Class StatsLibrary

java.lang.Object
    StatsLibrary

```
public class StatsLibrary
extends Object
```

Date: March 1-2023 This class has methods that will help with probability and applied statistics homework.

Version:

1.0

Author:

Brandon_Pacheco

Constructor Summary

Constructors

Constructor	Description
`StatsLibrary()`	

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
`private BigDecimal`	`BinomialDistribution (double probability, int trials, int successes)`	This method calculates the binomial distribution.
`private BigInteger`	`Combinations(int n, int r)`	This method calculates the combination value when given two values: n, r.
`private ArrayList <Object >`	`complement(ArrayList <Object > A, ArrayList <Object > S)`	This method gets the complement of a subset of objects
`private BigInteger`	`Factorial(int num)`	This method calculates the factorial (!) of a number.

private double	findMean(ArrayList <Integer > list)	This method finds the mean of an integer arraylist.
private double	findMedian(ArrayList <Integer > list)	This method finds the median of an integer arraylist.
private Integer	findMode(ArrayList <Integer > list)	This method finds the mode of an integer arraylist
private BigDecimal	GeometricDistribution (double probability, int trials)	This method calculates the Geometric Distribution.
private ArrayList <Integer >	insertionSort(ArrayList <Integer > list)	This method takes an unsorted integer list and sorts it using the insertion sort algorithm.
private ArrayList <Object >	intersection(ArrayList <Object > A, ArrayList <Object > B)	This method gets the intersection of two sets of objects
private boolean	isSorted(ArrayList <Integer > list)	This method will check if a integer list is sorted or not.
private ArrayList <Integer >	listRandomized()	This method makes an arraylist filled with random integers
private BigInteger	Permutations(int n, int r)	This method calculates the permutation when given two values: n, r
private Double	standardDeviation(ArrayList <Integer > list)	This method calculates the standard deviation of an integer list
void	testBinomialDistribution()	This method tests the binomial distribution method.
void	testCombinations()	This method tests the combinations method
void	testComplement()	This method tests the complement method.
void	testFactorial()	This method tests the factorial method.
void	testGeometricDistribution()	This method tests the Geometric Distribution method

void	testIntersection()	This method tests the intersection method.
void	testMean()	This method tests the mean method.
void	testMedian()	This method tests the median method.
void	testMode()	This method tests the mode method.
void	testPermutations()	This method tests the permutation method.
void	testStandardDeviation()	This method tests the standard deviation method.
void	testStatsLibrary()	This method tests all of testing methods that test the stats library class.
void	testUnion()	This method tests the union method.
private ArrayList <Object >	union(ArrayList <Object > A, ArrayList <Object > B)	This method joins two sets of objects in what's called the union

Methods inherited from class java.lang.Object

clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait , wait , wait

Constructor Details

StatsLibrary

```
public StatsLibrary()
```

Method Details

isSorted

```
private boolean isSorted(ArrayList <Integer > list)
```

This method will check if a integer list is sorted or not.

Parameters:

`list` - An arraylist of integers

Returns:

True if list is sorted. False if list is not sorted.

insertionSort

```
private ArrayList <Integer > insertionSort(ArrayList <Integer > list)
```

This method takes an unsorted integer list and sorts it using the insertion sort algorithm. Algorithm inspired by the book: Introduction to Algorithms 4th edition by Cormen, Leiserson, Rivest, and Stein.

Parameters:

`list` - An arraylist of integers.

Returns:

returns a sorted list from least to greatest.

listRandomized

```
private ArrayList <Integer > listRandomized()
```

This method makes an arraylist filled with random integers

Returns:

Returns a integer list of randomized numbers in random order.

findMean

```
private double findMean(ArrayList <Integer > list)
```

This method finds the mean of an integer arraylist.

Parameters:

`list` - An arraylist that is filled with integers

Returns:

Returns a double value that is the mean. 0 otherwise.

findMedian

```
private double findMedian(ArrayList <Integer > list)
```

This method finds the median of an integer arraylist.

Parameters:

`list` - An arraylist that is filled with integers.

