

## Reading Poetry Backwards – RPB v1.0

This assignment requires you to study, understand, and augment a Python program that (re)writes or “breaks” poems in various “deformed” manners, including printing the lines of a poem in reverse (last line to first line) or randomized orders or printing the individual words of a poem in complete reverse order.



Asking you to “read poems backwards” is not original with us (cf. Leithauser, 2013). In their article “Deformance and Interpretation”, McGann and Samuels (1999, p25) note:

In an undated fragment on a leaf of stationery, Emily Dickinson wrote what appears to be one of her “letters to the world”:

*‘Did you ever read one of her Poems backward, because the plunge from the front overturned you? I sometimes (often have, many times) have -- a Something overtakes the Mind –’*  
(Prose Fragment 30)

In addition to discovering varied meanings in a poem when reading in new ways (sometimes called “breaking a poem”), you may have come across poems that invite varying interpretations depending on the order read. For instance, “Lost Generation” by Jonathan Reed is a pessimistic view of the world when read from top to bottom, but sounds optimistic when the lines are read backwards (bottom to top). We encourage you to look it up and see for yourself.

Your program will help the reader more easily “read” poems in various deformed manners. For example with English poems, to read the lines of a poem “backwards”, that is from the last line to the first line, the (human) reader must concentrate to move their eyes “up” to the preceding line on the page upon finishing a line. So to aid the reader who wishes to read the lines of a poem in reverse order, your program will produce a version of the poem where the last line of the poem appears at the top of the poem. The next to the last line of the poem will appear second, and so on. Here is a trivial example that shows the start and ending of a children’s poem on the left. On the right are the lines of the poems printed “backwards” with the original line numbers highlighting to the reader that the lines appear in reverse order.

### **Mary had a little lamb**

**by Sarah Josepha Hale**

*Mary had a little lamb,  
whose fleece was white as snow.*

*And everywhere that Mary went,*

:

:

*"Why does the lamb love Mary so?"  
the eager children cry.*

*"Why, Mary loves the lamb, you know."  
the teacher did reply.*

### ***Lines Backwards***

***Title: Mary had a little lamb***

***Author: Sarah Josepha Hale***

23 *the teacher did reply.*

22 *"Why, Mary loves the lamb, you know."*

21

20 *the eager children cry.*

19 *"Why does the lamb love Mary so?"*

:

:

## Requirements:

- (0) I have given you a Starter Kit that includes a function that does most (but not all) of the work for opening a text file containing a poem and writing the lines of the poem in reverse order into a new file. Your task is to complete the **readingLinesBackwards ( )** function. The code will already produce a file with the lines of the poem printed backwards. Modify this function so that your version also writes the original lines numbers in the poem (which of course should appear in a decreasing order from last line to line #1).

```
23 the teacher did reply.  
22 "Why, Mary loves the lamb, you know."  
21  
20 the eager children cry.  
19 "Why does the lamb love Mary so?  
:  
:
```

- (1) Complete the **randomizeLines ( )** function. This function is similar to your first function, but this time, rather than print the lines of the poem in reverse order, the lines of the poem should appear in a random order. No line numbers should be printed in this new version. The first two lines of a randomized output is shown below:

```
to see a lamb at school.  
but still it lingered near,
```

- (2) Complete the **readingWordsBackwards ( )** function. This function must print all the words in the poem in complete reverse order, although the newlines (line breaks) should be preserved as in the original. Thus the very last word in the poem will be the first word in your new deformed version. This version should print the original line numbers, but of course the words on each line will appear in reverse order. Remove all punctuation in this version and print all words in lowercase letters only. The first two lines of output are shown:

```
23 reply did teacher the  
22 know you lamb the loves mary why
```

- (3) [+3 Extra Credit] Superior effort: **Create your own** deformation method. Write a function that produces a new file with the original poem printed once "deformed" in your method.

Leithauser, Brad (2013). Reading Poems Backward. *The New Yorker*. July 11, 2013.  
<http://www.newyorker.com/online/blogs/books/2013/07/reading-poems-backward.html>

Samuels, Lisa and McGann, Jerome J. (1999). Deformation and Interpretation. *New Literary History*, v30(1), 25-56.