|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  | | --- | --- | --- | |  | МИРЭА_ЭМБЛЕМА_приказ |  | | Министерство образования и науки РОССИЙСКОЙ ФЕДЕРАЦИИ | | |  Федеральное государственное бюджетное образовательное учреждение высшего образования  **«Российский технологический университет»**  **МИРЭА** | |
|  | |
|  | |
|  |  |

КАФЕДРА ВЫСШЕЙ МАТЕМАТИКИ

Лабораторная работа 1

 по курсу «**Теория вероятностей и математическая статистика, часть 2**»

Тема: \_\_\_\_\_\_\_ **Первичная обработка выборки из** \_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_ **дискретной генеральной совокупности**\_\_\_\_\_\_\_\_\_\_

Выполнил:

Студент 3-го курса

Жолковский Д. А.

Группа: КМБО-01-16

МОСКВА 2019

**Лабораторная работа по Математической статистике № 1 «Первичная обработка выборки из дискретной генеральной совокупности»**

**Задание 1.** Получить выборку, сгенерировав 150 псевдослучайных чисел, распределенных по биномиальному закону с параметрами n и p.  
 n=5 + Vmod17 p=0,1 + 0,01V

**Задание 2.** Получить выборку, сгенерировав 150 псевдослучайных чисел, распределенных по геометрическому закону с параметром p.

p =0,1 + 0,01V

**Задание 3.** Получить выборку, сгенерировав 150 псевдослучайных чисел, распределенных по закону Пуассона с параметром λ .  
 λ = 0,7 + 0,07V

Для всех выборок построить:

1) статистический ряд;

2) полигон относительных частот;

3) эмпирическую функцию распределения;

Найти:

1) выборочное среднее;

2) выборочную дисперсию;

3) выборочное среднее квадратическое отклонение;

4) выборочную моду;

5) выборочную медиану;

6) выборочный коэффициент асимметрии;

7) выборочный коэффициент эксцесса.

V − номер варианта. Вычисления проводить с точностью до 0,00001 .

**Теоретические сведения**

Полученную выборку {*x*1 , *x*2 , *x*3 , ..., *xN*} упорядочить по возрастанию, определить частоты и относительные частоты (частости) , построить статистический ряд:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | … |  |
|  |  | … |  |
|  |  | … |  |

Полигон относительных частот – ломаная линия, соединяющая последовательно точки с координатами .

**Эмпирическая функция распределения**

**Выборочное среднее**

**Выборочная дисперсия**

**Выборочный момент k-ого порядка**

**Выборочный центральный момент k-ого порядка**

**Выборочное среднее кврадратическое отклонение**

**Выборочная медиана**

**Выборочная мода** – это значение , которому соответствует максимальная частота.

**Выборочный коэффициент асимметрии**

**Выборочный коэффициент эксцесса**

**Ряд распределения** - структурная группировка с целью выделения характерных свойств и закономерностей изучаемой совокупности.

**Математическое ожидание** – понятие среднего значения случайной величины в теории вероятностей.

**Дисперсия** – отклонение величины от ее математического ожидания.

**Среднеквадратическое отклонение** – показатель рассеивания значений случайной величины относительно ее математического ожидания.

**Мода** – значение во множестве наблюдений, которое встречается наиболее часто.

**Медиана** – возможное значение признака, которое делит вариационный ряд выборки на две равные части.

**Коэффициент асимметрии** используется для проверки распределения на симметричность, а также для грубой предварительной проверки на нормальность.

Если плотность распределения симметрична, то выборочный коэффициент асимметрии равен нулю, если левый хост распределения тяжелее – больше нуля, легче – меньше.

**Коэффициент эксцесса** используется для проверки на нормальность.

Нормальное распределение имеет нулевой эксцесс. Если хвосты распределения «легче», а пик острее, чем у нормального распределения, то коэффициент эксцесса положительный; если хвосты распределения «тяжелее», пик «приплюснутый», чем у нормального распределения, то отрицательный.

**Биномиальное распределение**

**Биномиальное распределение** – распределение количества «успехов» в последовательности из n независимых случайных экспериментов, таких что вероятность «успеха» в каждом из них равна p.

Математическое ожидание: *np*

Дисперсия: *npq, q=1-p*

Среднеквадратическое отклонение:

Мода: [(n+1)p], если (n+1)p – дробное; (n+1)p- , если np – целое;

Медиана: *Round(np)*

Коэффициент асимметрии: 

Коэффициент эксцесса: 

**Геометрическое распределение**

**Геометрическое распределение** – распределение величины, равной количеству испытаний случайного эксперимента до наблюдения первого «успеха».

Математическое ожидание:, *q=1-p*

Дисперсия: , *q=1-p*

Среднее квадратичное отклонение: 

Мода: 0

Медиана: , если – дробное; , если - целое

Коэффициент асимметрии: 

Коэффициент эксцесса: 

**Распределение Пуассона**

**Распределение Пуассона** – вероятностное распределение дискретного типа, моделирует случайную величину, представляющую собой число событий, произошедших за фиксированное время, при условии, что данные события происходят с некоторой фиксированной средне интенсивностью и независимо друг от друга.

