Prefer Immutability

What is immutability?

- Once created, can't be changed.
- Instead of changing, have to make a copy.

Advantages of Immutability

- Immutability makes it easier to parallelize program as there are no conflicts among objects.
- Immutable objects are thread safe.
- Have better control over side-effects and unexpected data changes.

Bad Practice

```
class Student
{
    var students = mutableListOf("Aarthi", "Aadhi", "Bala", "Barathi", "Divya", "Harini")
    fun selectFirstBatchStudent()
    {
        students = students.subList( 0, ( students.size ) / 2 )
        println(students)
    }
    fun selectSecondBatchStudent()
    {
        students = students.subList(( students.size ) / 2 , students.size )
    }
}
fun main() {
    val student = Student()
    student.selectFirstBatchStudent()
    student.selectSecondBatchStudent()
}
```

The students list here is mutable, on calling the selectFirstBatchStudent() function the list students is modified. And that will affect the result of the selectSecondBatchStudent() function.

Good Practice

```
class Student
{
    val students = listOf("Aarthi", "Aadhi", "Bala", "Barathi", "Divya", "Harini")
    fun selectFirstBatchStudent() {
        val firstBatch = students.subList( 0, ( students.size ) / 2 )
        println(firstBatch)
    }
    fun selectSecondBatchStudent() {
        val lastBatch = students.subList(( students.size ) / 2 , students.size )
        println(lastBatch)
    }
}
fun main() {
    val student = Student()
    student.selectFirstBatchStudent()
    student.selectSecondBatchStudent()
}
```

By making the list immutable we can avoid the side effects.