Weakest Link

Rob Anderson is creating a spooky decoration for Halloween. Rob has a bracket in the ceiling, and he will hook on chain links to make a drooping chain to give his room a dungeony feel. Rob has a large collection of chain links to make the decoration. Each link has a particular grade, which represents the amount of weight (in chain links, all of which have equal weight) that can be hung from it without breaking. A grade of weight 0 means that any link hung underneath will snap the link.

Rob's roommate Anne does understand the concepts of link grades, and was told by Rob to hang up the links. Anne will take each link (in order of Rob's collection) and hang it on the lowest hanging chain or on the bracket, if no chains are currently hanging from the ceiling. If a chain section falls to the ground Anne will simply ignore it. This means that there is a very high chance that Rob will come home to a large number of chain links (broken and not) lying on the floor.

Problem

Take in a the grades of Rob's chain link collection, and determine how many links will break, which ones break, and how many chains should be hanging from the ceiling.

Input Specification

The input will contain multiple lines. The **i**-th line of input will contain a non-negative integer $\mathbf{g_i}$, representing the grade of the **i**-th link in the chain link collection. Input will be terminated with a negative grade.

Output Specification

Every time a link breaks, a line containing the phrase

```
Chain broken at link X. New height is Y.
```

Should be printed, where X is the value representing the chain link in input that broke a link, and Y is the number of chains currently hanging from the ceiling. The last 2 lines of output should be the following,

```
Ending chain height is X.
```

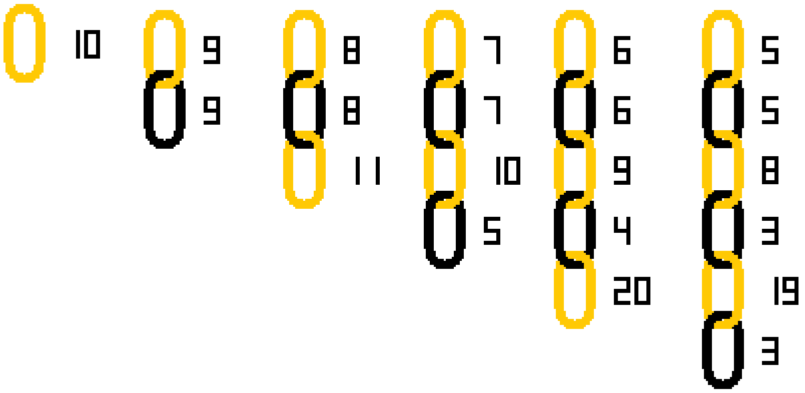
```
There are Y broken links and Z whole links on the floor.
```

The value X should be how many links are hanging from the ceiling, the value Y will be how many links broke during Annes link hanging, and Z will be the number of links that never broke, but fell due to high links breaking.

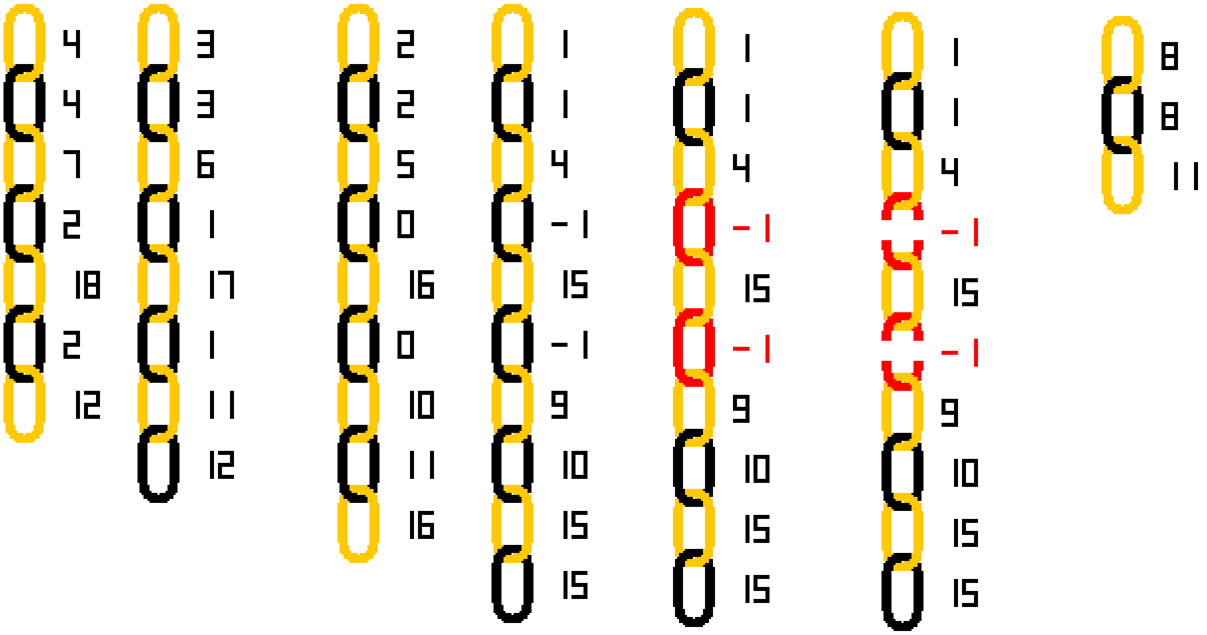
Sample Input	Sample Output
10 9 11 5 20 3 12 12 16 15 15 4 19 5 3 -1	Chain broken at link 10. New height is 3. Ending chain height is 8. There are 2 broken links and 5 whole links on the floor.
1 1 1 1 1 1 -1	Chain broken at link 3. New height is 0. Chain broken at link 6. New height is 0. Ending chain height is 0. There are 2 broken links and 4 whole links on the floor.
8 6 7 5 3 1 0 -1	Ending chain height is 7. There are 0 broken links and 0 whole links on the floor.

