Part5

**Do you Know?**

**Set10**

1. Where is the isValid method specified? Which classes provide an implementation of this method?  
这个特别的方法在grid中，UnboundedGrid类和BoundedGrid类提供了对这个方法的implementation，如下：

|  |
| --- |
| Java //Grid.java boolean isValid(Location loc);  //UnboundedGrid.java public boolean isValid(Location loc) {  return true; }  //BoundedGrid.java public boolean isValid(Location loc) {  return true; } |

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

AbstractGrid中的getValidAdjacentLocations调用了isValid方法；

因为其他的方法都直接或者间接的使用了此方法，所以不需要重复调用isValid（）。

|  |
| --- |
| Java  //AbstractGrid.java if (isValid(neighborLoc))  locs.add(neighborLoc); |

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

方法：getOccpiedAdjacentLocations(),get()

AbstractGrid类提供了对getOccpiedAdjacentLocations()的implements

BoundedGrid 类和UnboundedGrid类提供了对get()的implements

|  |
| --- |
| Java //AbstractGrid.java for (Location neighborLoc : getOccupiedAdjacentLocations(loc))  neighbors.add(get(neighborLoc));  //bstractGrid.java public ArrayList<Location> getOccupiedAdjacentLocations(Location loc) {  ArrayList<Location> locs = new ArrayList<Location>();  for (Location neighborLoc : getValidAdjacentLocations(loc))  {  if (get(neighborLoc) != null)  locs.add(neighborLoc);  }  return locs; }  //BoundedGrid.java  public E get(Location loc)   //UnboundedGrid.java public E get(Location loc) |

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?

由于get方法返回存储在给定位置的网格中的对象时，会返回此对象的引用，如果没有对象，则返回null，故需要返回E类型的对象。

getEmptyAdjacent方法通过使用get方法来判断返回的结果是否为null，也正因如此我们可以得到正确的空区。

|  |
| --- |
| Java //AbstractGrid.java public ArrayList<Location> getEmptyAdjacentLocations(Location loc) {  ArrayList<Location> locs = new ArrayList<Location>();  for (Location neighborLoc : getValidAdjacentLocations(loc))  {  if (get(neighborLoc) == null)  locs.add(neighborLoc);  }  return locs; } |

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?

会使得有效相邻位置的最大数量从 8 个减少到 4 个。只有东，西，南，北。

|  |
| --- |
| Java //AbstractGrid.java int d = Location.NORTH; for (int i = 0; i < Location.FULL\_CIRCLE / Location.HALF\_RIGHT; i++) {  Location neighborLoc = loc.getAdjacentLocation(d);  if (isValid(neighborLoc))  locs.add(neighborLoc);  d = d + Location.HALF\_RIGHT; } |

**Set11**

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

BoundedGrid的构造函数决定的：

|  |
| --- |
| Java //BoundedGrid.java public BoundedGrid(int rows, int cols) {  if (rows <= 0)  throw new IllegalArgumentException("rows <= 0");  if (cols <= 0)  throw new IllegalArgumentException("cols <= 0");  occupantArray = new Object[rows][cols]; } |

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

是由getNumCols()的return值决定的，BoundedGrid至少有一行一列。

|  |
| --- |
| Java //BoundedGrid.java public int getNumCols() {  return occupantArray[0].length; } |

3.What are the requirements for a Location to be valid in a BoundedGrid? In the next four questions, let r = number of rows, c = number of columns, and n = number of occupied locations.

0≤the row of the location < the number of rows

0≤the col of the location < the number of colums

|  |
| --- |
| Java //BoundedGrid.java public boolean isValid(Location loc) {  return 0 <= loc.getRow() && loc.getRow() < getNumRows()  && 0 <= loc.getCol() && loc.getCol() < getNumCols(); } |

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

返回类型：ArrayList<Location>

时间复杂度：O（r\*c）

|  |
| --- |
| Java //BoundedGrid.java public ArrayList<Location> getOccupiedLocations() {  ArrayList<Location> theLocations = new ArrayList<Location>();   // Look at all grid locations.  for (int r = 0; r < getNumRows(); r++)  {  for (int c = 0; c < getNumCols(); c++)  {  // If there's an object at this location, put it in the array.  Location loc = new Location(r, c);  if (get(loc) != null)  theLocations.add(loc);  }  }   return theLocations; } |

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

返回类型：E；

参数类型：Location

时间复杂度：O（1）

|  |
| --- |
| Java //BoundedGrid.java public E get(Location loc) {  if (!isValid(loc))  throw new IllegalArgumentException("Location " + loc  + " is not valid");  return (E) occupantArray[loc.getRow()][loc.getCol()]; // unavoidable warning } |

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

两种情况：

location不是有效的，即isValid(location)==0;

传进来的object为空

时间复杂度：O（1）

|  |
| --- |
| Java //BoundedGrid.java public E put(Location loc, E obj) {  if (!isValid(loc))  throw new IllegalArgumentException("Location " + loc  + " is not valid");  if (obj == null)  throw new NullPointerException("obj == null");   // Add the object to the grid.  E oldOccupant = get(loc);  occupantArray[loc.getRow()][loc.getCol()] = obj;  return oldOccupant; } |

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?

