Exercise 5.3.1

For this exercise, I will provide you with a text file (**test.txt**), and you will write a program that can count the number of characters, words, and lines in the provided file. Your program should be capable of also printing the contents of the file as well as appending new text to the end of the file.

- Function 1: **void menu (void);**
  - Ask the user for a file name
  - Open the file with access mode append and read(a+) and check for success or failure
  - Create an infinite loop. Inside the loop:
    1. call the menu function
    2. ask the user to select an action from the menu (check for valid input)
    3. call the appropriate function based on the user input
    4. the program ends when the user selects (e) for Exit
  - Please see the sample output on the next two pages
- Note that when the file has first opened, any operation will be done starting at the beginning of that file. As the file is read the new position is retained.
- For this exercise, use the function **rewind** (at the end of each function that is modifying or reading the file), to go back to the starting position
- Function 2: **int lineCount (FILE *);**
  - Count non-empty lines. One way to do this is to loop with **getc** and to count the newline (**'\n'**) character when the line is not empty.
- Function 3: **int wordCount (FILE *);**
  - Count the number of words in the file; a character is a word. One way to do this is to loop with **fscanf** until you get to the end of file character (**EOF**).
- Function 4: **int charCount (FILE *);**
  - Count the number of characters in the file. One way to do this is to loop with **getc** but not counting white spaces (**' '**) and the newline (**'\n'**) characters.
- Function 5: **void printFile (FILE * fp);**
  - Print the contents of the file. One way to do this is to loop with **getc** capturing and displaying everything until you get to the end of file character (**EOF**)
- Function 6: **void addStringToFile (FILE * fp);**
  - Append a string to the end of the file. One way to do this is to use the function **gets** to capture the user string, **fputc** to a new line character (**'\n'**) and **fputs** to add the string to the file

```
"C:\Users\haida\Documents\The Tech Series - C Language\Exrecises\exercise-5-3-1.exe"
Please enter a file name:  test.txt

File test.txt was opened successfully

-c            Print the character count
-w            Print the word count
-l            Print the non-empty line count
-p            Print contents of the file
-a            Add a string to the end of the file
-e            Exit
   Please enter an operation: c

There are 51 characters in file test.txt

-c            Print the character count
-w            Print the word count
-l            Print the non-empty line count
-p            Print contents of the file
-a            Add a string to the end of the file
-e            Exit
   Please enter an operation: w

There are 18 words in file test.txt

-c            Print the character count
-w            Print the word count
-l            Print the non-empty line count
-p            Print contents of the file
-a            Add a string to the end of the file
-e            Exit
   Please enter an operation: l

There are 3 lines in file test.txt

-c            Print the character count
-w            Print the word count
-l            Print the non-empty line count
-p            Print contents of the file
-a            Add a string to the end of the file
-e            Exit
   Please enter an operation: p
1- this is a test file

   2- this is a test file

3- this is a test file

-c            Print the character count
-w            Print the word count
-l            Print the non-empty line count
-p            Print contents of the file
-a            Add a string to the end of the file
-e            Exit
   Please enter an operation: a
Please type a string to be added to the file: Hi, my name is Haidar
The string "Hi, my name is Haidar" was added to the file
-c            Print the character count
-w            Print the word count
-l            Print the non-empty line count
-p            Print contents of the file
-a            Add a string to the end of the file
-e            Exit
   Please enter an operation:
```

```
-c          Print the character count
-w          Print the word count
-l          Print the non-empty line count
-p          Print contents of the file
-a          Add a string to the end of the file
-e          Exit
   Please enter an operation: e

Thank you, exiting...

Process returned 0 (0x0)   execution time : 137.371 s
Press any key to continue.
```

- Please save and submit your source code (.c file) solution.
- Solution file naming convention:
  - First_name_Last_name_exercise_number.c
  - Please use (-) to indicate the exercise number
  - Example: Haidar_Alaubiydy-5-3-1.c