***Conociendo el Proyecto***

1. **El directorio contiene lo siguiente:**
   * **Un archivo README, que contiene la información del Proyecto.**
   * **Package.bluej, que es el ejecutable principal del Proyecto en BlueJ.**
   * **Por cada una de las clases definidas en el Proyecto contiene lo siguiente:**
     + **Un archivo con extensión .class, que es el archive compilado de la clase.**
     + **Un archive con extensión .ctxt, que contiene información adicional acerca de la clase, como la documentación y la estructura de los comentarios.**
     + **Un archive con extensión .java, que es el archive Fuente de la clase.**
2. * **El Proyecto tiene 3 clases: Calmatfra, Matriz y Fraccionario.**
     + **Calmatfra utiliza Matriz.**
     + **Matriz utiliza Fraccionario.**
   * **La clase principal es Calmatfra, ya que es la clase que implementa las demás, por tanto, es la clase que debe instanciarse primero.**
   * **La clase diferente tiene el nombre FraccionarioTest, su propósito es realizar pruebas sobre la clase Fraccionario, para verificar que esta cumple con todos sus requisitos, y realiza su trabajo de manera correcta.**
3. * **Calmatfra: No tiene documentación del encabezado, la documentación de los métodos está muy incompleto.**
   * **Matriz:** En la documentación del encabezado solo se encuentra en autor, y en la documentación de los métodos, aunque incompleta, se encuentran ciertas descripciones en algunos métodos.
   * **Fraccionario:** La documentación de esta clase, tanto del encabezado como de los métodos se encuentra complete.
4. * **Calmatfra: La clase tiene definidos los métodos que la componen, sin embargo, tiene en cada uno de ellos el Código mínimo para que la clase compile.**
   * **Matriz:** **La clase tiene definidos los métodos que la componen, sin embargo, tiene en cada uno de ellos el Código mínimo para que la clase compile.**
   * **Fraccionario:** **La clase tiene definidos los métodos que la componen, sin embargo, tiene en cada uno de ellos el Código mínimo para que la clase compile.**

***Ingeniería reversa***



**Los contenedores que se utilizan en el Proyecto son dos: HashMap y Array.**

**Las diferencias que hay entre ellos, es que HashMap almacena una pareja llave-valor, en una estructura de datos tipo HashTable, en donde además se permiten llaves nulas; el Array por otro lado, requiere saber su tamaño al momento de su inicialización, y almacena sus datos de manera secuencial en memoria.**

***Conociendo pruebas en BlueJ***



**¿Cuáles etiquetas tiene?**

**Únicamente se encontró la etiqueta @test.**

**¿Cuantos métodos tiene?**

**La clase tiene en total 16 métodos.**

**¿Cuantos métodos son de prueba?**

**Todos los métodos son métodos de prueba.**

**¿cómo los reconoce?**

**Los métodos de prueba se reconocen por la etiqueta @test.**



**¿Cuantos tests se ejecutan?**

**Se ejecutaron los 16 tests de la clase FraccionarioTest**

**¿Cuantos pasan las pruebas? ¿por qué?**

**Únicamente 3 test pasaron las pruebas. Esto se debe a que los test que pasaron, prueban funcionalidades básicas del Sistema que ya venían implementadas por defecto en el Proyecto.**



**@test es una etiqueta perteneciente al framework Junit, y es la encargada de decirle al compilador que el método ‘public void’ bajo ella puede ser ejecutado como un caso de prueba. Si el método no arroja ninguna excepción, Junit asumirá que el caso de prueba fue exitoso.**



**assertTrue: Toma como argumento una condición booleana, y se asegura de que esta sea verdadera, si no es así, arroja un AssertionError.**

**assertFalse: Toma como argumento una condición booleana, y se asegura de que esta sea falsa, si no es así, arroja un AssertionError.**

**assertEquals:** su argumento son dos variables del mismo tipo, y esta función se encarga de verificar si esas dos variables tienen el mismo valor, para números de punto flotantes (float, double), se puede proporcionar un delta, es decir, el error máximo que puede ocurrir entre los dos valores.

**assertNull:** Toma como argumento un objeto, cuyo valore debe ser null, si no es así, arroja un AssertionError.

**fail: Falla un test, con un mensaje opcional.**



**La diferencia entre un fallo y un error en Junit, es que el primero significa que el código no cumple con los criterios de la prueba de aceptación, es decir las aserciones definidas en las pruebas no pasan en su totalidad, el segundo se da cuando ocurre un error en el tiempo de ejecución, como por ejemplo dividir entre cero o acceder a una variable nula.**

***Practicando pruebas en BlueJ***



**Debido a que un fraccionario se compone de dos números de tipo entero, se decidió usar dos variables de tipo entero representando el numerador y el denominador respectivamente.**



**Con el fin de pasar todas las pruebas definidas para la clase Fraccionario, se necesitaron implementar los siguientes métodos:**

* **mcd**
* **Constructor de Fraccionario para dos enteros**
* **Constructor de Fraccionario para un entero**
* **numerador**
* **denominador**
* **toString**
* **equals**

**También se creó un nuevo método privado para simplificar el fraccionario.**

***Desarrollando***

****

****

****

***Retrospectiva***

1. **20 horas.**
2. **Falta un ciclo por implementar (el de generar dos nuevas funcionalidades), no se terminó debido al factor tiempo, ya que se necesitó disponer tiempo para otras actividades académicas.**
3. **La práctica más útil fue la de unit testing, ya que poniéndola en práctica se pudieron hacer pruebas relevantes que mejoraran el proceso de desarrollo.**
4. **Aprender el proceso de desarrollo orientado a pruebas, a medida que se iba avanzando en el laboratorio, se notó una mejoría en la habilidad para plantear casos de prueba relevantes, y un modelamiento adecuado.**
5. **Aprender a definir pruebas relevantes y modelar la situación antes de codificar la solución, este proceso requirió tiempo de planeación de como iba a ser la ejecución del programa y los posibles casos en los que podía salir mal.**
6. **.**