# Министерство образования и науки Российской Федерации

## Федеральное государственное бюджетное образовательное учреждение высшего образования

## «Новосибирский государственный технический университет»

NSTU_Logo_blue

## Кафедра теоретической и прикладной информатики

### Лабораторная работа № 3 по дисциплине «Компьютерное моделирование»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| сигма градиент синий1 | Факультет: | ПМИ |  |  |
| Группа: | ПМИ-62 |  |  |
| Студенты: | Ершов П.К., Мамонова Е.В., Цыденов З.Б. |  |  |
| Вариант: | 9 |  |  |
| Преподаватель: | Волкова В.М. |  |  |

Новосибирск

2019

1. **Цель работы**

Научиться моделировать значения дискретно распределённой случайной величины и проводить статистический анализ сгенерированных данных.

1. **Задание**

| **№** | **Алгоритм** | **Закон распределения** | **Параметры распределений** | **Параметры распределения Пуассона (для нестандартного алгоритма)** |
| --- | --- | --- | --- | --- |
| **9** | **Стандартный** | **Отрицательный биномиальный** | **s = 1,  p = 0.2; s = 1,  p = 0.5; s = 1,  p = 0.8** | **λ = 18** |

1. **Описание формата входного файла**

В файле data.txt находятся входные данные для всех тестов.

41 – коэффициент a

51 – коэффициент b

13 – коэффициент c

1.0001 – число m, остатком от которого является псевдослучайное число

1 – текущее значение xn в алгоритме псевдослучайных чисел из второй лабораторной

40 – максимальное количество чисел N в последовательности

100 – максимальное количество чисел N в последовательности

18 – параметр λ распределения Пуассона.

1 – s параметр распределения (число успехов)

0.2 – p параметр распределения (начальная вероятность успеха)

0.5 – p параметр распределения (начальная вероятность успеха)

0.8 – p параметр распределения (начальная вероятность успеха)

0.05 – уровень значимости α.

