```
Bag (E, C): trait
  % Common bag operators
  includes
    BagBasics,
    DerivedOrders (C, ⊆ for ≤, ⊇ for ≥,
                   ⊂ for <, ⊃ for >)
  introduces
    delete: E, C → C
    {__}: E → C
    __∈ __, __∉ __: E, C → Bool
    size: C → Int
    __∪ __, __-__: C, C → C
  asserts
    ∀ e, e1, e2: E, b, b1, b2: C
      count(e1, delete(e2, b)) ==
        if e1 = e2 then max(0, count(e1, b) - 1)
        else count(e1, b);
      { e } == insert(e, { });
      e ∈ b == count(e, b) > 0;
      e ∉ b == count(e, b) = 0;
      size({ }) == 0;
      size(insert(e, b)) == size(b) + 1;
      count(e, b1 ∪ b2) ==
        count(e, b1) + count(e, b2);
      count(e, b1 - b2) ==
        max(0, count(e, b1) - count(e, b2));
      b1 ⊆ b2 == b1 - b2 = { };
  implies
    AbelianMonoid (∪ for ∘, { } for unit, C for T),
    JoinOp (∪, { } for empty),
    MemberOp ({ } for empty),
    PartialOrder (C, ⊆ for ≤, ⊇ for ≥,
                  ⊂ for <, ⊃ for >)
    ∀ e, e1, e2: E, b, b1, b2: C
      insert(e, b) ≠ { };
      count(e, b) ≥ 0;
      count(e, b) ≤ size(b);
      b1 ⊆ b2 ⇒ count(e, b1) ≤ count(e, b2)
    converts count, ∈, ∉, {__}, ∪, -:C,C→C,
      delete, size, ⊆, ⊇, ⊂, ⊃
```