EE231002 Introduction to Programming

Lab10. Word Processing

Due: Nov. 30, 2019

Word processing has been one of the major applications for computers. In this assignment you will try to write a program to perform simple word processing with a fixed character width, such as the workstation terminal display. Today's powerful word processors use a similar concept but with more complicated fonts and more flexible font positioning.

Let's assume the output has N characters per line. Thus, any line which has more than N characters will take more than one output line. It is very likely that line-change happens within a word. This would make the output less legible. Thus, a reasonable word processor would break a line only on word boundaries. Assuming each paragraph is read in as a single string, the number of characters of each paragraph can be less than, equal to or larger than N, your assignment is to print the paragraph in a two-column format with the left column right-edge aligned and the right column left-edge aligned. Example of output is shown at the end of this file.

The first three lines of the input are the title, the author and the date of the article. These three lines should printed at the center of the line. Each line is assumed to have 80 characters.

After that, each input line is assumed to be a paragraph of the article. Your program should read in a paragraph and print it out immediately. The maximum number of characters for each paragraph is assumed to be 1000. Thus, an input buffer of 1000 characters should be declared in your `main` function. The file is ended by a special character `EOF` which is defined in the `<stdio.h>` header file. You should test the input against this character to stop your program execution.

The next step is to break the paragraph into lines, with each line no more than 38 characters. Of course, no words can be broken to two different lines, i.e., line breaking can be only be done at a space character. It is recommended to declare a two dimensional array `outBuffer[52][38]` to store the paragraph. This array should also be in the `main` function.

With the `outBuffer` filled, one can now print out the paragraph in two-column format. The first half of the paragraph is printed on the left column followed by the 2nd half on the right column. The left column is right-edge aligned and the right column left edge aligned. A vertical line with a space character on both left and right sides of the line should be printed between the left and right columns. The blank lines between paragraphs should be printed as it is.

To make your program more general, please define the following macros:

```
#define PN 1000
#define PW 80
#define LN 52
#define CW 38
```

where `PN` is the maximum number of character for a paragraph, `PW` is the page width (for the 3 title lines), `LN` is the number of lines for the output buffer, and `CW` is the column width. Thus,

the input paragraph and the output buffer should be declared in your `main` function as

```
char para[PN];            // input paragraph
char outBuffer[LN][CW];   // output buffer, LN lines with CW chars
```

You are free to declare necessary functions to facilitate your code writing. As usual, your program should efficient, concise and legible.

Notes.

1. Create a directory **`lab10`** and use it as the working directory.
2. Name your program source file as **`lab10.c`**.
3. The first few lines of your program should be comments as the following.

   ```
   // EE231002 Lab10. Word Processing
   // ID, Name
   // Date:
   ```

4. After you finish verifying your program, you can submit your source code by

   ```
   $ ~ee2310/bin/submit lab10 lab10.c
   ```

   If you see a "submitted successfully" message, then you are done. In case you want to check which file and at what time you submitted your labs, you can type in the following command:

   ```
   $ ~ee2310/bin/subrec lab10
   ```

   It will show youe submission records for lab10.

5. You should try to write the program as efficient as possible. The format of your program should be compact and easy to understand. These are part of the grading criteria.

Example output is shown below.

```
$ ./a.out < Jobs.txt
                        You've got to find what you love
                                    Steve Jobs
                                  June 12, 2005

  I am honored to be with you today at | gotten to a college graduation. Today
     your commencement from one of the | I want to tell you three stories from
   finest universities in the world. I | my life. That's it. No big deal. Just
never graduated from college. Truth be | three stories.
   told, this is the closest I've ever |

   The first story is about connecting | the dots.

...
...
...
...
...
...

  Stewart and his team put out several | hitchhiking on if you were so
issues of The Whole Earth Catalog, and | adventurous. Beneath it were the
 then when it had run its course, they | words: "Stay Hungry. Stay Foolish." It
     put out a final issue. It was the | was their farewell message as they
 mid-1970s, and I was your age. On the | signed off. Stay Hungry. Stay Foolish.
 back cover of their final issue was a | And I have always wished that for
photograph of an early morning country | myself. And now, as you graduate to
road, the kind you might find yourself | begin anew, I wish that for you.

             Stay Hungry. Stay Foolish. |

               Thank you all very much. |
```