Математическое ожидание: λ

Дисперсия: λ

Среднеквадратическое отклонение: 

Мода: 

Медиана: 

Коэффициент асимметрии: 

Коэффициент эксцесса: 

**Средства языка Octave**

В программе расчёта используются следующие средства языка:

Функции:

• binornd(n, p, s, z) - возвращает матрицу случайных значений из биноми- ального распределения с параметрами n и p , где n есть число испытаний и p есть вероятность успеха, s – количество строк в возвращаемой матрице, z – ко- личество столбцов.

• geornd(p, s, z) - возвращает матрицу случайных значений из геометриче- ского распределения с параметром р, s – количество строк в возвращаемой мат- рице, z – количество столбцов.

• poissrnd(λ, s, z) - возвращает матрицу случайных значений из распреде- ления Пуассона с параметром λ, s – количество строк в возвращаемой матрице, z – количество столбцов.

• sort(x) - возвращает копию х с элементами, расположенными в порядке возрастания.

• sqrt(x) – возвращает квадратный корень из числа x.

• max(X) в случае одномерного массива возвращает наибольший элемент; в случае двумерного массива - это вектор-строка, содержащая максимальные элементы каждого столбца.

Операторы управления:

• for – endfor

• if – else – endif

• break

А также арифметические и логические операторы.

**Результаты расчетов с комментариями**

**Задание 1:**

n = 14, p = 0,19

Выборка:

Неупорядоченная:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 4 | 1 | 2 | 3 | 5 | 4 | 2 | 2 | 1 | 4 | 0 | 1 | 4 | 4 |
| 3 | 0 | 1 | 3 | 4 | 3 | 4 | 2 | 1 | 3 | 1 | 3 | 2 | 2 | 4 |
| 3 | 3 | 0 | 3 | 0 | 1 | 2 | 2 | 1 | 3 | 5 | 1 | 1 | 1 | 1 |
| 4 | 5 | 3 | 1 | 0 | 2 | 5 | 0 | 1 | 2 | 2 | 2 | 2 | 1 | 4 |
| 2 | 1 | 3 | 2 | 3 | 3 | 3 | 2 | 1 | 4 | 3 | 3 | 4 | 1 | 3 |
| 4 | 1 | 0 | 3 | 2 | 2 | 4 | 3 | 3 | 2 | 2 | 4 | 3 | 4 | 5 |
| 2 | 2 | 5 | 1 | 2 | 2 | 2 | 1 | 2 | 5 | 2 | 4 | 3 | 6 | 2 |
| 1 | 3 | 4 | 3 | 3 | 5 | 2 | 0 | 3 | 3 | 2 | 4 | 1 | 3 | 2 |
| 5 | 2 | 1 | 4 | 1 | 3 | 3 | 1 | 4 | 2 | 1 | 0 | 1 | 2 | 2 |
| 6 | 4 | 3 | 4 | 6 | 4 | 6 | 4 | 2 | 2 | 2 | 2 | 1 | 1 | 6 |

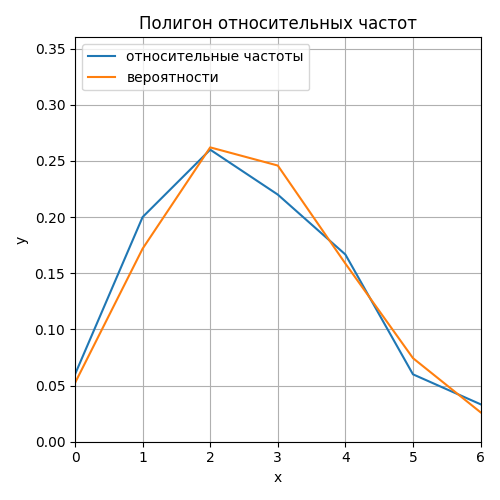
Упорядоченная:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 3 | 3 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 6 | 6 | 6 | 6 | 6 |

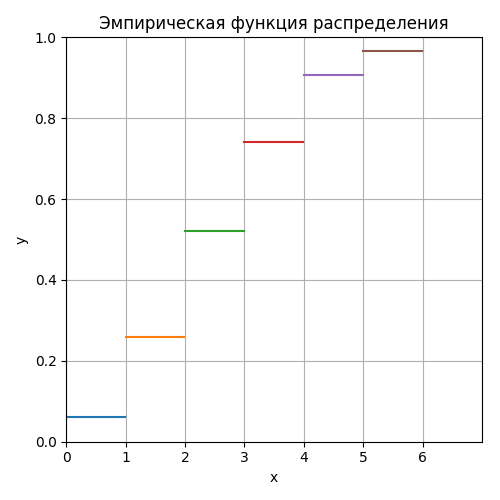
Статистический ряд:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| xi | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| ni | 9 | 30 | 39 | 33 | 25 | 9 | 5 |
| wi | 0.06 | 0.2 | 0.26 | 0.22 | 0.1666 | 0.06 | 0.0333 |

Полигон относительных частот:



Эмпирическая функция распределения:



Выборочное среднее: 2.5466

Выборочная дисперсия: 2.1011

Выборочное среднее квадратическое отклонение: 1.4495

Выборочная мода: 2

Выборочная медиана: 2

Выборочный коэффициент асимметрии: 0.3340

Выборочный коэффициент эксцесса: -0.4272

**Задание 2**

p = 0.19

Выборка:

