

**UNIVERSITY OF PUERTO RICO
MAYAGUEZ CAMPUS
COLLEGE OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT**

PROBABILIUS

Harry Hernández Rivera

harry.hernandez@upr.edu

Geraldo Lopez Rosa

geraldo.lopez1@upr.edu

Hidalgo Pomales Delgado

hidalgo.pomales@upr.edu

Lewis D. Rodriguez Sierra

lewis.rodriquez@upr.edu

ICOM4036

Prof. Wilson Rivera Gallego

July 2nd, 2017

Introduction:

What is Probability and Statistics? Probability and statistics are sections of mathematics that deal with data collection and analysis. Probability is the study of chance and is a very fundamental subject that we apply in everyday living, while statistics is more concerned with how we handle data using different analysis techniques and collection methods. Probability theory consists of a vast collection of axioms and theorems that provides the scientific community with many contributions, including the naming and description of random variables that occur frequently in applications, the theoretical results associated with these random variables, and the applied results associated with these random variables for statistical applications. Probabilius will be a programming language to solve basic probability and statistics problems. This language will feature some of the most used concepts and formulas from the probability and statistics environment. The user will be able to write down the desired operations such as combinations, permutations, standard deviation, mean and other measurements of dispersion and central tendency. Also, the user will be able to resolve basic probabilities and set theory problems. Probabilius will interpret what the user entered and display the desired outcome of the calculation. On the other hand, our motivations and reasons for developing this programming language includes, expanding our knowledge in the Probability and Statistics field, simplify the complex calculations for the user needs, improve performance of the user's work, challenge ourselves to explore an area outside of our expertise, and develop our critical thinking for problem solving skills.

Language Tutorial

The purpose of Probabilius is to help people calculate the basic functions of statics and probabilities. We are going to show how people can use the different functions that our language offers. First thing to do is to run Probabilius. To do this, open your computer's terminal and go to the directory of the Probabilius project. Once there, enter the following command to run the language:

```
python3 Probabilius.py
```

Now that Probabilius is running, the user can use the different functions. To make an example, let us assume that the user wants to calculate the average of a list of numbers. To make this possible the user must enter the following command, `avg(list of numbers)`, here is an example:

```
avg((10,20))
```

The system will answer the user by displaying the result of the average of those numbers:

```
Probabilius > 15
```

Now let us assume that the user wants to know the median of another list of numbers. The user will enter the following command, **median(list of numbers)**:

```
median((13, 18, 13, 14, 13, 16, 14, 21, 13))
```

Yielding a result of:

```
Probabilius > 14
```

Suppose that now the user want to know the trimmed mean of a list. To calculate this number, the user needs to input the percent he wants to trim followed by the list of numbers. The command to use is the following, **trim(percentage, list of numbers)**:

```
trim(20, (60, 81, 83, 91, 99))
```

Since the user wants to trim 20% of the list (that is 10% from the beginning of the list and 10% from the end), the system displays the following result:

```
Probabilius > 85
```

Another case can be that the user wants to calculate the union of two sets. The user needs to input: **union(list of numbers, list of numbers)**. The following is an illustration of how the user needs to enter the data:

```
union((1,2,3,4,5,7),(1,4,8,9))
```

This yields a result of:

```
Probabilius > (1,2,3,4,5,7,8,9)
```

The above command is similar to the intersection and complement command, just use **intersection(list of numbers)**, to calculate the intersection, and **comp(list of numbers)**, to calculate the complement.

The following example shows how the user can calculate the number of circular permutations using the command **cirPer(number)**:

```
cirPer(6)
```

Which yields:

```
Probabilius > 120
```

Unlike other commands, this command only needs one set of parentheses to work. The total permutations command is similar to the “cirPer” command except that you substitute the “cirPer” with “totalPer”:

```
Probabilius > totalPer(10)
```

```
Probabilius > 90
```

Now, the next several commands use the same format, which is: command((list of numbers)). These commands are:

Mode:

```
Probabilius > mode((1,2,1,4,5,8,1))
```

```
Probabilius > [('Mode: 1', 'Frequency: 2')]
```

Standard Deviation:

```
Probabilius > stanDev((600, 470, 170, 430, 300))
```

```
Probabilius > 147
```

