**risingCity.java**

private RedBlackTree redBlackTree;

private MinHeap minHeap;

public risingCity();

private void insert(int buildingNum, int totalTime);

private void print(int buildingNum);

private void print(int buildingNum1, int buildingNum2);

private void execCmd(String cmdAndParams);

private void run(String inputFile);

public static void main(String[] args)

**Colour.java**

public enum Colour {

RED, BLACK

}

**Record.java**

public Record(int buildingNum, int totalTime, int executedTime);

public Record(int buildingNum, int totalTime);

public void increExecutedTime(int increTime);

public int getExecutedTime();

public int getBuildingNum();

public int getTotalTime();

public boolean isFinished();

public String toString();

public int compareTo(Record o);

**RedBlackTree.java**

public TreeNode(int key, Record value, Colour color);

public boolean contains(int buildingNum);

private TreeNode search(int buildingNum, TreeNode node);

public Record get(int buildingNum);

public List<Record> getByInterval(int buildingNum1, int buildingNum2);

private void searchByInterval(int start, int end, TreeNode node, List<Record> res);

public void insert(Record record);

public void remove(Record key);

private TreeNode getParent(TreeNode node);

private boolean isRightChild(TreeNode node);

private boolean isLeftChild(TreeNode node);

private Colour getColor(TreeNode node);

private boolean isRed(TreeNode node);

private boolean isBlack(TreeNode node);

private void setBlack(TreeNode node);

private void setRed(TreeNode node);

private void leftRotate(TreeNode midNode);

private void rightRotate(TreeNode midNode);

private void insert(TreeNode node);

private TreeNode getInsertPos(TreeNode node);

private void insertFixUp(TreeNode node);

private void remove(TreeNode nodeToRemove);

private TreeNode getSuccessor(TreeNode node);

private void removeFixUp(TreeNode node, TreeNode parent);

**MinHeap.java**

public boolean isEmpty();

public MinHeap();

public void insert(Record value);

private void shiftUp(int i);

public Record extractMin();

private void heapify(int i);

private int getMinChild(int i, int j);

private void swap(int i, int j);