Неупорядоченная:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 24 | 0 | 2 | 8 | 0 | 7 | 0 | 11 | 10 | 0 | 2 | 9 | 1 |
| 6 | 1 | 12 | 6 | 2 | 1 | 3 | 0 | 0 | 6 | 4 | 16 | 1 | 2 | 1 |
| 14 | 6 | 0 | 14 | 3 | 0 | 1 | 10 | 13 | 7 | 11 | 2 | 3 | 2 | 1 |
| 0 | 6 | 1 | 5 | 4 | 8 | 3 | 5 | 2 | 2 | 0 | 10 | 4 | 1 | 5 |
| 0 | 10 | 4 | 5 | 3 | 0 | 17 | 1 | 0 | 0 | 2 | 2 | 11 | 1 | 32 |
| 2 | 2 | 4 | 5 | 0 | 1 | 4 | 7 | 5 | 2 | 11 | 11 | 0 | 2 | 3 |
| 4 | 7 | 1 | 11 | 0 | 3 | 3 | 9 | 1 | 5 | 2 | 9 | 0 | 0 | 0 |
| 4 | 0 | 5 | 0 | 3 | 5 | 3 | 1 | 0 | 0 | 1 | 10 | 5 | 0 | 0 |
| 15 | 3 | 21 | 1 | 1 | 3 | 1 | 10 | 7 | 4 | 8 | 3 | 10 | 15 | 8 |
| 3 | 0 | 10 | 34 | 2 | 20 | 4 | 5 | 9 | 1 | 8 | 3 | 11 | 10 | 5 |

Упорядоченная:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 3 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 6 | 6 | 6 |
| 6 | 6 | 7 | 7 | 7 | 7 | 7 | 8 | 8 | 8 | 8 | 8 | 9 | 9 | 9 |
| 9 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 11 | 11 | 11 | 11 | 11 |
| 11 | 11 | 12 | 13 | 14 | 14 | 15 | 15 | 16 | 17 | 20 | 21 | 24 | 32 | 34 |

Статистический ряд:

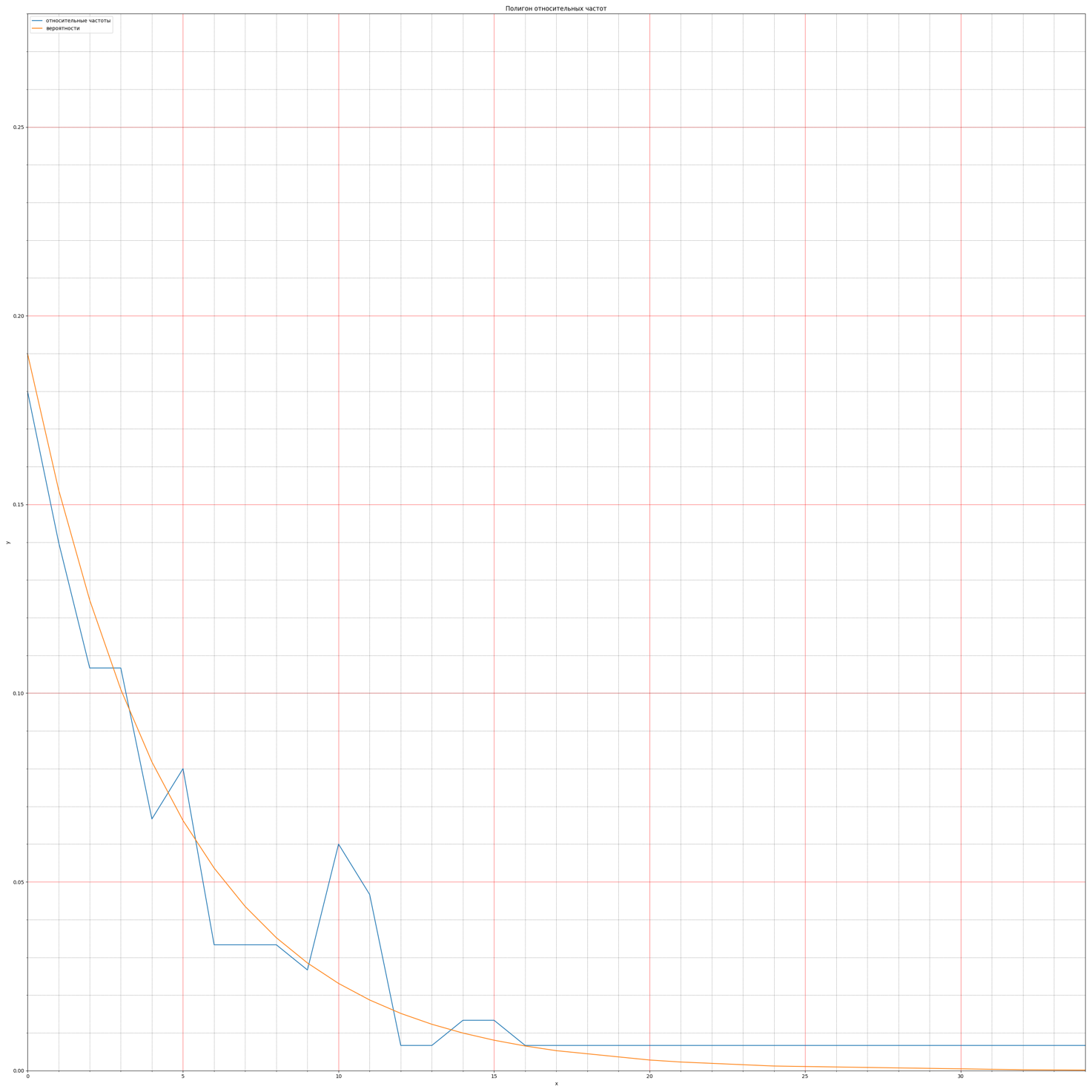
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| xi | 0 | 1 | 2 | 3 | 4 | 5 |
| ni | 27 | 21 | 16 | 16 | 10 | 12 |
| wi | 0.18 | 0.14 | 0.10667 | 0.10667 | 0.0667 | 0.08 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| xi | 6 | 7 | 8 | 9 | 10 | 11 |
| ni | 5 | 5 | 5 | 4 | 9 | 7 |
| wi | 0.0333 | 0.0333 | 0.0333 | 0.0266 | 0.06 | 0.04 |

