Operator-Overloading-3

Post-increment (f1++;)

→ How to differentiate b/w Pre & Post increment both has "++"?

⇒ operator++() → Pre operator++(int) → Post

→ basic of Post increment (i++)

```
int a = 5;
int b = a++;
Cout << b << " " << a << endl;
```

output :- 5 6

→ In our fraction ——

Fraction f4 = f1++;

according our output,

$f_4 = f_1$ & $f_1 = f_1 + 1$

i.e, f_4 [N=10, D=2] f_1 [N=6, D=1]

& we need to Return (N=10, D=2)

→ basis Code →

```
Fraction operator ++(int) {
    Fraction fNew(Numerator, Denominator);
    nume_ = numer_ + Denominator;
    SimPlify();
    return fNew;
}
```

SimPlify(); → fNew.SimPlify();

"This works fine"

— Lets try nesting here as before —

```
int i = 5;
(i++) ++j;
cout << i;
```

⟶ x ⟶ Error, "Post-increment has no nesting"

So, on fraction also there is no more nesting Possible.

⟹ Now, Lets try overload `+=` operator ——

```
int i=5, j=3;
i+=j;
```

⟶ $i = i+j$

$i = 8$

also Nesting is Possible on this ⟶ $(i+=j)+=j$

⟶ in our fraction class —

$f_1 += f_2$ ⟶ $f_1 = \underbrace{f_1 + f_2}$

this is what our 'add' operator does, Lets use it too.

one think to also keep in mind, we are having both (f_1 & f_2) in it, means it is on binary operator [Not Unary].

⟶ Visit the Code in the file, Comments will make Everything clear.