Estructuras de Datos 2021-1
Práctica 2: Anagramas

M. en C. Carlos Zerón Martínez
zeronmc@gmail.com

Emmanuel Cruz Hernández
emmanuel_cruzh@ciencias.unam.mx

José Antonio Vilchis Salazar
grand_paladin@ciencias.unam.mx

Sara Doris Montes Incin
isara22@ciencias.unam.mx

Fecha de entrega: 27 de octubre de 2020
Hora de entrega: 23:59 hrs

1. Introducción

Un anagrama es una palabra *p* que resulta de la transposición de letras de una palabra *p'*. Todos los anagramas de una palabra *p* corresponde a todas las permutaciones que se pueden hacer a partir de las letras de la palabra original. Al conjunto de permutaciones de una palabra las denominamos *anagramming* de una palabra p.

Por ejemplo, la *anagramming* de la palabra *cat* es el siguiente:

- cat
- cta
- atc
- act
- tca
- tac

La cantidad de elementos que tiene el *anagramming* de una palabra, es la longitud de de la palabra aplicado a la operación factorial. Por ejemplo, la palabra *cat* es de longitud 3, por lo que su *anagramming* asociada es de longitud 3!=6.

2. Actividad

Crea una clase llamada *AnagramSolver*. Considera los siguientes atributos de la clase:

- Un entero llamado *size* de tipo int que representa la longitud de la cadena a obtener el *anagramming*.
- Un arreglo de *char*'s de longitud *size*, llamado *word*.

NOTA: Puedes agregar más atributos si lo consideras necesario.

2.1. Actividad 1 *(4 puntos)*

Implementa un método llamado *rotate(int)* con la siguiente firma:

private static void rotate(int rotator){...}

Este método rota las últimas *rotator* posiciones del arreglo *word* a la izquierda. Por ejemplo, sí tenemos el arreglo word={'h', 'o', 'l', 'a'} y *r=3*, se deben rotar a la izquierda los últimos *r* caracteres. El resultado de rotar los últimos *r=3* caracteres del arreglo *word* es word={'h', 'l', 'a', 'o'}.

2.2. Actividad 2 *(0.5 puntos)*

Implementa un método llamado *displayWord()* con la siguiente firma:

private static void displayWord(){...}

Este método imprime los elementos del arreglo *word*, de tal forma que los caracteres contenidos en el arreglo formen una palabra.

2.3. Actividad 3 *(5 puntos)*

Implementa un método llamado *doAnagram(newSize)* con la siguiente firma:

public static void doAnagram(int newSize){...}

Este algoritmo recursivo consta de los siguientes pasos:

- Si newSize=1, entonces termina la ejecución del método. Este es el caso base.
- Hacer newSize iteraciones. Por cada iteración se realizan las siguientes operaciones:
  - Mandar a llamar recursivamente el método *doAnagram* con parámetro *newSize-1*.
  - Si newSize=2, mandar a llamar el método *displayWord*.
  - Mandar a llamar el método *rotate* con parámetro *newSize*.

2.4. Actividad 4 *(0.5 puntos)*

Solicita al usuario que ingrese una palabra en consola. Muestra el *anagramming* de la palabra ingresada por el usuario y finaliza el programa.

3. Reglas Importantes

- Cumple con los lineamientos de entrega.
- Tu programa debe ser robusto.
- Todos los archivos deberán contener nombre y número de cuenta.
- Tu código debe estar comentado. Esto abarca clases, interfaces, atributos, métodos, etc.
- Para cada clase e interfaz solicitada, crea un nuevo archivo.
- No se permite el uso de clases externas ni bibliotecas externas a excepción de Scanner.
- Utiliza correctamente las convenciones para nombrar variables, constantes, clases y métodos.