Variance:

```
Probabilius > var((600, 470, 170, 430, 300))
```

```
Probabilius > 21,704
```

Range:

```
Probabilius > range((4, 6, 9, 3, 7))
```

```
Probabilius > 3
```

Partial Permutations (This command only accepts a list of two numbers):

```
Probabilius > partial((2,5))
```

```
Probabilius > 20
```

Combinations (This command only accepts a list of two numbers):

```
Probabilius > combination((10,2))
```

```
Probabilius > 45
```

Reference Manual:

Probabilius allows the user to interact with it by providing keywords to calculate all the basic functions of probabilities and statistics. These keywords are the following:

avg(): This command calculates the average of a list of numbers. Inside the parenthesis, the user must input a list of numbers.

Example: avg((List of Numbers))

median(): This command calculates the median of a list of numbers. Inside the parenthesis, the user must input a list of numbers.

Example: median((List of Numbers))

trim(): This command calculates the trimmed mean of a list of numbers. Inside the parenthesis, the user must input the percent he likes to trim and a list of numbers.

Example: trim(Percentage,(List of Numbers))

mode(): This command calculates the mode of a list of numbers. Inside the parenthesis, the user must input a list of numbers.

Example: mode((List of Numbers))

stanDev(): This command calculates the standard deviation of a list of numbers. Inside the parenthesis, the user must input a list of numbers.

Example: stanDev((List of Numbers))

var(): This command calculates the variance of a list of numbers. Inside the parenthesis, the user must input a list of numbers.

Example: var((List of Numbers))

range():This command calculates the range of a list of numbers. Inside the parenthesis, the user must input a list of numbers.

Example: range((List of Numbers))

union():This command calculates the union of two sets. Inside the parenthesis, the user must enter two sets of numbers (list of numbers) separated by a comma.

Example: mode((List of Numbers), (List of Numbers))

intersection():This command calculates the intersection of two sets. Inside the parenthesis, the user must enter two sets of numbers (list of numbers) separated by a comma.

Example: intersection((List of Numbers), (List of Numbers))

complement():This command calculates the complement of two sets. Inside the parenthesis, the user must enter two sets of numbers (list of numbers) separated by a comma.

Example: complement((List of Numbers), (List of Numbers))

totalPer():This command calculates the total permutations possible, given a number. Inside the parenthesis, the user must enter a number to calculate its total permutations.

Example: totalPer(Number)

cirPer():This command calculates the circular permutations possible, given a number. Inside the parenthesis, the user must enter a number to calculate its total permutations.

Example: cirPer(Number)

partial():This command calculates the partial permutations, given two number (total number of items, selected items). Inside the parenthesis, the user must enter two number separated by a comma.

Example: partial((Total number of items, Selected Items)) => partial((List of Numbers))

combination():This command calculates the combination of a list of numbers. Inside the parenthesis, the user must input a list of numbers.

