Subject: PRF192- PFC
Workshop 08: FILES

Objectives:

- Practicing skills at analyzing and implementing the Files
- Practicing mad making programs using user-defined functions with multiple parameters.
- Upon successful completion of this workshop, you will have demonstrated the ability to read sequential text files.

Program 1 (3 marks)

WORD WRAP INTO TEXT FILES

We use word-wrapping in text-editing programs and text-presentation programs such as web browsers.

Design and code a function that converts a null-terminated string into word-wrapped format of specified width. The converted string, when displayed, fits into a field of the specified width. Your word-wrap function receives a null-terminated string and the field width and returns the number of lines needed to display the wrapped string. Your function also returns the wrapped string through the same parameter as the original string. You may assume that

- a word is a set of consecutive non-whitespace characters,
- no word in the string is longer than the field width, and
- any two consecutive words are separated by whitespace characters.

The function prototype should look like this:

```
int wordwrap(char wrapstring[], int fieldwidth);
```

Write a main program to test your function. Include strings with leading, trailing, multiple embedded whitespace and multiple-whitespace only in your set of test cases. Write the your returns the wrapped string into "**result.txt**" file.

For example, consider the following string and a field width of 10

```
This is week 13 of BTP100
```

For this, your program should write the following displays in to "**result.txt**":

```
This is
week 13 of
BTP100
```

Program 2 (3 marks)

RECORD SEARCH

Design and code a function that searches for the presence of a particular string within a file and displays each record that contains the search string. It should also return a value of 0 if the search string was found in one or more lines, and 1 of the search string was not found in the file at all. This is similar to a simplified version of the "grep" command, working with a search string instead of a regular expression.

The function prototype should look like this:

```
int mygrep(char filename[], char searchstring[]);
```

Write a main program to test your function.

For example, if a file named `problems.txt` contains

```
Design and code a function that searches
for the presence of a particular string
within a file and displays each record
that contains the search string.
```

then a function call **mygrep("problems.txt", "string")** would return the value 0 and display:

```
BTP100 grep
===========
File name : problems.txt
Search string : string
Lines that contain 'string'
for the presence of a particular string
that contains the search string.
```

Program 3 (4 marks)

INTRODUCTION

Statistics helps summarize data in interesting and meaningful ways. The data may consist of values from a variety of sources. Statistics identifies what is common and the degree of variation about what is common. In cases where a data set has several aspects, statistics determines the extent to which one aspect of the set is related to another aspect of the set.

For example, consider a video store with a large selection of rental videos at various prices. We determine the average price of a video in the store, the degree of price variation about this average and then infer something about the clientele who rent videos from that store.

If we suspect that the price of each video was determined by the date of its original release, we use statistics to determine numerically if there is indeed any correlation between the rental price and the original release date. Moreover, we determine the degree of any such correlation.

STATISTICS CALCULATOR

Design and code a program that calculates statistical measures for a set of data values stored in a file. The program prompts for and accepts the name of the data file and reads each record in the file, calculating the mean and the standard deviation of the data values. The data file contains one floating point value per record. Each record is delimited by a newline.

For a sample containing n values, the key statistical indicators include

- mean
  - $m = (a_1 + a_2 + a_3 + ... + a_n) / n$
- sum of the squares of the values
  - $s = a_1^2 + a_2^2 + a_3^2 + ... + a_n^2$
- standard deviation (d)

  $d^2 = (s / n) - m^2$

The program output might look something like:

```
Statistics Calculator
=====================
Enter the name of the data file : sample.txt

The number of data values read from this file was 39
Their statistical mean is 8.08
Their standard deviation is 2.15
```