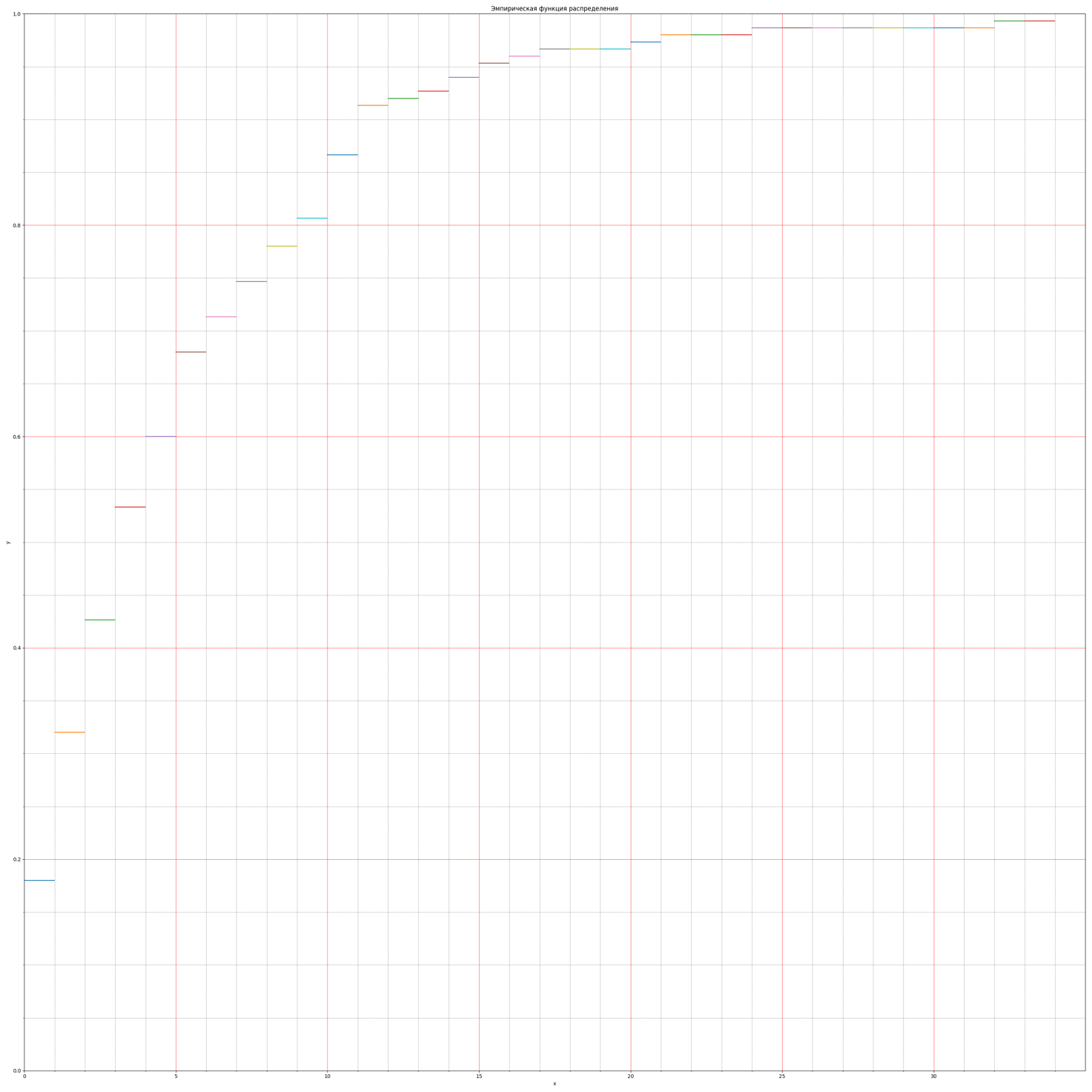
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| xi | 12 | 13 | 14 | 15 | 16 | 17 |
| ni | 1 | 1 | 2 | 2 | 1 | 1 |
| wi | 0.0066 | 0.0066 | 0.0133 | 0.0133 | 0.0066 | 0.0066 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| xi | 20 | 21 | 24 | 32 | 34 |
| ni | 1 | 1 | 1 | 1 | 1 |
| wi | 0.0066 | 0.0066 | 0.0066 | 0.0066 | 0.0066 |

Полигон относительных частот:



Эмпирическая функция распределения:



