Lab 9: Playing Card Object

Create a functional Card object, that represents a playing card.

It is important that this is a full, professional Card object, like you might see in a real program. As such, you are responsible for creating full, Javadoc-style comments for every public method and field (including constants). Without this, your lab will not get full credit. If you need a template, you can download the Vector or Rational types from last week.

Here are the public functions that you need to implement ("iff" means "if and only if"). Those that override a built-in function should be preceded by an @Override directive.

- Card(int rank, int suit), a constructor which takes a rank and a suit for the new Card.
- Card(), another constructor which produces a random Card. (Remember, you can use the Math.random() method for this.)
- String toString(), which returns a String representing the Card.
- boolean equals(Object otherObject), which returns true iff the argument is a Card, which has the same rank and suit as this one.
- boolean hasSameRank(Card other), which returns true iff the two Cards have the same rank.
- boolean hasSameSuit(Card other), which returns true iff the two Cards have the same suit.
- boolean hasGreaterRank(Card other), which returns true iff this Card outranks the other. Assume that aces are low.
- boolean hasLesserRank(Card other), which returns true iff the other Card outranks this Card. Assume that aces are low.
- boolean isFaceCard(), which true iff the Card is a jack, queen, or king.
- boolean isRed(), which returns true iff the Card is hearts or diamonds.
- boolean isBlack(), which returns true iff the Card is spades or clubs.
- accessors (getters) for rank and suit. (Setters are not needed.)

Many of these methods can be done in only one line!

You must also create a main() method for testing, that demonstrates each of the above functions you have written. Remember to test them both when they should return true, and when they should return false. The output should be sensical; don’t just print “true” and “false” a bunch of times and expect it to be readable. You may also implement as many private methods as you wish, if you feel it will help.