1. **Ход работы**

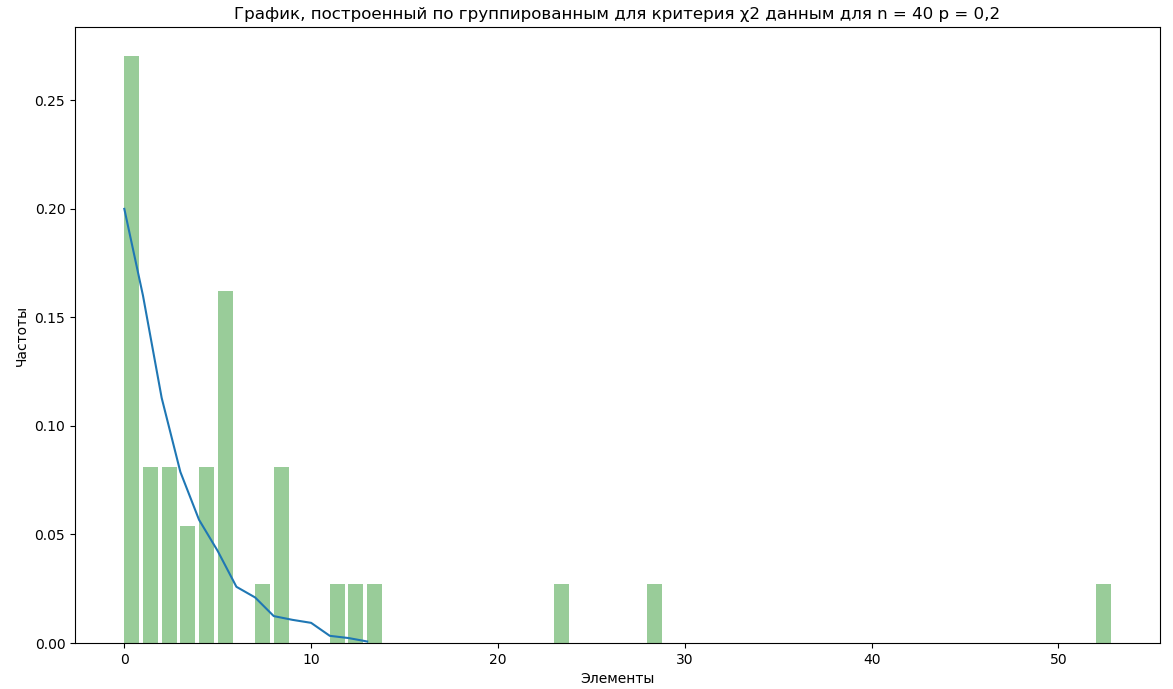
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Стандартный алгоритм с отрицательным биномиальным законом распределения** | | | | | | |
| **Длина последовательности** | 40 | | | 100 | | |
| **Значение P** | 0.2 | 0.5 | 0.8 | 0.2 | 0.5 | 0.8 |
| **Последовательность** | 8, 0, 3, 5, 0, 52, 0, 23, 2, 3, 0, 0, 4, 4, 1, 2, 4, 1, 5, 8, 5, 7, 0, 0, 5, 1, 0, 8, 5, 0, 28, 12, 13, 0, 2, 5, 11 | 3, 1, 3, 0, 1, 1, 0, 2, 3, 0, 2, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 2, 2, 2, 0, 0, 1, 2 | 1, 0, 1, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 | 8, 0, 3, 5, 0, 52, 0, 23, 2, 3, 0, 0, 4, 4, 1, 2, 4, 1, 5, 8, 5, 7, 0, 0, 5, 1, 0, 8, 5, 0, 28, 12, 13, 0, 2, 5, 11, 10, 1, 0, 0, 16, 1, 4, 1, 3, 0, 0, 9, 2, 6, 0, 0, 11, 30, 3, 5, 2, 8, 0, 149, 11, 1, 0, 0, 19, 1, 1, 0, 0, 60, 11, 0, 0, 0, 2, 2, 5, 9, 1, 1, 0, 10, 6, 1, 4, 0, 7, 2, 4, 6, 9, 9, 0, 6 | 3, 1, 3, 0, 1, 1, 0, 2, 3, 0, 2, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 2, 2, 2, 0, 0, 1, 2, 2, 0, 0, 0, 2, 0, 1, 0, 1, 3, 0, 0, 1, 0, 1, 0, 0, 2, 2, 1, 1, 0, 1, 0, 3, 2, 0, 0, 0, 2, 0, 0, 0, 0, 3, 2, 0, 0, 3, 0, 0, 0, 1, 1, 0, 0, 0, 2, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1 | 1, 0, 1, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 |
| **Число операций** | 16526 | 162 | 84 | 21492 | 312 | 158 |
| **Значение статистики S** | 78.103 | 3.416 | 0.656 | 180.26 | 2.555 | 3.862 |
| **Достигнутый уровень значимости** | 0.00084 | 0.49 | 0.720 | 9.944e-19 | 0.634 | 0.145 |
| **Результаты проверки критерия *χ*2** | Не пройден | Пройден | Пройден | Не пройден | Пройден | Пройден |
| **Вывод о качестве сгенерированной последовательности** | Является не очень качественной, так как не равномерная | Можно считать качественной, так как, возможно, равномерная | Является не очень качественной, так как, хотя критерий и пройден, один и тот же элемент встречается чаще других | Является не очень качественной, так как не равномерная | Можно считать качественной, так как, возможно, равномерная | Является не очень качественной, так как, хотя критерий и пройден, один и тот же элемент встречается чаще других |

|  |  |  |
| --- | --- | --- |
| **Стандартный алгоритм с законом распределения Пуассона** | | |
| **Длина последовательности** | 40 | 100 |
| **Последовательность** | 22, 19, 22, 13, 18, 18, 13, 21, 23, 14, 21, 17, 18, 13, 13, 18, 18, 15, 17, 18, 16, 18, 19, 18, 19, 10, 12, 18, 15, 14, 19, 18, 13, 21, 20, 20, 14, 17, 18, 20 | 22, 19, 22, 13, 18, 18, 13, 21, 23, 14, 21, 17, 18, 13, 13, 18, 18, 15, 17, 18, 16, 18, 19, 18, 19, 10, 12, 18, 15, 14, 19, 18, 13, 21, 20, 20, 14, 17, 18, 20, 20, 16, 11, 14, 20, 15, 18, 16, 18, 23, 13, 14, 19, 17, 19, 14, 13, 20, 21, 18, 18, 17, 19, 10, 22, 20, 15, 7, 14, 21, 16, 15, 8, 11, 22, 20, 14, 13, 23, 12, 17, 17, 18, 19, 16, 15, 14, 20, 18, 16, 18, 14, 19, 16, 18, 19, 19, 19, 10, 18 |
| **Число операций** | 308 | 874 |
| **Значение статистики S** | 21.672 | 26.377 |
| **Достигнутый уровень значимости** | 0.6 | 0.334 |
| **Результаты проверки критерия *χ*2** | Пройден | Пройден |
| **Вывод о качестве сгенерированной последовательности** | Можно считать качественной, так как, возможно, равномерная | Можно считать качественной, так как, возможно, равномерная |