Выборочное среднее: 5.04

Выборочная дисперсия: 33.2784

Выборочное среднее квадратическое отклонение: 5.7687

Выборочная мода: 0

Выборочная медиана: 3.0

Выборочный коэффициент асимметрии: 2.1863

Выборочный коэффициент эксцесса: 6.5721

**Задание 3**

λ = 1.33

Выборка:

Неупорядоченная:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 3 | 0 | 2 | 1 | 1 | 1 | 0 | 3 | 2 | 0 | 2 | 0 | 3 |
| 1 | 1 | 3 | 2 | 2 | 2 | 0 | 3 | 1 | 2 | 1 | 0 | 0 | 1 | 3 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 2 | 2 | 4 | 2 | 1 | 1 | 4 | 2 |
| 6 | 0 | 1 | 1 | 6 | 1 | 1 | 3 | 1 | 1 | 0 | 4 | 3 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 4 | 0 | 0 | 3 | 2 | 1 | 0 | 3 | 1 |
| 0 | 1 | 1 | 0 | 0 | 2 | 1 | 1 | 0 | 2 | 3 | 1 | 6 | 2 | 1 |
| 1 | 2 | 1 | 2 | 0 | 2 | 2 | 3 | 2 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 2 | 0 | 1 | 1 | 2 | 1 | 0 | 5 | 1 | 4 | 1 |
| 2 | 0 | 3 | 0 | 1 | 2 | 1 | 2 | 1 | 3 | 2 | 1 | 0 | 1 | 1 |
| 0 | 0 | 3 | 2 | 2 | 0 | 2 | 1 | 0 | 2 | 2 | 1 | 1 | 0 | 0 |

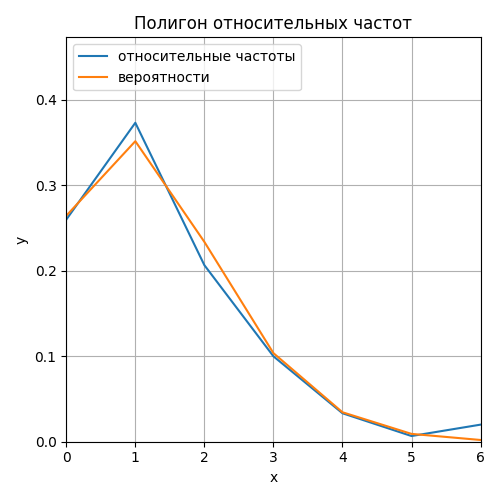
Упорядоченная:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 3 | 3 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 5 | 6 | 6 | 6 |

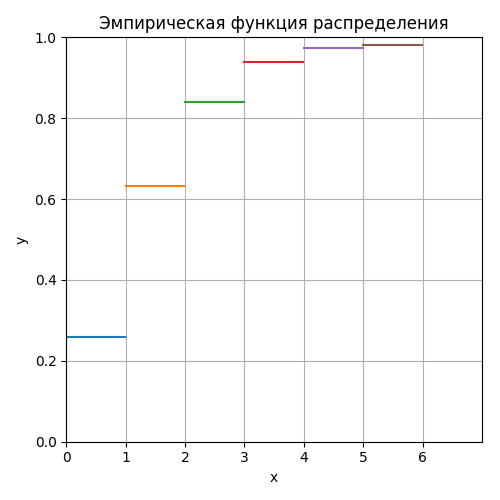
Статистический ряд:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| xi | 0 | 1 | 2 | 3 | 4 |
| ni | 39 | 56 | 31 | 15 | 5 |
| wi | 0.426 | 0.3733 | 0.2066 | 0.01 | 0.0333 |

|  |  |  |
| --- | --- | --- |
| xi | 5 | 1 |
| ni | 1 | 3 |
| wi | 0.0067 | 0.02 |

Полигон относительных частот:

Эмпирическая функция распределения:



Выборочное среднее: 1.3733

Выборочная дисперсия: 1.6339

Выборочное среднее квадратическое отклонение: 1.2782

Выборочная мода: 1

Выборочная медиана: 1

Выборочный коэффициент асимметрии: 1.2885

Выборочный коэффициент эксцесса: 2.0822

**Анализ результатов**

**Задание 1**

n = 14, q = 0,81, p = 0,19

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название показателя | Экспериментальное значение | Теоретическое  значение | Абсолютное отклонение | Относительное отклонение |
| Выборочное среднее | 2.5466 | 2.66 | 0.1134 | 4.4529% |
| Выборочная дисперсия | 2.1011 | 2.1546 | 0.0535 | 2.5462% |
| Выборочное среднее квадратичное отклонение | 1.4495 | 1.4678 | 0.0183 | 1.2625% |
| Выборочная мода | 2 | 2 | 0 | 0% |
| Выборочная медиана | 2 | 3 | 1 | 50% |
| Выборочный коэффициент асимметрии | 0.334 | 0.4223 | 0.3889 | 26.4371% |
| Выборочный коэффициент эксцесса | -0.4272 | 0.0355 | 0.4627 | 203.3802% |

{|wi-pi|}=[0.0076, 0.0281, 0.0020407421518351954, 0.0258, 0.008, 0.0144, 0.0071] max{|wi-pi|, i=1,…,m}=0.0281

