Set 10

1. Where is the isValid method specified? Which classes provide an implementation of this method?

Answer: isValid method is specified in Class Grid and Class Bounded provide an implementation of this method

2. Which AbstractGrid methods call the isValid method? Why don't the other methods need to call it?

Answer: getValidAdjacentLocations method call the isValid method. The other methods needn’t to call it because they can get the location list which is valid by the calling the method getValidAdjacentLocation

3. Which methods of the Grid interface are called in the getNeighbors method? Which classes provide implementations of these methods?

Answer: getOccupiedAdjacentLocations method is called in the getNeighbors. The Abstract Class provide implementation of these methods

4. Why must the get method, which returns an object of type E, be used in the getEmptyAdjacentLocations method when this method returns locations, not objects of type E?

Answer: Because the getEmptyAjacentLocation method wants to ensure every location in the ArrayList passed to it is null.

5. What would be the effect of replacing the constant Location.HALF_RIGHT with Location.RIGHT in the two places where it occurs in the getValidAdjacentLocations method?

Answer: Using Location.HALF_RIGHT can Gets the valid locations adjacent to a given location in all eight(north, northeast, east, southeast, south, southwest,west, and northwest).

However, Using Location.Right can only gets the valid locations adjacent to a given location in four(north, east, south, and west)

Set 11

1. What ensures that a grid has at least one valid location?

Answer: If the number of row or column is no bigger than 0, it will throw new IllegalArgumentException.

2. How is the number of columns in the grid determined by the getNumCols method? What assumption about the grid makes this possible?

Answer: occupantArray[0].length;

The assumption is the number of column in every row is the same.

3. What are the requirements for a Location to be valid in a BoundedGrid?

Answer:

(1)0 <= loc.getRow()

(2)loc.getRow() < getNumRows()

(3)0 <= loc.getCol()

(4)loc.getCol() < getNumCols()

In the next four questions, let r = number of rows, c = number of columns, and n = number of occupied locations.

4. What type is returned by the getOccupiedLocations method? What is the time complexity (Big-Oh) for this method?

Answer: return type ArrayList<Location>

The time complexity is O(r*c)

5. What type is returned by the get method? What parameter is needed? What is the time complexity (Big-Oh) for this method?

Answer: get method return type E, it need parameter a Location

The time complexity is O(1)

6. What conditions may cause an exception to be thrown by the put method? What is the time complexity (Big-Oh) for this method?

Answer: When the location is invalid or the object is null will cause an exception .

The time complexity for this method is O(1)

7. What type is returned by the remove method? What happens when an attempt is made to remove an item from an empty location? What is the time complexity (Big-Oh) for this method?

Answer: Remove method return type E. It will throw new IllegalArgumentException when an attempt is made to remove an item from an empty location.

The time complexity for this method is O(1)

8. Based on the answers to questions 4, 5, 6, and 7, would you consider this an efficient implementation? Justify your answer.

Answer:
Overall,yes.The only method that is inefficient is the getOccupiedLocations method, O(r*c).The other methods(put, get, and remove) are O(1).

Set 12

1. Which method must the Location class implement so that an instance of HashMap can be used for the map? What would be required of the Location class if a TreeMap were used instead? Does Location satisfy these requirements?

Answer: The Location class must implement the hashcode and equals methods so that an instance of HashMap can be used for the map.

The Location class must implement the compareTo method so that an instance of TreeMap can be used for the map.

The Location satisfy these requirements

2. Why are the checks for null included in the get, put, and remove methods? Why are no such checks included in the corresponding methods for the BoundedGrid?

Answer:

Because the isValid methods for the UnbooundedGrid always returns true. But the isValis methods can help BoundedGrid to check for null locations.

3. What is the average time complexity (Big-Oh) for the three methods: get, put, and remove? What would it be if a TreeMap were used instead of a HashMap?

Answer: The average time complexity (Big-Oh) for the three methods: get, put, and remove is O(1)

if a TreeMap were used instead of a HashMap, the average time will be O(log n)

4. How would the behavior of this class differ, aside from time complexity, if a TreeMap were used instead of a HashMap?

Answer: getOccupiedLocations method will return the occupants in a different order.

5. Could a map implementation be used for a bounded grid? What advantage, if any, would the two-dimensional array implementation that is used by the BoundedGrid class have over a map implementation?

Answer:
Yes, a map could be used to implement a bounded grid.
The two-dimensional array is better because its average time complexity is less and it uses less memory.

Exercise:

1.Implement the methods specified by the Grid interface using this data structure. Why is this a more timeefficient implementation than BoundedGrid?
Answer: As you don't need to access all the nodes in the grid, you don't need to compare the null grid so it's more time efficient

2. Consider using a HashMap or TreeMap to implement the SparseBoundedGrid. How

could you use the UnboundedGrid class to accomplish this task? Which methods of UnboundedGrid could be used without change:

Answer:
The methods that do not change are: getOccupiedLocations、get、 put、 and remove.

Methods	SparseGridNode	LinkLisr<OccupantInCol>	HashMap	TreeMap
getNeighbors	O(c)	O(c)	O(1)	O(logn)
getEmptyAdjacentLocations	O(c)	O(c)	O(1)	O(logn)
getOccupiedAjacentLocations	O(c)	O(c)	O(1)	O(logn)
getOccupiedLocations	O(r+n)	O(r+n)	O(c)	O(c)
get	O(c)	O(c)	O(1)	O(logn)
put	O(c)	O(c)	O(1)	O(logn)
remove	O(c)	O(c)	O(1)	O(logn)

3.Implement the methods specified by the Grid interface using this data structure. What is the BigOh efficiency of the get method? What is the efficiency of the put method when the row and column index values are within the current array bounds? What is the efficiency when the array needs to be resized?
Answer:
The time complexity of get method is O(1), when the row and column index are within the current array bounds the timecomplexity is O(1), otherwise the complexity is O(n*n), n is the number of the row.