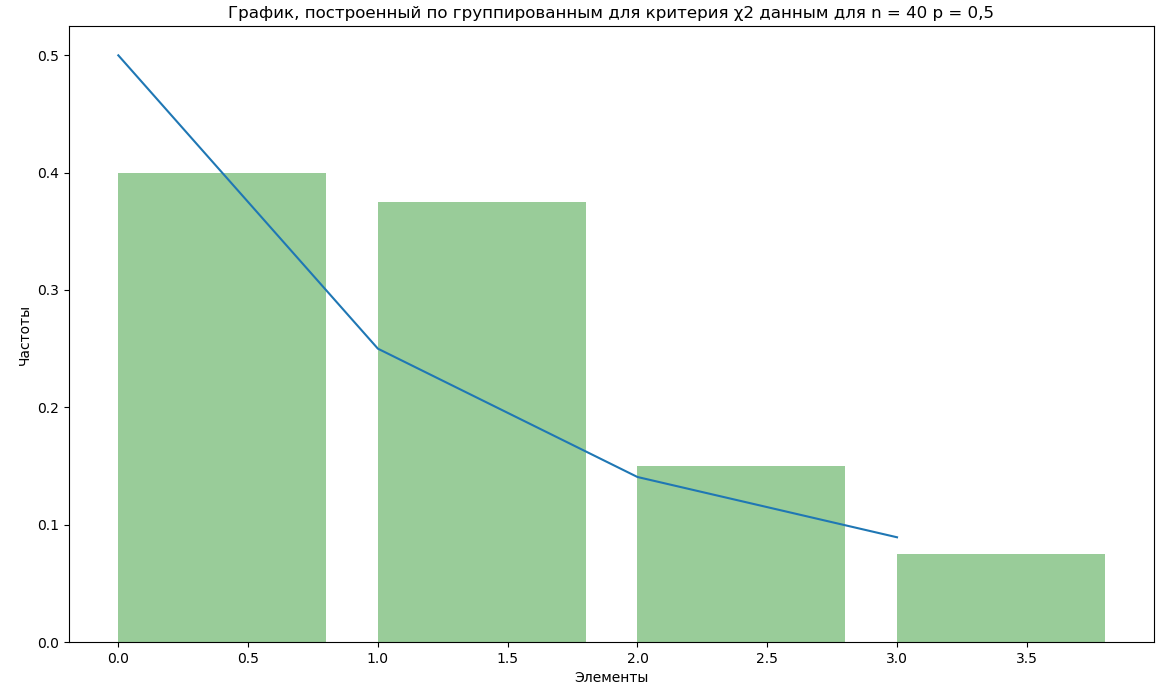
**Графики стандартного алгоритма с отрицательным биномиальным законом.**

