**Julio Pochet Edmead**

**Module 10 Discussion Post – Cloneable Interface & Rational Class**

This week, I explored the **‘Cloneable’** interface and the ‘Rational’ class—both really useful once you understand what they’re for.

The **‘Cloneable’ interface** lets you make a copy of an object using the ‘clone()’ method. It’s helpful when you want to work with a duplicate without changing the original. You just implement `Cloneable` and override ‘clone()’:

```java

class Book implements Cloneable {

String title;

Book(String title) { this.title = title; }

public Object clone() throws CloneNotSupportedException {

return super.clone();

}

}

```

This gives you a shallow copy, which works fine unless the object has other objects inside it—then you’d need to handle deep cloning.

The **‘Rational’ class** is great for exact math with fractions. Instead of using ‘double’, which can be imprecise (like 1/3 showing up as 0.333...), Rational keeps the real fraction:

```java

Rational r1 = new Rational(1, 2);

Rational r2 = new Rational(1, 3);

Rational sum = r1.add(r2); // Exact value

```

This makes it perfect for cases where accuracy matters, like math apps or calculations involving symbolic values.

Both features help you write cleaner, more reliable code depending on the situation.