Ruth Wittek

Project Bundle 2

PokerHand Program -

This program consists of three classes, a Card class, a Deck class, and a HandTester class. The Card class has a String suite, and an int number. It also contains a constructor as well as getter methods for suite and number. The Deck class consists of all the other methods. It has methods for constructing a deck of 52 cards, shuffling the deck, drawing five random cards, getting the deck, accounting for duplicate scenarios (pair, two pairs, three/four of a kind, and full house), a method for a straight, a flush, and a method for printing out the final probabilities of all hand scenarios. The ConstructDeck method creates an ArrayList suite and adds Spades, Diamonds, Hearts, and Clubs to it. It then accounts for all 13 possible numbers (Ace being 1, and then the three face cards being 11, 12, and 13). The five cards and shuffle methods both utilize Random. The method testing for duplicate scenarios has nested for loops and nested ifs. It creates the ArrayList test, then accounts for pairs of one, two, three, four, and six. The straight and flush methods also utilize many if statements. The straight method tests for five cards in descending/ascending number order. It does this with one for loop, and four if statements. If it finds this scenario in the trials, it will print out “Straight”, similar to how the other scenarios will print out their name as well, if they are found. The flush method tests if the five cards are all from the same suite. The HandTester class calls the method to construct the deck, as well as the method that prints out all of the statistics. It tests 10,000 trials before printing out the results.

Plotter/Salter/Smoother -

I chose to put all three (the plotter, the salter, and the smoother) into one program. I believed this would be easier to keep the data all together in one csv file, rather than it getting put into three separate files. I felt it was easiest to compare data and graphs this way. The PlotSaltSmooth class consists of three methods and a main. The first method, function, is the plotter. It sets the function as y. The salter method loops through the y values and adds/subtracts a random number from them. The smoother then loops through and replaces the y values with the average of the y values around it. The main method calls all three of these methods. First though, it has a try-catch block for the csv file. I set the parameters for the x values as -10 to 10 for the three sets of data. I chose to smooth the data twice in hopes for more accuracy and precision, trying to get the closest possible result to the original. I then created graphs of all three, proving that they worked. I decided to use scatter plots that connected points with a line to show that the plotter graph was straight, the salter graph was very jagged, and the smoother tamed down the salter.

Binomial Distribution -

The BinomialDistribution class reads input through a scanner. It has five variables and one method. The for loop calculates the formula for binomial distribution. You use binomial distribution when there are a fixed number of trials, each trial has two possible outcomes (success or failure), the probability of success (p) is the same for each trial, and the trials are all independent. The purpose of this distribution is to calculate the probability of observing x number of successes in N trials.

Geometric Distribution -

This was a very simple class. It consisted of three variables, p, n, and r, where the user would adjust the program to plug in the values that they need to solve their problem. The formula is then computed and printed to three decimal places. Unlike binomial which has a fixed number of trials, the geometric distribution is used when there is a fixed number of successes, and then it counts the number of trials needed to obtain the first success. It represents the number of failures before you get a success in the trials. It also assumes that there are two possible outcomes for each trial (success or failure), the trials are independent, and the probability of success (p) is the same for each trial (just like the binomial distribution).

Poisson Distribution -

This was a very simple class as well. It has two variables, an average and an x. The decimal places for the result are set for three places, just like for the geometric distribution. There is a method containing the formula for the poisson distribution, and a method for calculating a factorial (needed in the formula). The poisson distribution is a bit different from the others. You are able to use it to predict the probability of an event when you know how often the event has occurred (the average). It gives the probability of a given number of events happening in a fixed interval of time. It can also be used as an approximation to the binomial distribution, but only when the value of n in the binomial distribution is large and the value of p is very small.

Hypergeometric Distribution -

This program is a bit more complex than the others. It has a main method to print the solution, a method to compute the factorial, a method to compute the binomial, and a final method that computes the hypergeometric distribution formula. It accounts for BigDecimal. It has four main variables in it, relating to the formula. X for the number of successes in the population, x for the number of successes in the sample, N for the population size, and n for the sample size. The main method is where the user of the program would plug in their values for the variables. This distribution is used when you want to determine the probability of obtaining a certain number of successes without replacement from a specific sample size. The key here is “without replacement”, which is different from the other distributions which have independent trials. The binomial distribution is said to be a good approximation to the hypergeometric distribution , as long as you are sampling 5% or less of the population.