12th EECS Programming Competition

Sponsored by

ShowCode

Wednesday, 20th February 2019

Online judge: https://www.hackerrank.com/tests/866ll6pqd4

Password: prog-2019

☆ Frequency Sorting

An array of integers *arr*, of size *n*, is defined as *[a[0], a[1], ..., a[n-1]*]. You will be given an array of integers to sort. Sorting must first be by frequency of occurrence, then by value. For instance, given an array [4, 5, 6, 5, 4, 3], there is one each of 6's and 3's, and there are two 4's, two 5's. Therefore, the sorted list is [3, 6, 4, 4, 5, 5].

Function Description

Complete the function *customSort* in the editor below. The function **must print** the array, each element on a separate line, sorted ascending first by frequency of occurrence and then by value within frequency.

customSort has the following parameter(s):

arr[arr$_0$...arr$_{n-1}$]: an array of integers to sort

Constraints

- $1 \le n \le 2 \times 10^5$
- $1 \le arr[i] \le 10^6$

▼ Input Format For Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer *n*, the size of the integer array *arr*.

The next *n* lines each contain an element *arr[i]*.

▼ Sample Case 0

Sample Input 0

```
5
3
1
2
2
4
```

Sample Output 0

```
1
3
4
2
2
```

▼ Sample Case 1

Sample Input 1

```
10
8
5
5
5
5
1
1
1
4
4
```

Sample Output 1

```
8
4
4
1
1
1
5
5
5
5
```

☆ Efficient Caretaker

The caretaker at Hacker University is insanely efficient. By the end of each day, all of the waste from the trash cans in the school has been shifted into plastic bags which can carry waste weighing between 1.01 pounds and 3.00 pounds. All of the plastic bags must be dumped into the trash cans outside the school. The caretaker can carry at most 3.00 pounds at once. One trip is described as selecting a few bags which together don't weigh more than 3.00 pounds, dumping them in the outdoor trash can and returning to the school. The caretaker wants to make minimum number of trips to the outdoor trash can. Given the number of plastic bags, *n*, and the weights of each bag, determine the minimum number of trips if the caretaker selects bags in the optimal way.

For example, given $n = 6$ plastic bags weighing $weight = [1.01, 1.99, 2.5, 1.5, 1.01]$, the caretaker can carry all of the trash out in *3* trips: *[1.01 + 1.99 , 2.5, 1.5 + 1.01].*

Function Description

Complete the function *efficientCaretaker* in the editor below. The function must return a single integer that represents the minimum number of trips to be made.

efficientCaretaker has the following parameter(s):

weight[weight[0],...weight[n-1]]: an array of floating-point integers

Constraints

- $1 \leq n \leq 1000$
- $1.01 \leq weight[i] \leq 3.0$

▼ Input Format For Custom Testing

The first line contains an integer, *n*, that denotes the number of elements in *weight*.

Each line *i* of the *n* subsequent lines (where $0 \leq i < n$) contains an integer that describes *weight[i]*.

▼ Sample Case 0

Sample Input For Custom Testing

```
5
1.50
1.50
1.50
1.50
1.50
```

Sample Output

```
3
```

▼ Sample Case 1

Sample Input For Custom Testing

```
4
1.50
1.50
1.50
1.50
```

Sample Output

```
2
```

☆ How Many Strokes?

Julia wants to paint a picture but hates taking the brush off the canvas. In one stroke, Julia can only paint the same coloured cells which are joined via some edge.

Given the painting, determine the minimum number of strokes to completely paint the picture.

