Práctica Calificada 3

Curso: CC211 - A1 **Ciclo: 2019.1**

Debe enviar UN SOLO ARCHIVO de nombre `Test3.java`, no más; SE REVISARÁ SOLAMENTE EL ARCHIVO `Test3.java`.

1. (4 ptos.) Cree una clase llamada `Rational` para realizar operaciones aritméticas con fracciones. Use variables enteras para representar las variables de instancia `private` de la clase: el `numerator` y el `denominator`. Proporcione un constructor que permita que un objeto de esta clase se inicialice al ser declarado. El constructor debe almacenar la fracción en forma reducida. La fracción `2/4` es quivalente a `1/2` y debe guardarse en el objeto como 1 en el `numerator` y 2 en el `denominator`. En el caso de pasar al constructor una fracción con denominador cero se debe lanzar una excepción del tipo `IllegalArgumentException` con un mensaje explicando la causa. Proporcione un constructor sin argumentos con valores predeterminados, en caso de que no se proporcionen inicializadores. Proporcione métodos `public` que realicen cada una de las siguientes operaciones:

   a) (1 pto.) Establecer los valores de los atributos de un `Rational` a través de métodos `setNumerator` y `setDenominator`.

   b) (1 pto.) Obtener los valores de los atributos de un `Rational` a través de métodos `getNumerator` y `getDenominator`.

   c) (1 pto.) Sumar dos números `Rational`: el resultado de la suma debe almacenarse en forma reducida. Implemente esto como un método `static`.

   d) (1 pto.) Restar dos números `Rational`: el resultado de la resta debe almacenarse en forma reducida. Implemente esto como un método `static`.

   e) (1 pto.) Multiplicar dos números `Rational`: el resultado de la multiplicación debe almacenarse en forma reducida. Implemente esto como un método `static`.

   f) (1 pto.) Dividir dos números `Rational`: el resultado de la división debe almacenarse en forma reducida. Implemente esto como un método `static`.

   g) (1 pto.) Devolver una representación `String` de un número `Rational` en la forma `a/b` en donde `a` es el `numerator` y `b` es el `denominator` a través del método `fractionalRepresentation`.

   h) (1 pto.) Devolver una representación `String` de un número `Rational` en formato de punto flotante. Proporcione capacidades de formato, que permitan al usuario de la clase especificar el número de dígitos de precisión a la derecha del punto decimal. Llame dicho método: `decimalRepresentation`.

i) (1 pto.) Sobreescriba el método `toString` en el que retorne la representación fraccional y decimal de un objeto `Rational`.

j) (2 ptos.) Cree objetos `Rational`:

1) `r1: 4/-6`
2) `r2: -3/15`
3) `r3: 0/-2`
4) `r4: r1 + r2`
5) `r5: r1 - r2`
6) `r6: r1*r2`
7) `r7: r1/r3`

en la clase `Test3` y muestre la representación en cadena (método `toString`) de dichos objetos.

2. (5 ptos.) Cree una clase llamada `EnteroEnorme` que utilice un arreglo de 40 elementos de dígitos, para guardar enteros de hasta 40 dígitos de longitud cada uno. Proporcione los métodos `parse`, `toString`, `sumar` y `restar`. El método `parse` debe recibir un `String`, extraer cada dígito mediante el método `charAt` y colocar el equivalente entero de cada dígito en el arreglo de enteros. Para comparar objetos `EnteroEnorme`, proporcione los siguientes métodos: `esIgualA`, `noEsIgualA`, `esMayorQue`, `esMenorQue`, `esMayorOIgualA`, y `esMenorOIgualA`. Cada uno de estos métodos deberá ser un método predicado que devuelva `true` si la relación se aplica entre los dos objetos `EnteroEnorme`, y `false` si no se aplica. Proporcione un método predicado llamado `esCero`. [Nota: los valores `boolean` primitivos pueden imprimirse como la palabra "true" o la palabra "false", con el especificador de formato `%b`].

29 de abril de 2019

`Print only when necessary`.