lab10

```
$ gcc lab10.c

$ ./a.out 69 < US2981877.tex
                        Patent Number: 2,981,877
                 SEMICONDUCTOR DEVICE-AND-LEAD STRUCTURE
                 Robert N. Noyce, Los Altos, California
     Assignor to Fairchild Semiconductor, Mountain View, California
  5            Filed July 30, 1959, Serial Number 830,507
                        10 Claims. (Cl. 317-235)

----
Should be a blank line.
    This invention relates to electrical circuit structures
    incorporating semiconductor devices. Its principal objects are
 10 these: to provide improved device-and-lead structures for making
    electrical connections to the various semiconductor regions; to
    make unitary circuit structures more compact and more easily
    fabricated in small sizes than has heretofore been feasible; and
    to facilitate the inclusion of numerous semiconductor devices
 15 within a single body of material.

----
Should be a blank line.
    In brief, the present invention utilizes dished junctions
    extending to the surface of a body of extrinsic semiconductor, an
    insulating surface layer consisting essentially of oxide of the
 20 same semiconductor extending across the junctions, and leads in
...
...
600 said insulating layer and extending thereover across said
    junction to connect physically and electrically with said first
    contact.

----
Should be a blank line.
    References Cited in the file of this patent UNITED STATES PATENTS
605 2,813,326 Liebowitz Nov. 19, 1957 2,836,878 Shepard June 3, 1958
    2,842,723 Koch et al. July 8, 1958 2,849,664 Beale Aug. 26, 1958
```

```
score: 89.0
o. [Output] Program output is incorrect.
```

```
o. [Coding] lab10.c spelling errors: postion(2), postition(1)
o. [Format] Program format can be improved.
o. [Efficiency] can still be improved.
```

lab10.c

```c
// EE231002 Lab10. Word Processing
// 111060023, 黃柏霖
// Date: 111/12/1

#include <stdio.h>
#include <stdlib.h>

char PARA[1500];     // an input paragraph
int LN = 0;          // line number of printed text
int LW;              // line width of output lines

void print_lines(void);                        // print the index of line
int word_xcd(int line_pos, int curr_pos);     // check if added word exceed limit

int main(int argc, char *argv[])               // get string while input
{
    int i, j;                                  // loop index
    int len = 0;                               // length for a line
    int line_pos = 0;                          // where a char in a line

    LW = atoi(argv[1]) - 4;                    // compute LW
    // first six lines are titles
    for (i = 0; i < 6; i++) {
        len = 0;                               // length is 0 yet
        // read in the text
        for (j = 0; (PARA[j] =  getchar()) != '\n'; j++, len++);
        PARA[j] = '\0';                        // string should end with \0
        print_lines();                         // print the index of line
        // set a title to the middle
        for (j = 0; j < (LW - len) / 2; j++) {
            printf(" ");
        }
        printf("%s\n", PARA);                  // print the title
    }
    // print until EOF
    while ((PARA[0] = getchar ()) != EOF) {
    while ((PARA[0] = getchar()) != EOF) {
        print_lines();                         // print the index of line
        if (PARA[0] == '\n') printf("\n");
        else {
            line_pos = 0;                   // start from 0
            // read in the text
            for (i = 1; (PARA[i] = getchar()) != '\n'; i++);
            // print the text
            for (j = 0; PARA[j] != '\n'; j++) {
                // print a word, each word is separated by a space
                if (PARA[j] != ' ') {
                    printf("%c", PARA[j]);
                    line_pos++;             // move to next position in a line
                }
                // if there is a space, and next word won't exceed the limit
                else if (line_pos = word_xcd(line_pos, j)) {
                    printf(" ");            // print the space
                    line_pos++;             // move to next postion in a line
                }
            }
            printf("\n");                   // paragraph is printed, end line
        }
    }
    return 0;
}

// to print index of a line, if the index is a multiple of 5
// input: no input
// return: no return
// output: print index of a line
void print_lines(void)
{
    LN++;                                   // new line
    if (LN % 5 == 0) printf("%3d ", LN);    // print if index is multiple of 5
    else printf("    ");                    // print no number
}

// to determine whether an added word will exceed the limit of line
// input: int line_pos: which position is it in a line
//        int curr_pos: the current postition which should be examined
// return: return line position as 0 if it exceed
//         otherwise, return line position as input
// output: print a new line and the  line index if it's exceed
int word_xcd(int line_pos, int curr_pos)
{
    int i;                                  // loop index
    int j = line_pos;                       // test for line position

    // find how long a word is
    for(i = curr_pos + 1; PARA[i] != ' ' && PARA[i] != '\n'; i++) {
    for (i = curr_pos + 1; PARA[i] != ' ' && PARA[i] != '\n'; i++) {
        j++;
    }
    if (j < LW)
        return line_pos;                    // return line position if legal
    else {
        printf("\n");                       // print new line
        print_lines();                      // print line index
        return 0;                           // set line postion to 0
    }
}

```