Explanation

In the first case we have the following happen

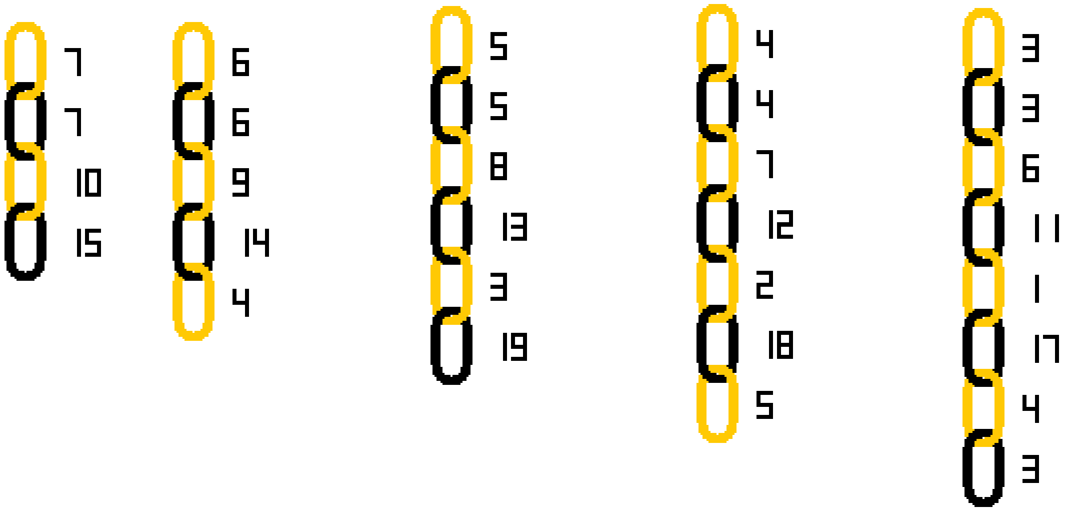


We add the first link which allows for 10 links underneath. The second link allows for 9 links underneath, and the first link can now support only 9 additional links. The third link is better than the first two, so the links we really need to care about are the two links that will allow for only 8 additional links. The 4th link (a grade 5 link) allows for only 5 additional links, and at this point the first two links allow for 7 additional links. Thus the 5 will break before the 7's. The next link is the strongest by far and will not cause issues prior to higher up chains breaking. Once we add the next link (the grade 3 link) we now have two links that will fail when 4 more links are added. We continue to add stronger links until the 3-grade and the 5-grade link break.



Once we add the link that creates too long of a chain (the grade 15 link). The chain snaps at two locations.