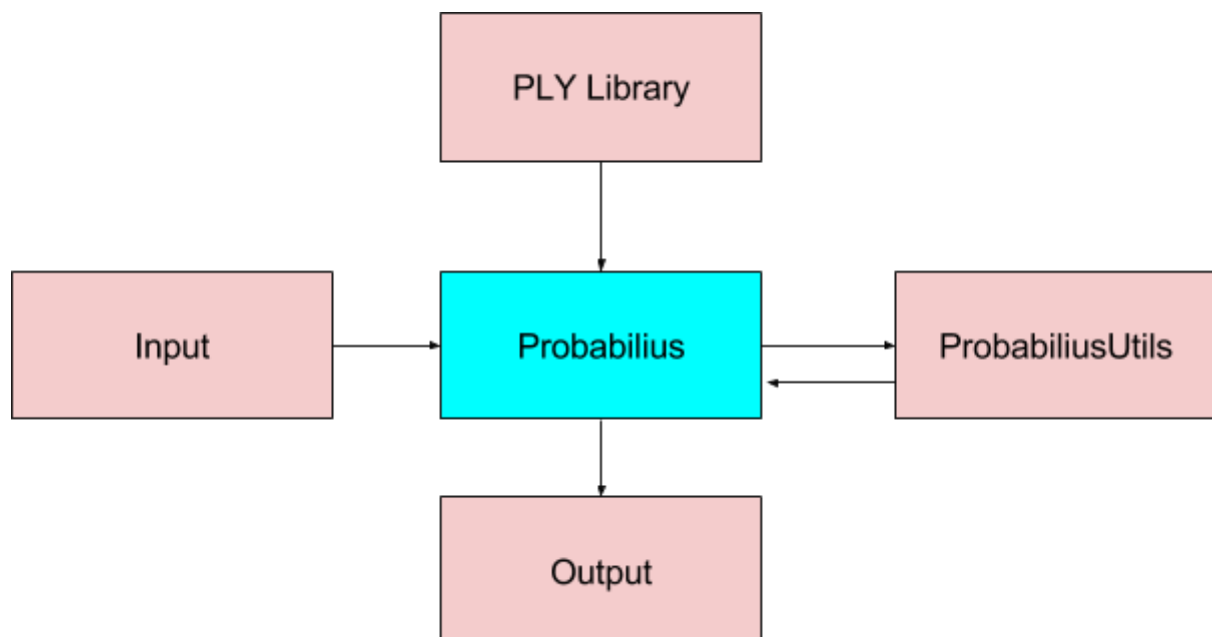
Example: combination((Number, Number)) => combination (List of Numbers))

The arguments that are inside the parenthesis have syntax rules that must be met:

1. **List of Numbers:** This argument is composed of numbers, commas, and parentheses. It can be a single number or two or more numbers separated by comma(s). The argument starts with a parenthesis followed by a combination of numbers and commas (combination must start with a number and end in a number), and ending with a closing parenthesis.
2. **Percent:** A positive integer that represents a percentage value.
3. **Number:** A positive integer.

Language Development

Translator Language Architecture:



Interface Between Modules:

During the execution of the Probabilius Programming Language, the Lexer interprets the user input to see if the syntax is correct accordingly to the programming language specifications. After the interpretation, the Parser starts looking for the meaning of each character that was presented by the user to finally form a valid regular expression. Next, the connection between the interfaces start. The

utilities class calculate the specific result depending on the regular expression and the parameters received from the Probabilius.py module and shows the output to the user.

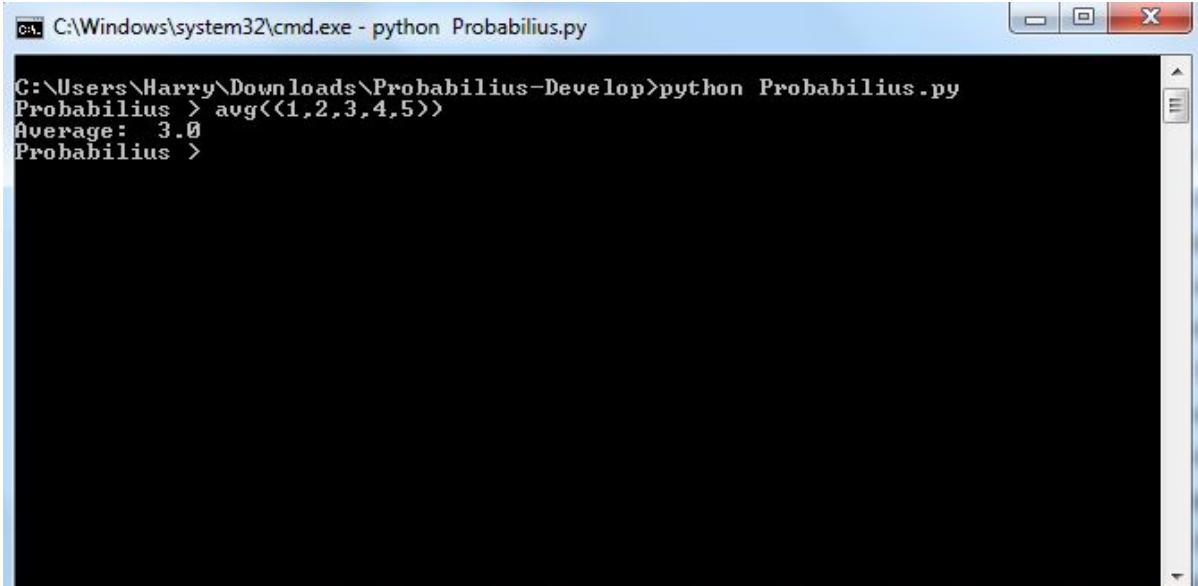
Software Development Environment:

- **Pycharm** - IDE used to develop Probabilius programming language.
- **Python** - A high-level general-purpose programming language.
- **GitHub** - Version control repository and Internet hosting service.

