Región de Murcia
Consejería de Educación
Ciencia e Investigación

UNIÓN EUROPEA
Fondo Social Europeo

IES José Planes

C/ Maestro Pedro Abadía
Tlfno: 968 83 46 90
Fax: 968 83 46 07
30100 Espinardo Murcia

PROGRAMACIÓN

1º DAW **2ª Evaluación** **22 -Marzo-2021**

NOMBRE:_____

Diseña las clases **Fraccion, Ejercicios** y **Ppal** de carácter público, siendo la clase Ppal la que contiene el método main.

1. Clase **Fraccion**:
   1. Propiedades privadas de tipo entero: num y den ***0.1 ptos***
   2. Métodos:
      1. (*Ya implementado*) **reducir** tal que simplifica la fracción actual.
      2. **Constructor** patrón, al que se le da un valor inicial de numerador y denominador. En el caso de que el denominador sea 0, se enviará una excepción del tipo IllegalArgumentException con el mensaje "Error, división por cero". Se reducirá la fracción. ***0.5 ptos***
      3. **Constructor** sobrecargado, sin datos de entrada. El numerador y denominador valen 1 (se pide que llame al constructor patrón) ***0.2 ptos***
      4. (Ya implementado) **Constructor** sobrecargado con una Fraccion considerada como dato de entrada, de forma que la fracción se va a inicializar con los valores de la Fracción dada. (Llama al constructor patrón)
      5. **Constructor** sobrecargado con 2 datos de entrada, un valor entero y una Fraccion. De ahí, se obtendrá una fracción como el ejemplo: (Obligatorio: es necesario llamar al constructor patrón) ***0.3 ptos***

$$3\ ¼ = (3*4)/\ 4 + 1/4 = 12/4 + ¼ = 13/4$$

      6. (Ya implementado) **Constructor** sobrecargado, con 3 datos de entrada, un número entero y los dos enteros que conformarán una fracción.
      7. (Ya realizados) **Getter**
      8. **Setter**. Se desea implementar el método que permite cambiar el denominador de la Fracción actual (se debe considerar el caso de que el denominador sea 0). ***0.3 ptos***
      9. (Ya implementado) **toString**, devuelve una cadena con formato: [num/den]
      10. **sumaFracciones** -método patrón-, tal que dada un numerador y un denominador de tipo entero, sume la fracción actual con esos numerador y denominador considerados como entrada, devolviendo una nueva Fraccion. ***0.3 ptos***

$$n1/d1 + n2/d2 = (n1*d2 + d1*n2)/d1*d2$$

      11. **sumaFracciones**, tal que dada una Fraccion considerada como dato de entrada, devuelva la suma de la fraccion actual con la de entrada, devolviendo una nueva Fraccion. Se pide llamar al método sumaFracciones patrón. ***0.2 ptos***
      12. **restaFracciones** patrón, tal que dada una Fraccion considerada como dato de entrada, devuelva una nueva Fraccion siendo la suma de la fracción actual con la de entrada. ***0.3 ptos***

$$n1/d1 - n2/d2 = (n1*d2 - d1*n2)/d1*d2$$

      13. **restaFracciones**, tal que dado un numerador y denominador de tipo entero, llame al método patrón restaFracciones y para que devuelva una Fraccion resultado de la suma de la fracción actual y la generada por los datos de entrada (numerador y denominador). ***0.2 ptos***
      14. (Ya implementado)**productoFracciones**, con dato de entrada una Fraccion devolviendo

una nueva Fraccion producto de la actual con la de entrada.

15. (Ya implementado) **productoFracciones**, con datos de entrada un numerador y un denominador, devolviendo una nueva Fraccion producto de la actual con la generada por los datos de entrada.
16. (Ya implementado)**cocienteFracciones**, con dato de entrada una Fraccion devolviendo una nueva Fraccion cociente de la actual con la de entrada.
    (Ya implementado)**cocienteFracciones**, con datos de entrada un numerador y un denominador, devolviendo una nueva Fraccion cociente de la actual con la generada por los datos de entrada.

