Questions

Bookmark this page

bag

1/1 point (graded)

Suppose Bag<E> is a mutable ADT representing what is often called a *multiset*, an unordered collection of objects where an object can occur more than once. It has the following operations:

```
/** make an empty bag */
public Bag<E>()

/** modify this bag by adding an occurence of e, and return this bag */
public Bag<E> add(E e)

/** modify this bag by removing an occurence of e (if any), and return this bag */
public Bag<E> remove(E e)

/** return number of times e occurs in this bag */
public int count(E e)
```

Suppose we run this code:

```
Bag<String> b1 = new Bag<>().add("a").add("b");
Bag<String> b2 = new Bag<>().add("a").add("b");
Bag<String> b3 = b1.remove("b");
Bag<String> b4 = new Bag<>().add("b").add("a"); // swap!
```

Which of the following expressions are true? Check all that apply.

- [x] b1.count("a") == 1
- [ ] b1.count("b") == 1
- [x] b2.count("a") == 1
- [x] b2.count("b") == 1
- [x] b3.count("a") == 1
- [ ] b3.count("b") == 1
- [x] b4.count("a") == 1
- [x] b4.count("b") == 1

✔

Explanation

`b1` and `b3` are aliases for the same Bag, which ends up containing just one occurrence of "a" and none of "b".
`b2` and `b4` are both references to different Bags, each of which has one occurence of "a" and one of "b".

Submit | Show Answer

Answers are displayed within the problem

bag behavior

1/1 point (graded)

If Bag is implemented with behavioral equality, which of the following expressions are true? Check all that apply.

- [ ] b1.equals(b2)
- [x] b1.equals(b3)
- [ ] b1.equals(b4)

- [ ] b2.equals(b3)
- [ ] b2.equals(b4)
- [x] b3.equals(b1)

✔

Explanation
Behavioral equality of mutable ADTs requires two references to be equal if and only if they are aliases for the same object. So `b1` and `b3` must compare equal, but `b2` must not be equal to them, and it must also not be equal to `b4`. Symmetry should also still apply.

Submit

Show Answer

Answers are displayed within the problem

bean bag

1/1 point (graded)

If Bag were part of the Java API, it would probably implement observational equality, counter to the recommendation in the reading.

If Bag implemented observational equality despite the dangers, which of the following expressions are true? Check all that apply.

- [ ] b1.equals(b2)
- [x] b1.equals(b3)
- [ ] b1.equals(b4)
- [ ] b2.equals(b3)
- [x] b2.equals(b4)
- [x] b3.equals(b1)

✔

Explanation
Equality is now defined by the observer operation `count`, so `b1` and `b3` are certainly equal, but `b2` and `b4` are now considered equal as well.
The Java Collections implement observational equality because it is often convenient, but it would be better to implement a *different* operation for observational equality of mutable types.

Submit

Show Answer

Answers are displayed within the problem

Previous Next