返回类型：E

如果这个位置是isValid的，则不会有错误，否则会出现错误。

时间复杂度：O（1）

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

本小组统一这是一种有效的implementation，因为时间复杂度都较低，最大的也不过为O(r\*c)。

**Set12**

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?

HashMap：Location类实现了hashCode()方法和equals()方法；

TreeMap：Location实现了Compare接口，并且需要实现compareTo()，同时TreeMap需要key是可以进行比较的。

|  |
| --- |
| Java //Location.java public class Location implements Comparable  public int hashCode() {  return getRow() \* 3737 + getCol(); }  public boolean equals(Object other) {  if (!(other instanceof Location))  return false;  Location otherLoc = (Location) other;  return getRow() == otherLoc.getRow() && getCol() == otherLoc.getCol(); } |

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?

在UnboundedGrid中，数据存储在哈希映射中，键是位置，值是相应位置中的对象。哈希映射的键不能为空。因此，我们需要检查参数位置是否不为 null。

在BoundedGrid中，数据存储在数组中。我们检查位置是否在方法中有效。因此，如果它为 null，则也会引发异常。

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?

get,put,remove的平均时间复杂度：O（1）；

TreeMap的平均时间复杂度：O（logN）;

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

对于getOccupieLocations方法获取占领位置，当使用哈希映射时，返回的信息大多以不同的顺序排列，因为HashMap中的key被放置在哈希表中，然后我们通过key获取它。但是，如果我们使用TreeMap，我们会将数据存储在二叉树中，并以稳定的顺序遍历此树（不排除无序遍历的情况）。

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?

map是可以用在bounded grid的；

采用HashMap：get,put,remove的平均时间复杂度：O（1），getOccupiedLocations()的平均时间复杂度为：O（n）

采用TreeMap:get,put,remove的平均时间复杂度：O（nlogn）,getOccupiedLocations()的平均时间复杂度为：O（n）

相较之下，HashMap更省一些时间，但总体的复杂度都不高，都可以接受。

同时采用2维数组也有优势，如果网格中有多个物体，2维数组只需要存储网格相应位置的对象，而Map需要同时存储位置和对象。

**Exercises**

1.代码如下：

|  |
| --- |
| Java package SparseBoundedGrid3;  import info.gridworld.grid.AbstractGrid; import info.gridworld.grid.Grid; import info.gridworld.grid.Location; import java.util.ArrayList; import java.util.Map; import java.util.HashMap;  public class SparseBoundedGrid3<E> extends AbstractGrid<E> {  private Map<Location, E> occupantMap;  private int numRows;  private int numCols;   public SparseBoundedGrid3(int rows, int cols) {  occupantMap = new HashMap<Location, E>();  numRows = rows;  numCols = cols;  }   @Override  // 要改的  public int getNumRows() { return numRows; }   @Override  // 要改的  public int getNumCols() { return numCols; }   @Override  // 要改的  public boolean isValid(Location loc) {  return loc.getRow() >= 0  && loc.getRow() < getNumRows()  && loc.getCol() >= 0  && loc.getCol() < getNumCols();  }   @Override  // 不用改的  public ArrayList<Location> getOccupiedLocations() {  ArrayList<Location> a = new ArrayList<Location>();  for (Location loc : occupantMap.keySet())  a.add(loc);  return a;  }   @Override  // 不用改的  public E get(Location loc) {  if (loc == null)  throw new NullPointerException("loc == null");  return occupantMap.get(loc);  }   @Override  // 不用改的  public E put(Location loc, E obj) {  if (loc == null)  throw new NullPointerException("loc == null");  if (obj == null)  throw new NullPointerException("obj == null");  return occupantMap.put(loc, obj);  }   @Override  // 不用改的  public E remove(Location loc) {  if (loc == null)  throw new NullPointerException("loc == null");  return occupantMap.remove(loc);  } } |

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? Fill in the following chart to compare the expected Big-Oh efficiencies for each implementation of the SparseBoundedGrid. Let r = number of rows, c = number of columns, and n = number of occupied locations

getOccupiedLocations, get, put, remove 方法

详见表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Methods | SparseGridNode version | LinkedList version | HashMap version | TreeMap version |
| getNeighbors | O(c) | O(c) | O(1) | O(logn) |
| getEmptyAdjacentLocations | O(c) | O(c) | O(1) | O(logn) |
| getOccupiedAdjacentLocations | O(c) | O(c) | O(1) | O(logn) |
| getOccupiedLocations | O(r+n) | O(r+n) | O(n) | O(n) |
| 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.Consider an implementation of an unbounded grid in which all valid locations have non-negative row and column values. The constructor allocates a 16 x 16 array. When a call is made to the put method with a row or column index that is outside the current array bounds, double both array bounds until they are large enough, construct a new square array with those bounds, and place the existing occupants into the new array. Implement the methods specified by the Grid interface using this data structure. What is the Big-Oh 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?

O(1) ：仍然可以从数组中访问值

O(1) ：在当前的边界范围内，就像边界网格一样。

O(row×column)：行和列是当前数组的行和列的数。需要扫描当前的阵列，并将其复制到新的阵列中。