Questions

Bookmark this page

interfaces

1/1 point (graded)

Consider this Java interface and Java class, which are intended to implement an immutable set data type:

```
      /** Represents an immutable set of elements of type E. */
      interface Set<E> {
        /** make an empty set */
/*A*/     public Set();
          /** @return true if this set contains e as a member */
          public boolean contains(E e);
          /** @return a set which is the union of this and that */
/*B*/     public ArraySet<E> union(Set<E> that);
      }

      /** Implementation of Set<E>. */
      class ArraySet<E> implements Set<E> {
          /** make an empty set */
          public ArraySet() { ... }
          /** @return a set which is the union of this and that */
          public ArraySet<E> union(Set<E> that) { ... }
          /** add e to this set */
          public void add(E e) { ... }
      }
```

Which of the following statements are true about Set<E> and ArraySet<E>? Check all that apply.

- [x] The line labeled /*A*/ is a problem because Java interfaces can't have constructors.
- [ ] The line labeled /*B*/ is a problem because Set mentions ArraySet, but ArraySet also mentions Set, which is circular.
- [x] The line labeled /*B*/ is a problem because it isn't representation-independent.
- [x] ArraySet doesn't correctly implement Set because it's missing the contains() method.
- [ ] ArraySet doesn't correctly implement Set because it includes a method that Set doesn't have.
- [x] ArraySet doesn't correctly implement Set because ArraySet is mutable while Set is immutable.

✔

Explanation

/*A*/: Java interfaces can't have constructors, so there would be a static error on this line.
/*B*/: circular references between classes and interfaces are legal in Java, and sometimes necessary.
/*B*/: returning ArraySet makes all clients of Set aware of, and possibly dependent on, the ArraySet implementation. The union operation should return a Set, not an ArraySet.
ArraySet needs to implement all the methods of Set in order to satisfy Java's requirement for "implements Set". The fact that it doesn't have contains() will cause a static error.
Java does allow classes to have more methods than the interface they're implementing.
ArraySet has a mutator (add), which makes it a mutable class. This violates an important part of the spec for Set, so ArraySet is not a legal implementation of Set. Java won't catch this problem, however. Neither a static error nor a dynamic error will be raised automatically.

Submit Show Answer

Answers are displayed within the problem

Previous Next