Test Methodology:

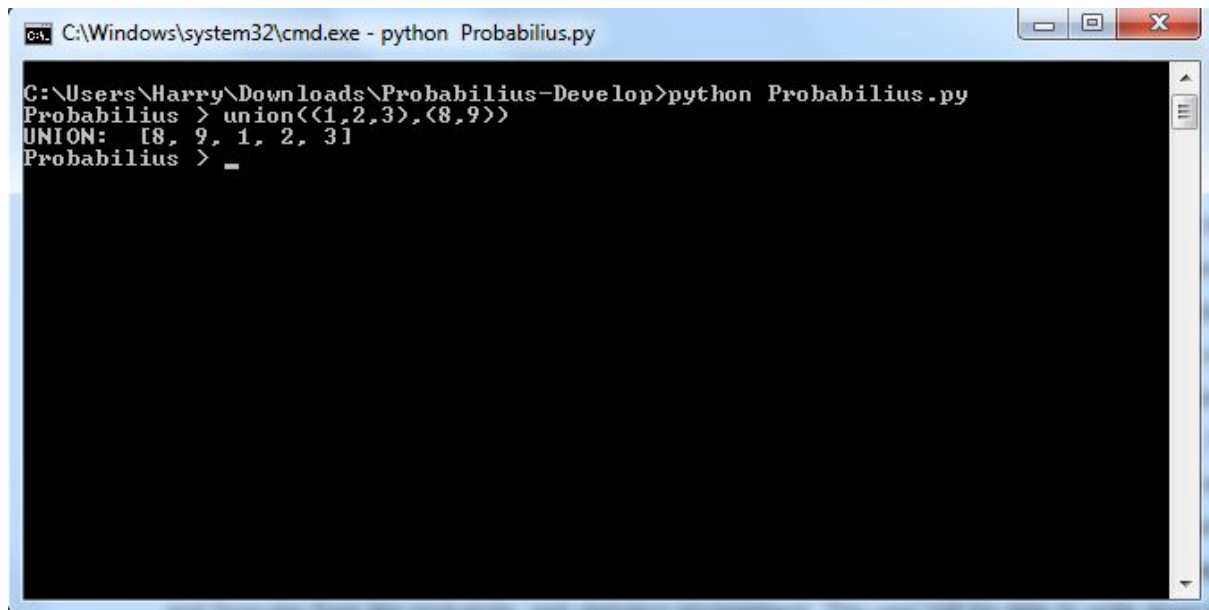
- Tested by running the program after implementing a part and verifying that it displays the correct output.
- User feedback from selected participants.

Programs For Testing:



```
C:\Windows\system32\cmd.exe - python Probabilius.py
C:\Users\Harry\Downloads\Probabilius-Develop>python Probabilius.py
Probabilius > avg(<1,2,3,4,5>)
Average: 3.0
Probabilius >
```

Figure 1. Example run of the average keyword.



```
C:\Windows\system32\cmd.exe - python Probabilius.py
C:\Users\Harry\Downloads\Probabilius-Develop>python Probabilius.py
Probabilius > union(<1,2,3>,<8,9>)
UNION: [8, 9, 1, 2, 3]
Probabilius > _
```

Figure 2. Example run of the union keyword.

Conclusion:

The team built the lexer and parser to create the programming language Probabilius using python. You can run the language from any terminal. The language solves basic probability and statistic problems. The features that Probabilius calculates are: addition, multiplication, division, subtraction, average, mode, range, median, max, min, permutations, combinations, union, intersection, complement, variance, standard deviation, and trimmed mean. The user can put the desired function with the numbers or the list of numbers and it will solve depending on the operation.