In this lab, you will code 5 functions which conducts a number of operation on a book. The book is represented as a 2-dimensional character array. The book has a fixed number of page and is able to contain fixed size of characters within a page. Each row in the 2D array represents a page in the book.

In the lab exam, you are *NOT* going to write a main function. Instead, you will define a function named "execute", which will be called when you execute your program in lab. Note that, this function will not be called for evaluation and grading. It is provided only to enable you to test your functions manually. Prototype for execute function is as follows:

```
void execute();
```

Prototypes of functions that you will implement will be given in the header file "lab2.h". You will have to define execute() function and the other functions even if you leave them empty. You should include "lab2.h" header in your .c code file by writing the line below:

```
#include "lab2.h"
```

In this header file, a 2D global char array is defined, which represents the book. The name of this array is "book". You can directly access this global variable in your code. The number of page that "book" contains (row count of the array "book") and number of cells within a page (column count of the array "book") are accessed through "NUMPAGE" and "PSIZE" read-only variables, respectively. Note that, your code will be tested with several books each having a different number of pages and page sizes.

The 2D array used for representing the book will contain a character in all cells. That means, if the text represented in a page is shorter than the lenght of the row, empty cells at the end of the row are padded with space character in the input. Similarly, a blank page is represented as a row in the array which has all space characters its cells.

Functions

---

(1). print_book()

This function will print global variable "book" row by row. Print new line character at the end of the rows.

EXAMPLE:

Let "book" defined as the following in global:

```
#define NUMPAGE 5
#define PSIZE 5

char book[NUMPAGE][PSIZE] = {{'a', 'b', 'c', 'd', 'e'}, {'A', 'B', 'C', 'D',
'E'},{'a', 'b', 'c', 'd', 'e'}, {'A', 'B', 'C', 'D', 'E'}, {'a', 'b', 'c',
'd', 'e'}};
```

```
void execute(){

    print_book();
}
```

will print:

```
abcde
ABCDE
abcde
ABCDE
abcde
```

(2). int count_char(char* inputchar)

This function will return number of occurance of the character represented as the character which "inputchar" points to (*inputchar) in "book".

EXAMPLE:

Let "book" defined as the following in global:

```
#define NUMPAGE 5
#define PSIZE 5

char book[5][5] = {{'a', 'b', ' ', 'd', 'e'}, {'A', ' ', 'C', 'D', 'E'}, {'a', 'b', 'c', 'd', 'e'}, {'A', 'B', 'C', 'D', 'E'}, {'a', 'b', 'c', 'd', 'e'}};

void execute(){

    char chr = 'c';
    char* chrp = &chr;

    printf("Number of c character in the book is %d.\n", count_char(chrp));
}
```

will print:

```
Number of c character in the book is 2.
```

(3). int word_count()

This function will return the word count in "book". According to this function, a word is a character sequence which starts with the first character of a page or space character, and ends with the last character of a page or a blank character.

EXAMPLE:

Let "book" defined as the following in global:

```
#define NUMPAGE 5
```

```
#define PSIZE 5

char book[5][5] = {{'a', 'b', ' ', 'd', 'e'}, {'A', ' ', 'C', 'D', 'E'},
        {'a', 'b', 'c', 'd', 'e'}, {'A', 'B', 'C', 'D', 'E'}, {'a', 'b', 'c',
'd', 'e'}};

void execute(){

    printf("Word count of the book is %d.\n", word_count());
}
```

will print:

```
Word count of the book is 7.
```

(4). find_word(char* word, int* result)

This function will find the index which represents the first occurance of the character set given in "word" in "book". "word" is a null terminated character array. Different from word_count() function, this function will return the first occurance of this character set even in the case that it does not start or end with a space character. You will assign the resulting index as the value pointed by "result".

EXAMPLE:

Let "book" defined as the following in global:

```
#define NUMPAGE 5
#define PSIZE 5

char book[5][5] = {{'a', 'b', ' ', 'd', 'e'}, {'A', ' ', 'C', 'D', 'E'},
        {'a', 'b', 'c', 'd', 'e'}, {'A', 'B', 'C', 'D', 'E'}, {'a', 'b', 'c',
'd', 'e'}};

void execute(){

    char word[5] = "DE";
    char* wordp = word;
    int intarr[5];
    int* intp = intarr;

    find_word(wordp, intp);

    printf("The first occurance of the character set DE is %d.\n", *intp);
}
```

will print:

```
The first occurance of the character set DE is 8.
```

(5). void compact()

This function will compact the characters to the cells which locate in the beginning of "book". This function will simply erase space characters in "book" and slide the preceeding non-space characters to the cell which is erased without changing the character order. Your function will pad the empty spaces remaining at the end of the "book" "numpage" is the number of page and "psize" is the number of characters within a page.

EXAMPLE:

Let "book" defined as the following in global:

```
#define NUMPAGE 5
#define PSIZE 5

char book[5][5] = {{'a', 'b', ' ', 'd', 'e'}, {'A', ' ', 'C', 'D', 'E'},
        {'a', 'b', 'c', 'd', 'e'}, {'A', 'B', 'C', 'D', 'E'}, {'a', 'b', 'c',
'd', 'e'}};

void execute(){

    compact();

    print_book();
}
```

will print:

```
abdeA
CDEab
cdeAB
CDEab
cde
```