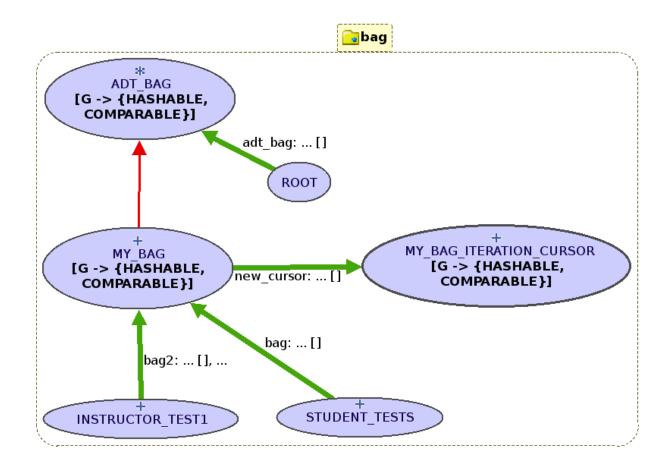
EECS3311 Lab 4 Report

Name: Matthew MacEachern

Prism a/c: mcmaceac

Student number: 213960216

To the best of my knowledge, I have completed this lab in full. I affirm that this lab is solely my own work.



```
ADT_BAG [G -> {HASHABLE, COMPARABLE}]
                                                                                     MY_BAG_ITERATION_CURSOR [G ->
                                                                                         {HASHABLE, COMPARABLE}]
make_empty *
make_from_tupled_array *
  ->a_array: ARRAY[TUPLE[x: G; y: INTEGER]]
    ? a_array.count >=0
    \forall t \in a\_array: t.y >= 0
is_nonnegative: BOOLEAN *
  ->a_array: ARRAY[TUPLE[x: G; y: INTEGER]]
   ! Result = ∀a ∈ a_array: a.y >=0
bag_equal: BOOLEAN *
  ->other: like Current
    ! Result = (other = Current)
total: INTEGER
                                                                                                              new_cursor: ... []
count: INTEGER *
domain: ARRAY[G] *
   ! Result.object_comparison
    ∀j ∈ Result: j ∈ Result
    ∀j ∈ Result: Result[j] <= Result[j+1]
occurrences: INTEGER *
  ->key: G
   ! Result >=0
    key ∈ array -> Result > 0
has: BOOLEAN
  ->a_item: G
   ! Result = (occurrences (a_item) > 0)
is_subset_of: BOOLEAN *
                                                                                     MY_BAG [G -> {HASHABLE, COMPARABLE}]
  ->other: like Current
    ! ∀g ∈ domain: (has(g) -> other.has(g)) and
     occurrences(g) <= other.occurrences(g)
extend *
  ->a_key: G
    a_quantity: INTEGER
    ? a_quantity >= 0
    ! has(a_key) = (old has(a_key) or a_quantity > 0)
    occurrences(a_key) = old (occurrences(a_key)) + a_quantity
add_all *
                                                                                                                             bag2: ...[], ...
                                                                                bag: ... []
  ->other: like Current
remove *
  ->a_key: G
    a_quantity: INTEGER
    ? a_quantity >= 0
   ! has(a_key) = (a_quantity < old occurrences(a_key))
  occurrences(a_key)=(old (occurrences(a_key))-a_quantity).max(0)
remove_all *
  ->other: like Current
                                                                                STUDENT_TESTS
                                                                                                                   INSTRUCTOR_TEST1
number_of: INTEGER
  ->f: PREDICATE[G,INTEGER]
                          ---Invariant--
count = domain.count
\forall x \in domain: occurrences(x) > 0
Current = Current
```