**Задание 2**

q = 0,81, p = 0,19

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название показателя | Экспериментальное значение | Теоретическое  значение | Абсолютное отклонение | Относительное отклонение |
| Выборочное среднее | 5.04 | 4.2631 | 0.7769 | 15.4146% |
| Выборочная дисперсия | 33.2784 | 22.4376 | 10.8408 | 32.576% |
| Выборочное среднее квадратичное отклонение | 5.7687 | 4.7368 | 1.0319 | 17,8879% |
| Выборочная мода | 0.0 | 0.0 | 0 | 0% |
| Выборочная медиана | 3.0 | 3.0 | 0 | 0% |
| Выборочный коэффициент асимметрии | 2.1863 | 2.0111 | 0.1752 | 8.0135% |
| Выборочный коэффициент эксцесса | 6.5721 | 6.0445 | 0.5276 | 8.0278% |

{|wi-pi|}=[0.01, 0.0138, 0.0179, 0.0056, 0.0151, 0.0137, 0.0203, 0.0101, 0.0018, 0.0018, 0.0369, 0.0279, 0.0084, 0.0056, 0.0033, 0.0052, 0.0001, 0.0013, 0.0038, 0.0043, 0.0054, 0.0064, 0.0065]

max{|wi-pi|, i=1,…,m} = 0.0369

**Задание 3**

λ = 1.33

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название показателя | Экспериментальное значение | Теоретическое  значение | Абсолютное отклонение | Относительное отклонение |
| Выборочное среднее | 1.3733 | 1.33 | 0.0433 | 3.2556% |
| Выборочная дисперсия | 1.6339 | 1.33 | 0.3039 | 22.8496% |
| Выборочное среднее квадратичное отклонение | 1.2782 | 1.1532 | 0.125 | 10.8394% |
| Выборочная мода | 1 | 1 | 0 | 0% |
| Выборочная медиана | 1 | 1 | 0 | 0% |
| Выборочный коэффициент асимметрии | 1.2885 | 0.8671 | 0.4214 | 48.5987% |
| Выборочный коэффициент эксцесса | 2.0822 | 0.7518 | 1.3304 | 76.9619% |

{|wi-pi|}=[0.0044, 0.0215, 0.0272, 0.0037, 0.0011, 0.0025, 0.0179]

max{|wi-pi|, i=1,…,m}=0.0179

**Литература по математической статистике**

1. Гмурман В.Е. Теория вероятностей и математическая статистика: Учеб. пособие

для вузов — М.: Высш. образов., 2006. — 480 с.

2. Математическая статистика [Электронный ресурс]: метод. указания по выполнению лаб. работ / А.А.Лобузов — М.: МИРЭА, 2017. — Электрон. опт. диск (ISO)

3. Справочное пособие по теории вероятностей и математической статистике

(законы распределения): Учеб.пособие для вузов / Г.А.Соколов, Н.А. Чистякова. —

М.: Высш. шк., 2007. — 248 с.

