## APCSA – Strings & Text Files 70-, 85-, and 100-point Versions

## Text Files Lab

## Problem Description:

Complete a program that processes 3 different data files by writing several methods in the FilesLab class as detailed below.

Be sure to add your 6-line comment at the top of the file and method comments above each method heading. Remember to follow proper programming practices as directed in the style guide. Sample input and output files are shown on the back.

## 70-Point Version

Method processIntegers reads groups of 3 integers from the given file. The method creates an output file containing the smallest and largest integer from each group on separate lines. **Propagate** the FileNotFoundException for the main to catch. Do this in the next 2 methods as well.

* inFile – this parameter is the name of the input file to be read
  + First line – contains an integer, the number of data sets (sets of 3 integers) that follow in the file
  + Each data set – contains 3 integers separated with white space on a single line
* outFile – this parameter is the name of the output file to be created
  + For each data set in the input file, write the smallest and largest integers from the set of 3 on a single line.
  + On each output line, write the smallest integer, 2 spaces, then the largest integer

## 85-Point Version

Method testLines reads in a string to locate and an unknown number of lines of text in which to search. The method creates an output file with only those lines of text that contain the given string on separate lines (**NOT case-sensitive**). Update the output to the console window as shown in the sample execution on the back.

* inFile – this parameter is the name of the input file to be read
  + First line – contains a string to look for in the subsequent lines of text.
  + Subsequent lines – an unknown number of lines of text follow the first line.
* outFile – this parameter is the name of the output file to be created
  + Write each line of text from the input file that contains the string from the first line of the input file.
  + Each line of text is written on a separate line in the output file.

## 100-Point Version

Method extractSubstrings reads in an unknown number of lines, each identifying the substring to extract from a string. The method creates an output file containing the requested substrings on separate lines. The method uses the private helper method getSubstring to extract the substring and ensure that the indexes are valid. If the requested substring has invalid indexes, an error message is written to the file for that substring. A *try-catch* must be used inside the loop that reads the input file to catch the exception thrown by the helper method. See the sample output files on the back.

* inFile – this parameter is the name of the input file to be read
  + Contains an unknown number of lines in the file.
  + Each line contains 2 indexes and a string such that the characters in indexes [index1, index2) are extracted from the remainder of the line; note that index1 is included, index2 is not included, and all spaces after the 2nd index should be *trimmed* away.
* outFile – this parameter is the name of the output file to be created
  + Uses method getSubstring to extract the requested substring. The extractSubstrings method tries to get the substring and write the returned substring between colons (“:”) on a separate line.
  + If an IllegalArgumentException is caught, writes “Error in file *filename*:” followed by the exception’s message on a separate line in the output file.
* getSubstring method – private helper method to extract the requested substring
  + Throws an IllegalArgumentException whenever the given indexes are invalid for the given text string. A string indicating the indexes and text string must be given to the exception at instantiation.
  + Extracts and returns the substring [index1, index2) from the given text when indexes are valid.

## Sample Execution – Console Window

Smallest and largest integers from test1.txt written to output1.txt

Lines containing "the" from test2a.txt written to output2a.txt

Lines containing "Plano West" from test2b.txt written to output2b.txt

Requested substrings from test3a.txt written to output3a.txt

Requested substrings from test3b.txt written to output3b.txt

## 70-Point Version Input/Output Files

|  |  |  |
| --- | --- | --- |
| test1.txt |  | output1.txt |
| 5  14 -9 -15  37 49 30  10 -16 -16  -46 -45 -17  42 42 42 |  | -15 14  30 49  -16 10  -46 -17  42 42 |

## 85-Point Version Input/Output Files

|  |  |  |
| --- | --- | --- |
| test2a.txt |  | output2a.txt |
| **the**  Plano West is rising to the top, to the top  The Wolves are strong and proud;  Stand up and shout out loud;  We love tHe colors, blue, black, and white  Fight! Fight! Fight!  Plano West determined we will be  You will see  THE mighty Wolves will march to victory  Hail to Plano West!  Another great fight song! |  | Plano West is rising to **the** top, to the top  **The** Wolves are strong and proud;  We love **tHe** colors, blue, black, and white  **THE** mighty Wolves will march to victory  Ano**the**r great fight song! |

|  |  |  |
| --- | --- | --- |
| test2b.txt |  | output2b.txt |
| **Plano West**  Plano West is rising to the top, to the top  The Wolves are strong and proud;  Stand up and shout out loud;  We love the colors, blue, black, and white  Fight! Fight! Fight!  PLANO west determined we will be  You will see  The mighty Wolves will march to victory  Hail to pLaNo wEsT!  plaplaplaNo weSTern |  | **Plano West** is rising to the top, to the top  **PLANO west** determined we will be  Hail to **pLaNo wEsT**!  plapla**plaNo weST**ern |

## 100-Point Version Input/Output Files

|  |  |  |
| --- | --- | --- |
| test3a.txt |  | output3a.txt |
| 2 8 Plano West  0 6 Wolves  7 25 Royal blue, black, and white  0 10 Hail to Plano West! |  | :ano We:  :Wolves:  :lue, black, and wh:  :Hail to Pl: |

|  |
| --- |
| test3b.txt |
| -2 5 Plano West  7 7 Wolves  5 5 Plano West  15 10 Royal blue, black, and white  8 -10 Hail to Plano West! |

|  |
| --- |
| output3b.txt |
| Error in file test3b.txt: Invalid indexes [-2, 5) for "Plano West"  Error in file test3b.txt: Invalid indexes [7, 7) for "Wolves"  ::  Error in file test3b.txt: Invalid indexes [15, 10) for "Royal blue, black, and white"  Error in file test3b.txt: Invalid indexes [8, -10) for "Hail to Plano West!" |