In the game of Rummikub® you get tiles numbered from 1-13 of four different colors (red, blue, orange, and black) with no duplicate tiles. The purpose of the game is to make groups or runs from your 14 tiles. Groups are 3 or 4 tiles of the same number with different colors, and runs are three or more consecutive numbered tiles of the same color. The point value of a group or run is the sum of point values of the tiles. The object is to play all of your tiles. This is like the card game gin or gin rummy. What a great program to write!

To complete this program you have been provided the completed RummikubTiles class below.

public class RummikubTiles

{

private String color; // red, blue, orange, black

private int number; // 1, 2, 3, ...., 12, 13

public RummikubTiles(String c, int n) {

color = c; // red, blue, orange and black

number = n; // 1, 2, 3, ...., 12, 13

}

public String getColor() { return color; }

public int getNumber() { return number; }

public boolean equals(Object obj) {

RummikubTiles temp = (RummikubTiles) obj;

return getColor().equals(temp.getColor()) && temp.getNumber() == number;

}

public int hashCode() {

Integer temp = new Integer(number);

return color.hashCode() + temp.hashCode();

}

}

The Rummikub has a single constructor with a List<RummikubTiles> as its only parameter. The List<RummikubTiles> will always contain 14 tiles. In this problem you are to complete four methods in the Rummikub class. The methods are hasGroup, hasRun, maxPointValueInHand and the static method pointValue.

The hasGroup() method returns true if the List<RummikubTiles> containing 14 tiles contains a group of 3 or 4 tiles with the same number and different color. For example, hasGroup() in the table below returns true since the List contains a "red" 5, a "blue" 5 and a "orange" 5.

The hasRun() method returns true if the List<RummikubTiles> containing 14 tiles contains runs of three or more consecutive tiles of the same color. For example, hasRun() in the table below returns true since the List contains a "red" 5, a "red" 6 and a "red" 7.

The maxPointValueInHand() method returns the point value of the highest scoring play (group or run) contained in the List. For example, maxPointValueInHand() in the table below returns 18 the List contains a group of three 5’s worth 15 points and a "red" run worth 18 points (5+6+7).

The following code shows the results of the hasGroup, hasRun and maxPointValueInHand method.

|  |  |
| --- | --- |
| The following code | Returns |
| ArrayList<RummikubTiles> tiles = new  ArrayList<RummikubTiles>();  tiles.add(new RummikubTiles("red", 5));  tiles.add(new RummikubTiles("blue", 5));  tiles.add(new RummikubTiles("orange", 5));  tiles.add(new RummikubTiles("red", 6));  tiles.add(new RummikubTiles("red", 7));  tiles.add(new RummikubTiles("blue", 13));  tiles.add(new RummikubTiles("red", 12));  tiles.add(new RummikubTiles("orange", 1));  tiles.add(new RummikubTiles("orange", 9));  tiles.add(new RummikubTiles("black", 1));  tiles.add(new RummikubTiles("black", 2));  tiles.add(new RummikubTiles("black", 7));  tiles.add(new RummikubTiles("black", 8));  tiles.add(new RummikubTiles("black", 12));  Rummikub r = new Rummikub(tiles); |  |
| r.hasGroup() | true |
| r.hasRun() | true |
| r.maxPointValueInHand() | 18 |

If the tile is changed from a "red" 5, a "red" 2, then the List of tiles does not contain a group and does not contain a run as shown in the table below.

|  |  |
| --- | --- |
| tiles.set(0, new RummikubTiles("red", 2)); |  |
| r.hasGroup() | false |
| r.hasRun() | false |
| r.maxPointValueInHand() | 0 |

The final method is the pointValue() method which is described on the next page.

The final method is the pointValue() method which returns the point value contained in its parameter (a List<RummikubTiles>.

* You may assume that all cards in rTiles are used in calculating the point value.
* You may assume rTiles contains either a group or a run, but not both a group and a run

For example, in the following tables pointValue()returns 14 when rTiles contains a 4-5-6 run and returns 39 when rTiles contains a group with three 13 tiles.

|  |  |
| --- | --- |
| The following code | Returns |
| Rummikub r = new Rummikub(tiles);  ArrayList<RummikubTiles> rTiles  = new ArrayList<RummikubTiles>();  rTiles.add(new RummikubTiles("orange", 5));  rTiles.add(new RummikubTiles("orange", 4));  rTiles.add(new RummikubTiles("orange", 6)); |  |
| r.pointValue(rTiles) | 14 = 4+5+6 |

Another example of the pointValue() method is given in the table below.

|  |  |
| --- | --- |
| rTiles = new ArrayList<RummikubTiles>();  rTiles.add(new RummikubTiles("red", 13));  rTiles.add(new RummikubTiles("orange", 13));  rTiles.add(new RummikubTiles("black", 13)); |  |
| r.pointValue(rTiles) | 39 = 3 \* 13 |