“Primitives are passed by value, i.e., a copy of the primitive itself is passed. Whereas for objects, the copy of the reference is passed, not the object itself (Stack Overflow).” In this program, **a1** was declared as a primitive type “int” while **a2** and **a3** are declared as two Num objects. When the value passed to the Modifier, **a1** was passed as a copy of itself, and the program changed the value of this specific copy. But when the value of **a2** and **a3** was passed to the Modifier, the code is actually processing a reference to these two objects. The modifier operates two methods to each **a2** and **a3**; it uses .setValue() to **a2** (passed in as f2) which “permanently” changed its value whereas it creates a new object for **a3**(passed in as f3). But this object is attached to the Modifier class only. Therefore, it just changed the value of f3, not the actual **a3** value since it only passed in the reference.

Reference: <https://stackoverflow.com/questions/8643276/object-vs-primitive>