My algorithm is almost linear with respect to N for M operations where n=m

//note that my union-find algorithm uses path compression and weighted quick union, which, according to lecture slide 150, is almost linear for find and union operations. The only things I added to union are comparisons, and HashMap get and put, which are all O(1) operations, so they don't affect the big-O of union.

SetCompatible:

Perform relevant checks and throw exception if necessary

Otherwise, Union the two Xenos (close to linear- see above)

AreCompatible:

Perform relevant checks and throw exception if necessary

Otherwise, perform a find on each xeno and compare it (a total of 2*close to linear-see above)

SetIncompatible:

Perform relevant checks and throw exception if necessary

Otherwise, perform a find on each Xeno (close to linear)

If either, or both, Canonical elements from find have incompatibles in Hashmap:

Union the other canonical element to this one's incompatible (close to linear)

AreIncompatible:

Perform relevant checks and throw exception if necessary

Otherwise, perform a find on one of the xenos and return whether the HashMap entry of it contains the result of find of the other xeno (close to linear)

As illustrated in the pseudocode of the methods above, all methods contain at most unions and finds - both of which are almost linear, as well as arithmetic operations and gets and puts in a HashMap, which are constant time operations. As such, M operations performed will be O(almost linear) with respect to n.