**Приложение**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261  262  263  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278  279  280  281  282  283  284  285  286  287  288  289  290  291  292  293  294  295  296  297  298  299  300  301  302  303  304  305  306  307  308  309  310  311  312  313  314  315  316  317  318  319  320  321  322  323  324  325  326  327  328  329 | *#!/usr/bin/python*  *# -\*- coding: UTF-8 -\*-*  import sys  import argparse  from collections import Counter  from itertools import islice  from functools import reduce  from scipy import stats as st  import numpy as np  from glob import glob  import pickle  import re  import matplotlib.pyplot as plt  variant = 9  def createParser():  parser = argparse.ArgumentParser()  parser.add\_argument('-w', '--write', default=0, type=int)  parser.add\_argument('-r', '--read', default=1, type=int)  return parser  def save(r, type\_ryad):  reg = 'data/' + type\_ryad + '\*'  files = glob(reg) *# type: ryadN.pickle*  if files:  n = int(re.sub(r'[^\d]', '', files[0]))  n += 1  else:  n = 0  filename = reg[:-1] + str(n) + '.pickle'  with open(filename, 'wb') as f:  pickle.dump(r, f)  def load(type\_ryad, n = -1):  reg = 'data/' + type\_ryad + '\*'  files = glob(reg)  if not files:  return  if abs(n) > len(files):  return  with open(files[n], 'rb') as f:  return(pickle.load(f))  def draw\_poligon(count, pr, name, sizex=5, sizey=5):  X = list(count.keys())  Y = [count[x][1] for x in X]  fig, ax = plt.subplots(figsize=(sizex, sizey))  ax.plot(X, Y, label='относительные частоты')  ax.plot(X, pr, label='вероятности')  ax.set\_title('Полигон относительных частот')  ax.legend(loc='upper left')  ax.set\_ylabel('y')  ax.set\_xlabel('x')  ax.set\_xlim(xmin=0, xmax=max(X))  ax.set\_ylim(ymin=0, ymax=max(Y) + 0.1)  if sizex != 5:  *# Don't allow the axis to be on top of your data*  ax.set\_axisbelow(True)  *# Turn on the minor TICKS, which are required for the minor GRID*  ax.minorticks\_on()  *# Customize the major grid*  ax.grid(which='major', linestyle='-', linewidth='0.5', color='red')  *# Customize the minor grid*  ax.grid(which='minor', linestyle=':', linewidth='0.5', color='black')  else:  ax.grid()  fig.tight\_layout()  fig.savefig('data/poligon\_' + name + '.png')  def draw\_cdf(Xlist, Ylist, name, sizex=5, sizey=5):  fig, ax = plt.subplots(figsize=(sizex, sizey))  for X, Y in zip(Xlist, Ylist):  ax.plot(X, Y, label='')  ax.set\_title('Эмпирическая функция распределения')  ax.legend(loc='upper left')  ax.set\_ylabel('y')  ax.set\_xlabel('x')  ax.set\_xlim(xmin=0, xmax=max(X))  ax.set\_ylim(ymin=0, ymax=1)  fig.tight\_layout()  if sizex != 5:  *# Don't allow the axis to be on top of your data*  ax.set\_axisbelow(True)  *# Turn on the minor TICKS, which are required for the minor GRID*  ax.minorticks\_on()  *# Customize the major grid*  ax.grid(which='major', linestyle='-', linewidth='0.5', color='red')  *# Customize the minor grid*  ax.grid(which='minor', linestyle=':', linewidth='0.5', color='black')  else:  ax.grid()  fig.savefig('data/cdf\_' + name + '.png')  def expect(nv, key):  if key in nv.keys():  return nv[key][1]  else:  return 0.0  *# elif key not in nv.keys() and key != 0:*  *# return expect(nv, key-1)*  *# else:*  *# return 0.0*  def generate\_ef(nv, r):  delta = 0.01  items = list(zip(list(range(max(r) + 2)), islice(list(range(max(r) + 2)), 1, None))) *# [(0, 1), (1, 2), ...]*  keys = list(range(list(nv.keys())[-1] + 1)) *# list(nv.keys())*  Y = [sum([expect(nv, key) for key in keys[:i + 1]]) for i, el in enumerate(keys)]  Xlist = [[x \* delta for x in range(item[0] \* int(1 / delta), item[1] \* int(1 / delta))]  for item in items]  Ylist = [[y] \* len(Xlist[i]) for i, y in enumerate(Y)]  *# Ylist = [[st.binom.cdf(x, n, p) for x in line] for line in Xlist]*  return Xlist, Ylist, Y  def exp\_stats(nv, r, st\_r):  mean = sum([key \* val[1] for key, val in nv.items()]) *# mean*  var = sum([(key - mean) \*\* 2 \* val[1] for key, val in nv.items()])  standart = var \*\* 0.5  mu = lambda k: sum([(key - mean) \*\* k \* val[1] for key, val in nv.items()])  skew = mu(3) / (standart \*\* 3)  kurtosis = mu(4) / (standart \*\* 4) - 3  mode = st.mode(r)  if len(st\_r) % 2 != 0:  med = st\_r[int(len(st\_r) / 2)]  else:  med = 0.5 \* (st\_r[int(len(st\_r) / 2)] + st\_r[int(len(st\_r) / 2) - 1])  return mean, var, standart, skew, kurtosis, mode, med  def binomial(np, size, write, read):  n, p = np  r = load('binomial') if read else st.binom.rvs(n, p, size=size)  if write:  save(r, 'binomial')  st\_r = sorted(r)  p\_ch = Counter(r)  nv = {key: (val, val / sum(p\_ch.values())) for key, val in p\_ch.items()}  ex\_mean, ex\_var, ex\_standart, ex\_skew, ex\_kurtosis, ex\_mode, ex\_med = exp\_stats(nv, r, st\_r)  pr = [st.binom.pmf(x, n, p) for x in list(nv.keys())]  draw\_poligon(nv, pr, 'binomial')  *# delta = 0.01*  *# items = list(zip(list(range(max(r) + 2)), islice(list(range(max(r) + 2)), 1, None))) # [(0, 1), (1, 2), ...]