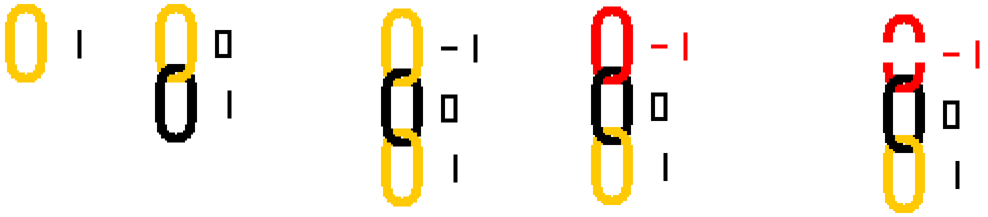
The chain reverts to a smaller chain similar to one we have seen before. Every link we add afterwards does not cause any issues for any of the prior links.



Note that only 2 links broke, 5 links fell to the floor, and the number of links in the current chain is 8.

In Case 2

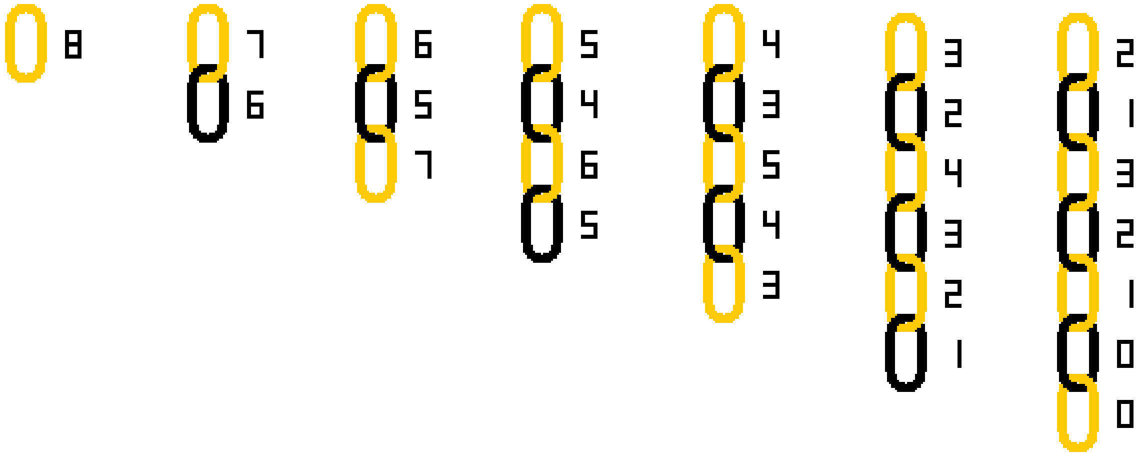
We add a very bad link. However, it can at least support 1 link, which it does until the third link is added.



The chain becomes empty and we repeat the same process of adding a bad link and breaking it. 2 links are broken and 4 lines fall to the floor.

In Case 3

No links break. Here is the resulting supportable link numbers.



Solution Thoughts

I recommend for this assignment creating a stack that stores not only the value that some link can support, but also how many links a chain can support at some given point.

```
struct Node {
   int value, maxSupport;
   Node * next;
};
```

This way you can determine when one link breaks, how many links will fall and how many will break. This sort of data structure will store the maximum number of links that can be supported, which will be computed in the next “Node” by minning with the current maxSupport (sort of) and the current links support. The maxSupport value should decrease across the chain, since the chain will only be able to support fully what the weakest link can support.

Grading Details

Read/Write from/to standard input/output – 10 points

Good comments, whitespace, and variable names – 15 points

No extra input output (e.g. input prompts, “Please enter the number of words”) – 10 points

Write a stack – 10 points

Utilize the previous support value – 5 points

Your program will be tested on 10 test cases – 5 points each

No points will be awarded to programs that do not compile using gcc -std=gnu11 (gnu “eleven”).

Sometimes a requested technique will be given, and solutions without the requested technique will have their maximum points total reduced. For this problem you must use a stack. ***Without a stack your program will earn at most 50 points!***

Any case that causes your program to return a non-zero error return code will be treated as completely wrong. Additionally any case that takes longer than the maximum allowed time (the max of {5 times my solution, 5 seconds}) will also be treated as wrong. You will most likely need to write an efficient stack whose operations will run in amortized O(1) per operation.

No partial credit will be awarded for an incorrect case.

Note: There is no given bound on the number of links. Static memory might have severe issues on the larger test cases.

Follow the output format as specified for full points.