ic08 Interfaces & Subtyping

Bag.ts

```
import { BasicBag } from './BasicBag.js';

/**
 * A mutable multiset, also called a bag.
 * In a multiset, elements may occur more than once.
 * For example, { a, a, a, b, b } is a multiset with 3 occurrences of a and 2 occurrences of
b.
 * { a^3, b^2 } is another common way to write it.
 */
export interface Bag<E> {

    //////
    // observers:

    /**
     * Get size of the bag.
     * @returns the number of elements in this bag
     */
    size(): number;

    /**
     * Test for membership.
     * @param elt a possible element
     * @returns true iff this bag contains elt
     */
    contains(elt: E): boolean;


    //////
    // mutators:

    /**
     * Modifies this bag by adding one occurrence of elt to the bag.
     * @param elt element to add
     */
    add(elt: E): void;

    /**
     * Modifies this bag by removing one occurrence of elt, if found.
     * If elt is not found in the bag, has no effect.
     * @param elt element to remove
     */
    remove(elt: E): void;

}

/**
 * @returns a new empty bag
 */
```

```
export function makeBag<E>(): Bag<E> {
    return new BasicBag<E>();
}
```

BasicBag.ts

```
import assert from 'assert';
import { Bag } from './Bag.js';

/**
 * A mutable bag of arbitrary elements of type E.
 * For example, { a, a, a, b, b } is a multiset with 3 occurrences of a and 2 occurrences of
b.
 * { a^3, b^2 } is another common way to write it.
 */
export class BasicBag<E> implements Bag<E> {

    private elements: Array<E> = [];

    // Representation invariant:
    //   true
    // Abstraction function:
    //   AF(elements) = the multiset (A, m) such that A is the set of elements in `elements
`, and
    //                                      m(e) = the number of times an element e occurs
in `elements`
    // Safety from rep exposure:
    //   elements is private
    //   no public method takes or returns an array (the only mutable type used in the rep)

    private checkRep():void {
    }

    /**
     * Make a new empty bag.
     */
    public constructor() {
        this.checkRep();
    }

    /**
     * @inheritDoc
     */
    public size(): number {
        this.checkRep();
        return this.elements.length;
    }

    /**
     * @inheritDoc
     */
    public contains(elt: E): boolean {
        this.checkRep();
        return this.elements.includes(elt);
    }
    /**
     * @inheritDoc
     */
    public add(elt: E): void {
        this.elements.push(elt);
        this.checkRep();
    }

    /**
     * @inheritDoc
     */
    public remove(elt: E): void {
        const i = this.elements.indexOf(elt);
        if (i !== -1) {
            this.elements.splice(i, 1);
        }
        this.checkRep();
    }

}
```

spring 2024 course site archive | latest site at mit.edu/6.102 | accessibility

BagTest.ts

```
import assert from 'assert';

import { Bag, makeBag } from '../src/Bag.js';

describe('Bag', function() {

    // Testing strategy
    //
    // For all operations:
    //    partition on bag size: 0, 1, >1
    //
    // For contains, add, remove:
    //    partition on multiplicity of elt: 0, 1, >1


    it('covers size=0, contains elt with multiplicity 0', function() {
        const bag: Bag<string> = makeBag<string>();
        assert.strictEqual(bag.size(), 0);
        assert( ! bag.contains("a"));
    });

    it('covers add with size=0; contains elt with multiplicity 1; size=1; add elt with multi
plicity 0', function() {
        const bag: Bag<string> = makeBag<string>();
        bag.add("b"); // s is now { "b" }
        assert.strictEqual(bag.size(), 1);
        assert(bag.contains("b"));
        assert( ! bag.contains("c"));
    });

    it('covers remove with size > 1; remove elt with multiplicity >1', function() {
        const bag: Bag<string> = makeBag<string>();
        bag.add("d"); // s is now { "d" }
        bag.add("d"); // s is now { "d", "d" }
        bag.remove("d"); // s is now { "d" }
        assert.strictEqual(bag.size(), 1);
        assert(bag.contains("d"));
        assert(! bag.contains("e"));
    });

    // not shown: additional tests to cover the rest of the partitions

});
```