*  *# Xlist = [[x \* delta for x in range(item[0] \* int(1 / delta), item[1] \* int(1 / delta))]*  *# for item in items]*  *# Ylist = [[st.binom.cdf(x, n, p) for x in line] for line in Xlist]*  Xlist, Ylist, Y = generate\_ef(nv, r)  draw\_cdf(Xlist, Ylist, 'binomial')  cdf\_func = [el[0] for el in Ylist]  *# cdf\_func.append(1.0)*  *# cdf\_func.insert(0, 0.0)*  mean, variance, skew, kurtosis = st.binom.stats(n, p, moments='mvsk') *# среднее, дисперсия, ассиметрия, эксцесс*  standart, med = st.binom.std(n, p), st.binom.median(n, p) *# стантартное отклонение, медиана*  mode = st.mode(r) *# мода*  pmf = [abs(val[1] - st.binom.pmf(key, n, p)) for key, val in nv.items()]  return (r, st\_r, cdf\_func, nv, float(mean),  float(variance),  float(skew),  float(kurtosis),  standart,  med,  float(mode.mode), ex\_mean,  ex\_var,  ex\_standart,  ex\_skew,  ex\_kurtosis,  float(ex\_mode.mode),  ex\_med, pmf, Y)  def geometr(p, size, write, read):  r = load('geometr') if read else st.geom.rvs(p, loc=-1, size=size)  if write:  save(r, 'geometr')  st\_r = sorted(r)  p\_ch = Counter(r)  nv = {key: (val, val / sum(p\_ch.values())) for key, val in p\_ch.items()}  ex\_mean, ex\_var, ex\_standart, ex\_skew, ex\_kurtosis, ex\_mode, ex\_med = exp\_stats(nv, r, st\_r)  pr = [st.geom.pmf(x, p, loc=-1) for x in list(nv.keys())]  draw\_poligon(nv, pr, 'geometr', 30, 30)  *# delta = 0.01*  *# items = list(zip(list(range(max(r) + 2)), islice(list(range(max(r) + 2)), 1, None))) # [(0, 1), (1, 2), ...]*  *# Xlist = [[x \* delta for x in range(item[0] \* int(1 / delta), item[1] \* int(1 / delta))]*  *# for item in items]*  *# Ylist = [[st.geom.cdf(x, p) for x in line] for line in Xlist]*  Xlist, Ylist, Y = generate\_ef(nv, r)  draw\_cdf(Xlist, Ylist, 'geometr', 30, 30)  cdf\_func = [el[0] for el in Ylist]  cdf\_func.append(1.0)  cdf\_func.insert(0, 0.0)  mean, variance, skew, kurtosis = st.geom.stats(p, loc=-1, moments='mvsk') *# среднее, дисперсия, ассиметрия, эксцесс*  standart, med = st.geom.std(p, loc=-1), st.geom.median(p, loc=-1) *# стантартное отклонение, медиана*  mode = st.mode(r) *# мода*  pmf = [abs(val[1] - st.geom.pmf(key, p, loc=-1)) for key, val in nv.items()]  return (r, st\_r, cdf\_func, nv, float(mean),  float(variance),  float(skew),  float(kurtosis),  standart,  med,  float(mode.mode), ex\_mean,  ex\_var,  ex\_standart,  ex\_skew,  ex\_kurtosis,  float(ex\_mode.mode),  ex\_med, pmf, Y)  def puasson(mu, size, write, read):  r = load('puasson') if read else st.poisson.rvs(mu, size=size)  if write:  save(r, 'puasson')  st\_r = sorted(r)  p\_ch = Counter(r)  nv = {key: (val, val / sum(p\_ch.values())) for key, val in p\_ch.items()}  ex\_mean, ex\_var, ex\_standart, ex\_skew, ex\_kurtosis, ex\_mode, ex\_med = exp\_stats(nv, r, st\_r)  pr = [st.poisson.pmf(x, mu) for x in list(nv.keys())]  draw\_poligon(nv, pr, 'puasson')  *# delta = 0.01*  *# items = list(zip(list(range(max(r) + 2)), islice(list(range(max(r) + 2)), 1, None))) # [(0, 1), (1, 2), ...]*  *# Xlist = [[x \* delta for x in range(item[0] \* int(1 / delta), item[1] \* int(1 / delta))]*  *# for item in items]*  *# Ylist = [[st.poisson.cdf(x, mu) for x in line] for line in Xlist]*  Xlist, Ylist, Y = generate\_ef(nv, r)  draw\_cdf(Xlist, Ylist, 'puasson')  cdf\_func = [el[0] for el in Ylist]  cdf\_func.append(1.0)  cdf\_func.insert(0, 0.0)  mean, variance, skew, kurtosis = st.poisson.stats(mu, moments='mvsk') *# среднее, дисперсия, ассиметрия, эксцесс*  standart, med = st.poisson.std(mu), st.poisson.median(mu) *# стантартное отклонение, медиана*  mode = st.mode(r) *# мода*  pmf = [abs(val[1] - st.poisson.pmf(key, mu)) for key, val in nv.items()]  return (r, st\_r, cdf\_func, nv, float(mean),  float(variance),  float(skew),  float(kurtosis),  standart,  med,  float(mode.mode), ex\_mean,  ex\_var,  ex\_standart,  ex\_skew,  ex\_kurtosis,  float(ex\_mode.mode),  ex\_med, pmf, Y)  def save\_result(result):  reg = 'data/result\*'  files = glob(reg)  if files:  n = int(re.sub(r'[^\d]', '', files[0]))  n += 1  else:  n = 0  filename = reg[:-1] + str(n) + '.txt'  with open(filename, 'w') as f:  f.write(result)  if \_\_name\_\_ == '\_\_main\_\_':  parser = createParser()  namespace = parser.parse\_args(sys.argv[1:])  result = ['выборка: {0}',  'упорядоченная: {1}',  'эмпирическая функция распределения: {2}',  'статистический ряд: {3}',  'теор. среднее: {4} <=> среднее: {11}',  'теор. дисперсия: {5} <=> дисперсия: {12}',  'теор. ассиметрия: {6} <=> ассиметрия: {14}',  'теор. эксцесс: {7} <=> эксцесс: {15}',  'теор. среднее квадратичное отклонение: {8} <=> среднее квадратичное отклонение: {13}',  'теор. медиана: {9} <=> медиана: {17}',  'теор. мода: {10} <=> мода: {16}',  'pmf: {18}',  'ex\_pmf {19}']  line = '\n\n'.join(result)  task = {'binomial': (binomial, [5 + variant % 17, 0.1 + 0.01 \* variant]),  'geometr': (geometr, 0.1 + 0.01 \* variant),  'puasson': (puasson, 0.7 + 0.07 \* variant)}  out = ''  for r\_type, fdata in task.items():  func, params = fdata  res = func(params, size=150,  write = namespace.write,  read = namespace.read)  out += r\_type + '\n\n' + line.format(\*res) + '\n\n<' + '='\*50 + '>\n\n'  save\_result(out) |