2. Clase **Ejercicio**:
    1. Propiedades privadas: profesora de tipo String, n de tipo entero y un ArrayList de tipo Fraccion, con nombre fracciones. * Importante, la profesora puede ser una de las siguientes: “ISABEL”, “PIEDAD”, “PAQUI”, “MAYCA”, “VICKY” ***0.5 ptos***
    2. Métodos:
        1. **Constructor** con los datos de entrada profesora y número de ejercicio. Se inicializa el ArrayList en vacío. Se lanzará una excepción del tipo IllegalArgumentException si el valor profesora no es correcto.***0.5 ptos***
        2. Se desea implementar el método que devuelva el valor del número de ejercicio.***0.1 ptos***
        3. Se desea implementar el método que devuelva el array de String que contiene los nombres de las profesoras.***0.3 ptos***
        4. (Ya implementado) **Setter**
        5. **addFraccion**, tal que dada una Fraccion, se añada al ArrayList la Fraccion, siempre y cuando, no esté ya en el ArrayList fracciones.***0.4 ptos***
        6. **toString**, devuelve: el número de ejercicio y el mensaje: “sin ejercicios” si no hay fracciones o el grupo de fracciones del ArrayList del ejercicio actual (se pide hacerlo de 2 formas: usando foreach y usando for)***0.7 ptos***

3. Clase **Ppal**:
    1. En el método main() //Se tendrá en cuenta la captura de las posibles Excepciones que puedan ocurrir, generando en su lugar la Fraccion por defecto (1/1):
        1. Crea la Fraccion f con los valores (3,5). ***0.1 ptos***
        2. Crea la Fraccion g con los valores (8,0) ***0.4 ptos***
        3. Crea la Fraccion h con los valores por defecto.***0.1 ptos***
        4. Crea la Fraccion i con los valores de la fraccion f.***0.1 ptos***
        5. Crea la Fraccion j con los valores de 3 y la fraccion f.***0.1 ptos***
        6. Crea la Fraccion k con los valores 4, 5 y 2.***0.1 ptos***
        7. Crea el Ejercicio ej1, con los valores “PIEDAD”, número 1.***0.1 ptos***
        8. Añade las fracciones f,g,h al Ejercicio ej1. ***0.2 ptos***
        9. Añade la fracción resultante de la suma de las fracciones f con la fracción resultado del producto de k con i.***0.4 ptos***
        10. Muestra el Ejercicio ej1.***0.1 ptos***
        11. Crea el Ejercicio ej2, con los valores “ISABEL”, el número 2.***0.1 ptos***
        12. Añade las fracciones i,j y k al Ejercicio ej2. ***0.2 ptos***
        13. Comprueba si en el Ejercicio ej2 hay alguna fracción cuyo numerador sea mayor de 20, si es así, indica cuántas fracciones hay. ***0.5 ptos***
        14. Se desea crear un array estático v que guarde los valores reales de las fracciones del Ejercicio ej1. ***0.4 ptos***
        15. Muestra el array v ***0.2 ptos***
        16. Muestra el array v de forma que los valores aparezcan tan solo en formato entero (sin parte decimal) ***0.3 ptos***
        17. Se desea saber si existe algún elemento en el array v cuyo valor sea 4.0 (Tan solo se quiere saber si al menos, hay uno) ***0.4 ptos***
        18. Se desea saber cual es el mayor valor del array v y las veces que se repite. ***0.4 ptos***
        19. Se desea cuál es el menor valor del array v y qué posición ocupa.***0.4 ptos***
        20. Se desea mostrar de los Ejercicios ej1 y ej2, tan solo el nombre corto de la profesora y el número de ejercicio. Para obtener el nombre corto de la profesora, hay extraer las 3 primeras letras del nombre de la profesora y después ‘.’. ***0.5 ptos***