Returns:

Returns a double value that is the median. 0 otherwise.

findMode

```
private Integer  findMode(ArrayList <Integer > list)
```

This method finds the mode of an integer arraylist

Parameters:

`list` - An arraylist that is filled with integers.

Returns:

Returns an integer if that value was counted more than once and appeared the most than all other values. Returns a null value to signal that no value appeared more than once.

standardDeviation

```
private Double  standardDeviation(ArrayList <Integer > list)
```

This method calculates the standard deviation of an integer list

Parameters:

`list` - A list of integer values

Returns:

Returns the standard deviation value.

Permutations

```
private BigInteger  Permutations(int n,
                                 int r)
```

This method calculates the permutation when given two values: n, r

Parameters:

`n` - An int variable that represents the total "objects" in the problem.

`r` - An int variable that represents the given "object(s)" from n to rearrange at a time.

Returns:

Returns a BigInteger value of the calculated permutation.

Combinations

```
private BigInteger  Combinations(int n,
                                 int r)
```

This method calculates the combination value when given two values: n, r.

Parameters:

`n` - An int value that represents the total number of "objects" in the problem.

`r` - An int value that represents the total objects to take from n.

Returns:

Returns a BigInteger value of the calculated combinations.

Factorial

```
private BigInteger  Factorial(int num)
```

This method calculates the factorial (!) of a number.

Parameters:

`num` - An int value that is the number to be calculated

Returns:

Returns a BigInteger value that is the calculated factorial.

union

```
private ArrayList <Object > union(ArrayList <Object > A,
                                  ArrayList <Object > B)
```

This method joins two sets of objects in what's called the union

Parameters:

`A` - An object arraylist

`B` - An object arraylist

Returns:

Returns an arraylist that is the union of the two passed lists.

intersection

```
private ArrayList <Object > intersection(ArrayList <Object > A,
                                         ArrayList <Object > B)
```

This method gets the intersection of two sets of objects

Parameters:

`A` - An arraylist of objects

`B` - An arraylist of objects

Returns:

Returns an arraylist that is the intersection of the two passed lists

complement

```
private ArrayList <Object > complement(ArrayList <Object > A,
                                       ArrayList <Object > S)
```

This method gets the complement of a subset of objects

Parameters:

`A` - An arraylist of objects which is the subset

`S` - An arraylist of objects which is the set.

Returns:

Returns an arraylist of objects that is the complement of the passed subset.

BinomialDistribution

```
private BigDecimal  BinomialDistribution(double probability,
                                         int trials,
                                         int successes)
```

This method calculates the binomial distribution.

Parameters:

`probability` - A double value that is the probability of success

`trials` - An int value that is the number of trials to run the experiment

`successes` - An int value that is the number of successful trials in the experiment

Returns:

Returns a BigDecimal value that is the calculated binomial distribution.

GeometricDistribution

```
private BigDecimal  GeometricDistribution(double probability,
                                          int trials)
```

This method calculates the Geometric Distribution.

Parameters:

`probability` - A double value that represents the probability of success.

`trials` - An int value that is the number of trials to run

Returns:

Result a BigDecimal that is the calculation of the Geometric Distribution.

testMean

```
public void testMean()
```

This method tests the mean method.

testMedian

```
public void testMedian()
```

This method tests the median method.

testMode

```
public void testMode()
```

This method tests the mode method.

testStandardDeviation

```
public void testStandardDeviation()
```

This method tests the standard deviation method.

testUnion

```
public void testUnion()
```

This method tests the union method.

testIntersection

```
public void testIntersection()
```

This method tests the intersection method.

testComplement

```
public void testComplement()
```

This method tests the complement method.

testStatsLibrary

```
public void testStatsLibrary()
```

This method tests all of testing methods that test the stats library class.

testBinomialDistribution

```
public void testBinomialDistribution()
```

This method tests the binomial distribution method.

testFactorial

```
public void testFactorial()
```

This method tests the factorial method.

testPermutations

```
public void testPermutations()
```

This method tests the permutation method.

testCombinations

```
public void testCombinations()
```

This method tests the combinations method

testGeometricDistribution

```
public void testGeometricDistribution()
```

This method tests the Geometric Distribution method