SOEN331: Introduction to Formal Methods for Software Engineering
Assignment 4 on algebraic specifications

Tarek Ait Hamouda (40044119), Abhijit Gupta (40066502),
Ethel Narra Pangan (40061530)

April 7, 2019

Spec: Location;

Sort: Location;

Imports:String, Point

Description: A location contains a point and a description.

- newlocation (String, Point): Creates a new location with the given description and point.
- setdescription (String): Modifies the description of a location, and returns the updated location.
- getdescription: Returns the description of the location.
- setpoint (Point): Modifies the point of the location, and returns the updated location.
- getpoint: Returns the point of the location.

Operations:

newlocation: String $\times$ Point $\rightarrow$ Location;

setdescription: String $\times$ Location $\rightarrow$ Location;

getdescription : Location → String;

setpoint : Point × Location → Location;

getpoint : Location → Point;

Variables:

newDesc, d: String; newPoint, p: Point;

Axioms:

[A1] getdescription(newlocation(d, p)) = d;

[A2] getpoint(newlocation(d, p)) = p;

[A3] setdescription(newDesc, newlocation(d,p)) = newlocation(newDesc, p);

[A4] setpoint(newPoint, newlocation(d,p)) = newlocation(d, newPoint);

Spec: Map (Location);

Sort: Map;

Imports: String, Point, Boolean, Location,ℕ;

Description: A Map ADT contains a collection of locations.

- newmap: Creates a new empty map.
- addlocation (Location): Adds a new location on the map. A location whose description already exists in the map will override the corresponding location with a new point. Returns updated Map.
- deletelocation (String): Deletes the location from the map that corresponds to a given description. Returns updated map
- containsdescription (String): Determines whether the map contains the given description. Returns true if a description is found, and it returns false otherwise.
- containspoint (Point): Determines whether the map contains the given point. Returns true if a description is found, and it returns false otherwise.
- findlocation (String): Returns the point for the location on the map that corresponds to the given description.

- isempty: Determines whether the map is empty of annotations. Returns true if the map contains no annotations and it returns false otherwise.
- clear: Erases all locations from a map.
- size: Returns the number of annotations in the map.

Operations:

newmap: $\rightarrow$ Map;

addlocation: Map $\times$ Location $\rightarrow$ Map;

deletelocation : Map $\times$ String $\rightarrow$ Map;

containsdescription : Map $\times$ String $\rightarrow$ Boolean;

containspoint : Map $\times$ Point $\rightarrow$ Boolean;

findlocation : Map $\times$ String $\rightarrow$ Point;

isempty : Map $\rightarrow$ Boolean;

clear : Map $\rightarrow$ Map;

size : Map $\rightarrow \mathbb{N}$

Variables:

d: String; p, q: Point; loc: Location; map: Map

Axioms:

[A1] isempty(newmap) = true;

[A2] isempty(clear(map)) = true;

[A3] containsdescription(addlocation(map, new), getdescription(loc)) = true;

[A4] containsdescription(map,d) $\rightarrow$ findlocation(addlocation(map, newlocation(d,q)), d)
== q

[A5] size (addLocation(map, newlocation(d,q))) =
if (containsdescription(map, d))
then size(map)
else size(map) + 1

[A6] isempty(deleteLocation(addlocation(newmap, newlocation(d,p)), d)) = true

[A7] findlocation(addlocation(map, newlocation(d,p)), d) = p

[A8] findlocation(newmap, d) = undefined;

[A9] deletelocation(newmap, d) = undefined;

preconditions:

pre : deletelocation(map: Map, d: String) = containsdescription (map, d);

pre : findlocation(map: Map, d: String) = containsdescription (map, d);