COSC190
Assignment 1
Due date: 31st January 2024

Instructions:

1. Place all necessary files in a package called "**assignment1**".
2. Name your files according to the specifications listed below. [Files with incorrect name will not be marked].

BadPass and GetPasswords [10 marks]

- Create a new exception class called **BadPass** that has:
  1. a default (no argument constructor) and
  2. a constructor that takes a String message.
- Write a program **GetPasswords** that asks for user passwords which are between 8 and 12 characters in length, and contains at least two special symbols from the set "%&^!@". If these conditions are not met, then your program will throw a **BadPass** exception.

Count Words [15 marks]

Write a program **CountWords** that for every command line argument that is given to that corresponds to a file, will go through, and count the number of characters, words, and lines in the file. If any one of the arguments does not correspond to an existing file, then that should be indicated on the output as well.

I went through and created an executable jar file in the artifacts directory for this file and then ran it using an argument to the Files Directory under the project. This is the output that was produced. *[There is no compulsion to create a jar file]*

```
PS C:\Users\barrie> cd C:\Java\COSC190\out\artifacts\countWords\
PS C:\Java\COSC190\out\artifacts\countWords> java -jar .\COSC190.jar C:\Java\COSC190\Files\*
Report for file CarInfo.csv
Line Count 501
Word Count 2630
Character Count 18362
Report for file first.txt
Line Count 1
Word Count 3
Character Count 14
Report for file student.txt
Line Count 3
Word Count 6
Character Count 23
PS C:\Java\COSC190\out\artifacts\countWords>
```

COSC190
Assignment 1
Due date: 31st January 2024

Show Usage [15 marks]

Write a program called **ShowUsage** that when given a command line argument (that corresponds to a file) will create a usage list that shows the number of occurrences of any word in the file that is longer than 3 characters. This list should be sorted in descending order with the most common words used first. You might want to create an extra class named Word to help you with this.

For this example, I have downloaded a file called charge.txt which contains the text from Tennysons "Charge of the Light Brigade" and ran my program against this file (this file is included in your assignment). The first few lines of the output look like this:

```
hundred:7
them:7
they:7
rode:6
cannon:6
death:5
into:4
left:4
half:3
```

COSC190
Assignment 1
Due date: 31st January 2024

Encrypt [15 marks]

- One of the simplest encryption techniques is to substitute each character in the message with a corresponding character (position with relations to '**a**') from a substitution keypad that re-orders the alphabet in some fashion.
- You should create a program called "**Encyrpt.java**" which will contain the following specified methods:

public static String genKeyPad()

This method will generate a completely random ordering of all lower-case characters (different each time you run it) as a string. For example, calling this method might return: "gphijonmqdefrxyzklstuvabcw"

public static String encryptLine(String sPlainText, String sKeypad)

This method will take a line of text and encrypt it using a randomly generated keypad as described above i.e., substituting each lower-case character with the corresponding character from the keypad.

main method

The main method of this program should take in the name of file to be encrypted as a command line argument. If and if only the given text file has a "**txt**" suffix, then proceed to encrypt the entire file using the encrypt method for each of the lines in the file. Numbers and punctuation should be written through to the encrypted file. All upper-case letters are converted to lower case before being encrypted. The keypad string will be recorded in a file called "**enc.txt**". You are going to be overwriting the original file will the encrypted file. If the indicated file does not exist or is not a .txt file, then the program should exit with an appropriate error message and program exit status code of 1.

Decrypt [10 marks]

Write a program **Decrypt.java** which takes in the name of an encrypted file as a command line argument and the name of a file containing an appropriate keypad string. This program will proceed to decrypt that file using the given keypad. This program is the reverse of the previous program and if you ran these 2 programs in sequence you should wind up with the original file (with all uppercase letters being replaced by lower case letters). As before if the given encrypted file or the keypad file does not exist then exit with an error code of 1 and appropriate error message.

COSC190
Assignment 1
Due date: 31st January 2024

Country [05 marks]

- Provided to you is a CSV file called Country.csv which lists the following for each country in the world.
    a. Abbreviation
    b. Latitude
    c. Longitude
    d. Name
- Create a Class file "**Country.java**" for storing information pertaining to the country information as specified above. The class should contain fields for the above information and appropriate getter methods. It should also contain a constructor that takes in an Array of Strings

Country Info [30 marks]

Create a Java file called "**CountryInfo.java**" and include the following methods in the file:

public static ArrayList<Country> loadInfo(String sFileName)

[05 marks]

This method should take in the name of the file that stores our country information (Country.csv) and return an ArrayList of Country objects as defined above.

public static void writeCountryInfo(ArrayList<Country> obList, String sFile)

[05 marks]

This method will go though and write out all the Country objects in the given list to the file whose name is given as the 2nd argument to this method. You should write these values out utilizing Object Serialization.

public static void writeWesternCountries(ArrayList<Country> obList, String sFile)

[05 marks]

This method will write out to the specified file all countries in the given list that are Western Countries (countries with a negative longitude). Write these records out using Object Serialization.

public static void writeRAFCountry(ArrayList<Country> obList, String sFileName)

[10 marks]

This method will write out all Country entries in the given ArrayList to the given file in Random Access format. Note that you are going to have to modify your Country class to achieve this as all Country records should be the same size.

public static Country getRAFRec(String sFile, int nPosition)

[05 marks]

This method will return the Country record in the nPosition record in the given file (which is assumed to be in Random Access format). Return null if the nPosition refers to a record that is out of scope for the given file.

COSC190
Assignment 1
Due date: 31st January 2024

Payroll [10 marks]

A sample file exists at the URL liveexample.pearsoncmg.com/data/Salary.txt that on each line indicates a Faculty member's first name, last name, position (i.e., associate professor), and annual salary. Write a Java program **Payroll.java** that reads this file in and determines the high, low, and average salary for each of the following positions assistant professor, associate professor, full professor, and faculty. In addition, provide counts that indicate the number of people in each of those positions.

Sort Question [20 marks]

A variation of merge sort involves calling a bubble or insertion sort when we reduce to an array size that is small – say less than 50. Rewrite the merge sort so that if the array partition we are looking at has a size less than 50, call insertion sort on that partition. Then using System.nanotime for a rough estimate and using random Integer arrays of Various Sizes (N) indicate how efficient the two sorts are using the following table to record your results. Also include estimates for the time it would take to do this using a bubble sort and an insertion sort.

Include all the necessary methods to complete this task in a class called ***SortQuestion***

N	Merge Sort	Modified Merge Sort	Bubble Sort	Insertion Sort
1000				
10000				
100000				
1000000				
10000000				

GenericQ [15 marks]

Create a class GenericQ and implement the following methods:

Get Unique

Write a static generic method called **getUnique** that will take in an array of Type T and return another array of T that consists of unique values from the provided list.

Get Average

Write a static generic method called **getAverage** that will take in a list of a subclass of Number and return the average for that list as a double value.

Get Largest

Write a generic method **getLargest** that will take in an Array of Type T (where T is comparable) and return the largest element in that array.