Take for example, the canvas with height given by $h = 3$ and width given by $w = 5$ is to be painted with picture *picture=["aabba", "aabba", "aaaca"]*, the diagram below shows the *4* strokes needed to paint the canvas.

```
       Strokes
Canvas  1   2  3 4
aabba   aa  bb   a
aabba   aa  bb   a
aaaca   aaa    c a
```

Function Description
Complete the function *strokesRequired* in the editor below. The function must return an integer, the minimum number of strokes required to paint the canvas.

strokesRequired has the following parameter(s):
picture[picture[0],...picture[h-1]]: an array of strings where each string represents one row of the picture to be painted

Constraints

- $1 \le h \le 10^5$
- $1 \le w \le 10^5$
- $1 \le h*w \le 10^5$
- *len(picture[i]) = w (where $0 \le i < h$)*
- *picture[i][j] ∈ {'a', 'b', 'c'} (where $0 \le i < h$ and $0 \le j < w$)*

▼ Input Format For Custom Testing

The first line contains an integer, *h*, that denotes the height of the picture and the number of elements in *picture*.

Each line *i* of the *h* subsequent lines (where $0 \le i < h$) contains a string that describes *picture[i]*.

▼ Sample Case 0

Sample Input For Custom Testing

```
3
aaaba
ababa
aaaca
```

Sample Output

```
5
```

Explanation
The 'a's can be painted in *2* strokes, 'b's in *2* strokes and 'c' in *1* stroke, for a total of 5.

```
       Strokes
Canvas  1   2 3 4 5
aaaba  aaa    b   a
ababa  a a  b b   a
aaaca  aaa      c a
```

▼ Sample Case 1

Sample Input For Custom Testing

```
4
bbba
abba
acaa
aaac
```

Sample Output

```
4
```

Explanation
The 'a's can be painted in *1* stroke, the 'b's in *1* stroke and each 'c' requires *1* stroke.

```
       Strokes
Canvas  1   2   3 4
bbba   bbb    a
abba    bb a  a
acaa       a aa c
aaac       aaa    c
```

☆ Magical Subsequence

We define a *magical subsequence* to be a sequence of letters within a string that contains all five vowels in order: *a, e, i, o, u*. There can be any number of occurrences of each vowel, but they must be in that order. For instance, *aeeiou* is a magical subsequence, but *aeaioua* is not. The magical subsequences of each of these strings would be *aeeiou* and *aeiou*, with lengths of 6 and 5 respectively.

Julia has a string, *s*, consisting of one or more of the following letters: *a*, *e*, *i*, *o*, and *u*. She wants to determine the longest magical subsequence in her string.

Function Description

Complete the function *longestSubsequence* in the editor below. The function must return the length of the longest magical subsequence within the input string. If one does not occur in the string, return *0*

longestSubsequence has the following parameter(s):

s: the string to analyse

Constraints

- $5 < |s| < 5 \times 10^5$
- String *s* is composed of English vowels (i.e., *a*, *e*, *i*, *o*, and *u*).

▼ Input Format For Custom Testing

Input from stdin will be processed as follows and passed to the function.

There is one line containing *s*.

▼ Sample Case 0

Sample Input For Custom Testing

```
aeiaaioooaauuaeiou
```

Sample Output

```
10
```

Explanation

In the table below, the component characters of the longest magical subsequence are red:

a	e	i	a	a	i	o	o	o	a	a	u	u	a	e	i	o	u

▼ Sample Case 1

Sample Input For Custom Testing

```
aeiaaioooaa
```

Sample Output

```
0
```

Explanation

String *s* does not contain the letter *u*, so it is not possible to construct a magical subsequence.

☆ The Duel

Gandalf has challenged his old friend and new enemy Merlin to a duel at a place of Merlin's choosing. Merlin decides the best place to be is behind a lot of minor wizards that Gandalf will have to fight first. As each minor wizard is encountered, he will fight and lose power. Along the path, he also may find power packs. Given that his power must never drop below *1*, what is the minimum power he must start out with so he can get to the duel?

