Computer Science 204

Assignment #3

Bunko-Poker

Due Date : Wednesday, March 16, 2009

40 Points

Objective

This is your first big assignment. It is going to test your ability to do string processing, use various loop structures, and work with decision. As you start to develop the program you should also get a sense of top-down program design and learn how to test individual pieces as you build a large piece of code. Be warned – NO WAY YOU WILL WRITE THIS IN A SINGLE NIGHT. It may take days.

Assignment Summary

Your program is going to implement a game called Bunko-Poker. The rules for gameplay are described later, but is essentially a more easily programmed version of the popular game *Yahtzee.*

Your program will make use of the supplied static functions `Dice.roll()` and, if necessary, `Dice.ordered()` to perform the operation of "throwing the dice". You will have to construct your code so that it will simulate a complete game of Bunko-Poker and your output should match that of mine.[1]

The `Dice.class` file will be available on the class website. You should include in your *Eclipse* Java build path. The API for `Dice` is also available on the class website.

GamePlay

In the game of Bunko-Poker, you roll the dice first looking for ones. If one or more of the five dice are a one, then you add up the number of ones. This total is the initial "hand score". You then take the "hand" and look for poker-style hands (described in the table below). If you have any of the poker hands, you will modify the hand score by the appropriate value from the table below. The resulting score is your *adjusted hand score* for that particular roll of the dice. You roll again and if you get ones again, you repeat the procedure and add the adjusted hand score for this roll to the adjusted hand score from the prior roll to get your total score. If you do not get a one, then you move on to twos. You roll again and if you get twos, you add them up and then repeat the procedure to calculate the adjusted hand score. You keep rolling for a particular number until you roll the dice and do not get that number (the example run below will help clarify this). You keep repeating the procedure until you complete rolling for sixes.

[1] Because the dice pull from a uniformly distributed random number generator, it is highly unlikely that you will reproduce the exact results I have.

Hand Name	Sample Dice Roll	Example Dice String	Hand Score Modification
Five of a Kind	⚃⚃⚃⚃⚃	44444	+ 100
Straight	⚀⚁⚂⚃⚄	12345	+ 80
Four of a Kind	⚂⚄⚄⚄⚄	35555	+ 60
Full House	⚃⚃⚀⚀⚀	44111	+ 50
Three of a Kind	⚃⚃⚃⚁⚂	44423	+ 30
Two Pair	⚃⚃⚁⚁⚄	44225	+ 20
Pair	⚂⚂⚁⚃⚅	33246	+ 10
Nothing	⚀⚁⚂⚃⚅	12346	+ High Card

Notes

1. Your source file should be named `BunkoPoker.java` and will submitted to be via e-mail on March 18 by 11:59 PM. The program should be attached to an e-mail with the subject ASSIGNMENT 3 SUBMISSION – YOUR NAME. You may use the body of your e-mail to tell me anything I might need to know about your program.

2. You should make heavy use of methods in your program. All of your method definitions should follow the `main` method as I described in class. While you do not have to use *javadoc* to annotate your program this time, I do expect you to liberally comment all your methods to explain any of your logic. While you will most likely have many methods with names of your own choosing, I do expect to see the following boolean methods (each taking a tt String argument) defined and utilized to determine what poker hands you have:

   (a) `fiveOfAKind`

   (b) `straight`

   (c) `fourOfAKind`

   (d) `fullHouse`

   (e) `threeOfAKind`

   (f) `twoPair`

   (g) `pair`

   I will be grading these carefully for ease of readability, simplicity, and style. Please note, each boolean method described above should only return true if the hand contains only that particular combination. For example, if a particular hand has three of a kind and two of a kind, only the `fullHouse` method should return `true`.

3. Use proper indentation and documentation throughout. Provide comments for all variables and important statements. As this is your first program with multiple loops and nested blocks of code, you will start to see the power of proper indenting. Let *Eclipse* help you!

4. Other than these guidelines, I am letting you pick your own names for variables and methods. Make sure you use descriptive names.

Sample Run

```
  *** BUNKO-POKER GAME ***

  *** ROLLING FOR ONES ***
63513    1    -->  High Card       (+ 6) -->   Total:    7
53112    2    -->  Pair            (+10) -->   Total:   19
43522
  *** ROLLING FOR TWOS ***
54341
  *** ROLLING FOR THREES ***
23433    9    -->  Three of a Kind (+30) -->   Total:   58
36245    3    -->  Straight        (+80) -->   Total:  141
23142    3    -->  Pair            (+10) -->   Total:  154
11113    3    -->  Four of a Kind  (+60) -->   Total:  217
55666
  *** ROLLING FOR FOURS ***
33333
  *** ROLLING FOR FIVES ***
22435    5    -->  Pair            (+10) -->   Total:  232
45345   10    -->  Two Pair        (+20) -->   Total:  262
34261
  *** ROLLING FOR SIXES ***
26266   18    -->  Full House      (+50) -->   Total:  330
32146    6    -->  High Card       (+ 6) -->   Total:  342
33421

  *** G A M E   O V E R ***

Total Score:  342

Scoring Rolls: 10
```

Revision Policy

All assignments handed in on or before the due date that you do not receive full credit for in implementation are eligible for revision.