Встроенные функциональные интерфейсы

1)
Создайте класс Fraction (дробь) с полями:

```
private int flag; //числитель
private int dig;  //знаменатель
```

Добавьте конструктор по умолчанию и со всеми параметрами.
Реализуйте в классе лямбда-выражения вместо методов.
Используйте для лямбда выражений функциональные интерфейсы:

- Supplier<Integer>, IntSuplier – геттеры
- Consumer<Integer>, IntConsumer – сеттеры
- Supplier<String> – преобразование дроби в строку "1/2"
- Supplier<Double> – преобразование дроби в десятичную 1/2 –> 0.5
- Predicate<Fraction> – сравнение этой дроби с другой
- UnaryOperator<Fraction> – умножение этой дроби на другую
- UnaryOperator<Fraction> – деление этой дроби на другую

Реализуйте статические лямбда-выражения с помощью функциональных интерфейсов:

- Function<Fraction, Double> – преобразование дроби в десятичную 1/2 -> 0.5
- Function<String, Fraction> преобразовать строку в дробь. Строка должна иметь вид: "1/2". При конвертации использовать метод split("/") для строки.
- BiPredicate<Fraction, Fraction> – сравнение двух дробей
- BinaryOperator<Fraction> – умножение двух дробей
- BinaryOperator<Fraction> – деление двух дробей

Примеры:

```
public final Consumer<Integer> setFlag = p -> flag = p;

public final UnaryOperator<Fraction> multyTo = p
-> new Fraction(flag * p.flag, dig * p.dig);

public static final Function<Fraction, Double> decimal = p
```

```
-> (double) p.flag / p.dig;

//Вызов в main
f1.setFlag.accept(1);
```

Функциональные интерфейсы:
https://metanit.com/java/tutorial/9.3.php
https://javanerd.ru/%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D1%8B-java/%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5-%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D1%8B-%D0%B2-java-8/
