0”
2           —> x = 2
3           —> y = 3
```

Sample Output
8

Explanation:
The better string is 0100 with one substring 01 and two substring 10 marking total errors generated = 2x1 + 3x2 = 8.

Sample Case 1:

Sample Input For Custom Testing

```
STDIN      FUNCTION

! ! ! ! ! ! !   —> errorString = “! ! ! ! ! ! !”
23                —> x = 23
47                —> y = 47
```

Sample Output
0

Explanation:
There is a tie for the best string generated, 0 0 0 0 0 or 1 1 1 1 1, with zero substrings 01 or 10.