For example, his path is described as *p = [-2, 3, 1, -5]* where a negative number represents an opponent and a positive value is a power pack. When he fights an opponent, his power drops by the absolute value of the opponent's value. When he picks up a power pack, his power is boosted by the value. The following shows Gandalf's energy level if he starts with *4* units of power.

```
Gandalf's   Enemy
Power       or boost
---------   --------
4           -2
2            3
5            1
6           -5
1
```

Function Description

Complete the function *minPower* in the editor below. The function must return the minimum integer energy value Gandalf needs when he starts out.

minPower has the following parameter(s):

p[p[0],...p[n-1]]: an array of integers

Constraints

- $1 \le n \le 10^5$
- $-100 \le p[i] \le 100$

▼ Input Format For Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer *n*, the size of the array *p*.

Each of the next *n* lines contains an integer *p[i]*.

▼ Sample Case 0

Sample Input 0

```
10
-5
4
-2
3
1
-1
-6
-1
0
5
```

Sample Output 0

```
8
```

▼ Sample Case 1

Sample Input 1

```
5
-5
4
-2
3
1
```

Sample Output 1

```
6
```

Explanation 1

```
Gandalf's   Enemy
Power       or boost
---------   --------
6           -5
1            4
5           -2
3            3
6            1
7
```

☆ Angry Animals

Pi's father, Danny, runs the Hackerville Zoo. He is moving to Rookieville and wants to take all of the zoo animals with him via ship. He is confused about how to arrange them because a few of the species cannot be kept together in the same cabin.

There are *n* animals placed in a straight line. Each animal is identified by a unique number from *1* to *n* in order. There are *m* pairs *(a[i], b[i])* which imply that animals *a[i]* and *b[i]* are enemies and should not be kept in the same cabin. Pi is good at solving problems and he came up with following challenge: count the number of different groups that do not contain any pair such that they are enemies. A group is defined as an interval (x, y) such that all animals in the range from x to y form a group. Determine the number of groups that can be formed according to the Pi's challenge.

For example, given $n = 3$ animals and $m = 3$ pairs of enemies, $a = [1, 2, 3]$ and $b = [3, 3, 1]$, animal *1* is an enemy of animal *3*, and animal *3* is an enemy of animals *1* and *2*. Because *3* is an enemy of both *1* and *2*, it must be in its own cabin. Animals *1* and *2* can be roomed together or separately. There are four possible groupings meeting the constraints: *{1, 2}* , *{1}, {2}, {3}*. Note that the intervals are along the original line of animals numbered consecutively from *1* to *n*, i.e. *[1, 2, 3]* in this case. They cannot be reordered. Note also that the representation given of enemy animals may not be minimal, e.g. with $a = [1, 2, 3]$ and $b = [3, 3, 1]$ the fact that *1* and *3* are enemies is recorded twice.

Function Description
Complete the function *angryAnimals* in the editor below. The function must return the number of groups that can be formed according to Pi's challenge.

angryAnimals has the following parameters:
n: an integer that denotes the number of unique animals
a[a[0],...a[m-1]]: an array of integers
b[b[0],...b[m-1]]: an array of integers

Constraints
- $1 \le n \le 10^5$
- $1 \le m \le 10^6$
- $1 \le a[i], b[i] \le n$

▼ Input Format For Custom Testing

The first line contains an integer, *n*.
The second line contains an integer, *m*, that denotes the number of elements in *a*.

Each line *i* of the *m* subsequent lines (where $0 \le i < m$) contains an integer that describes *a[i]*.
The next line **again** contains the integer, *m*, that denotes the number of elements in *b* (***note**: a and b will have the same length).*

Each line *i* of the *m* subsequent lines (where $0 \le i < m$) contains an integer that describes *b[i]*.

▼ Sample Case 0

Sample Input For Custom Testing

```
4
2
1
2
2
3
4
```

Sample Output

```
7
```

Explanation
(1), (1,2), (2), (2,3), (3), (3,4), (4) are the groups that can be formed according to Pi's challenge.

▼ Sample Case 1

Sample Input For Custom Testing

```
5
2
1
2
2
3
5
```

Sample Output

```
11
```

Explanation
(1), (1,2), (2), (2,3), (2,3,4), (3), (3,4), (3,4,5), (4), (4,5), (5) are the groups that can be formed according to Pi's challenge.