**Для N = 40.**

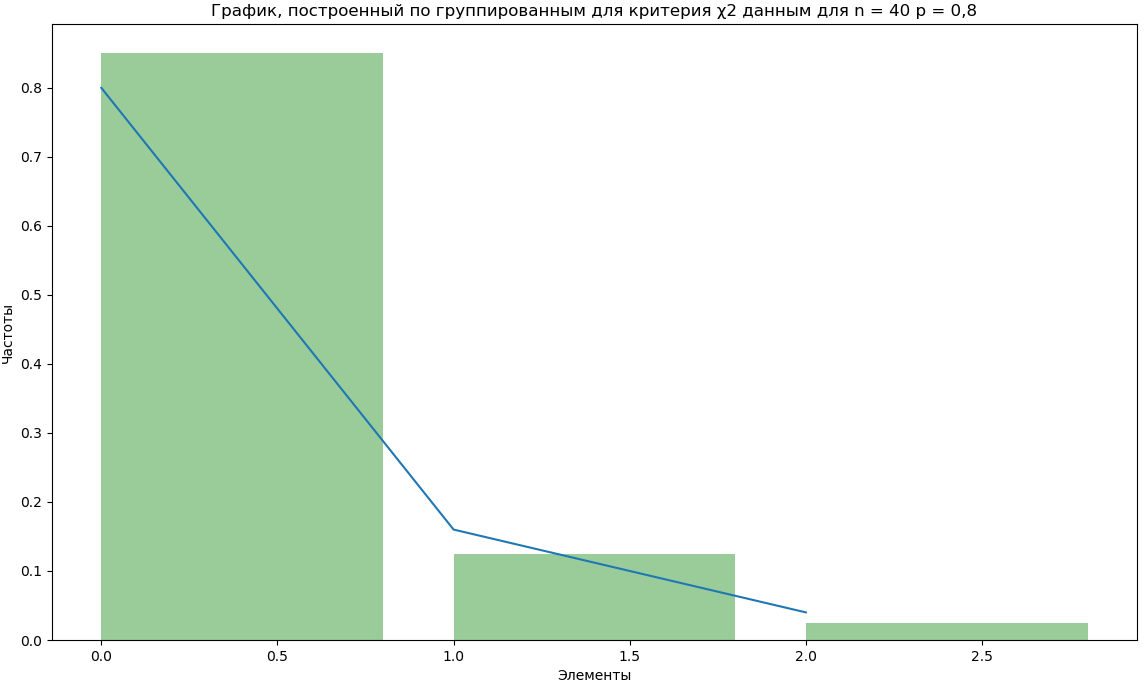
**P = 0.2**



**P = 0.5**

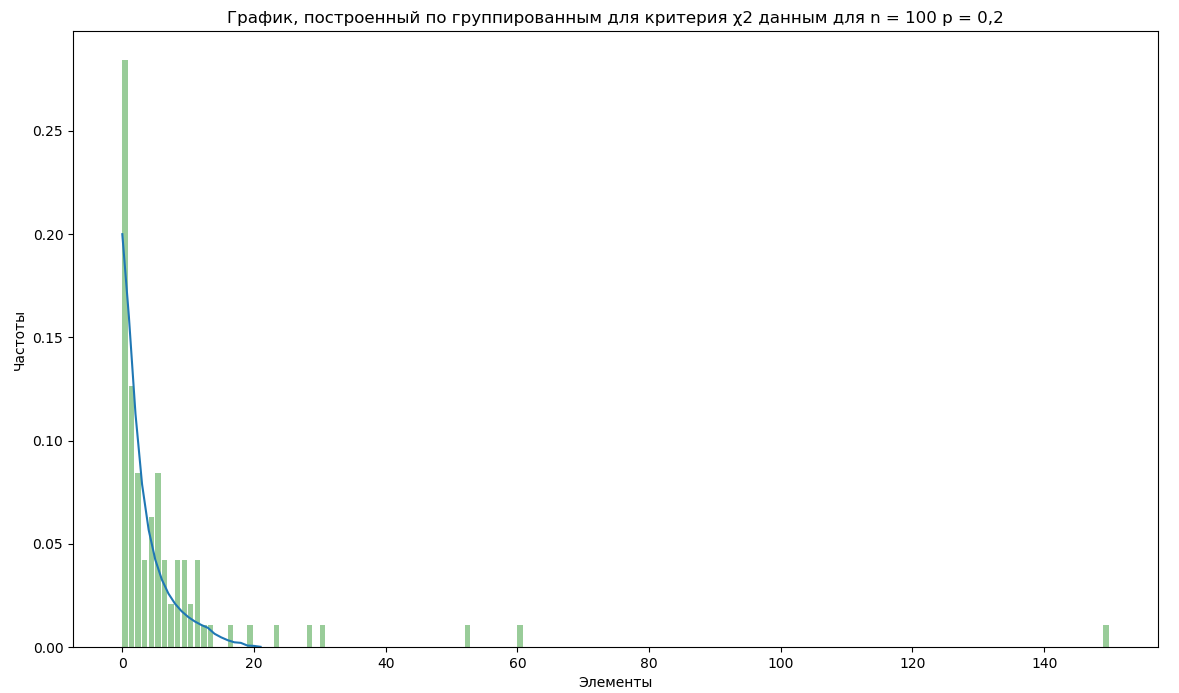


**P = 0.8**

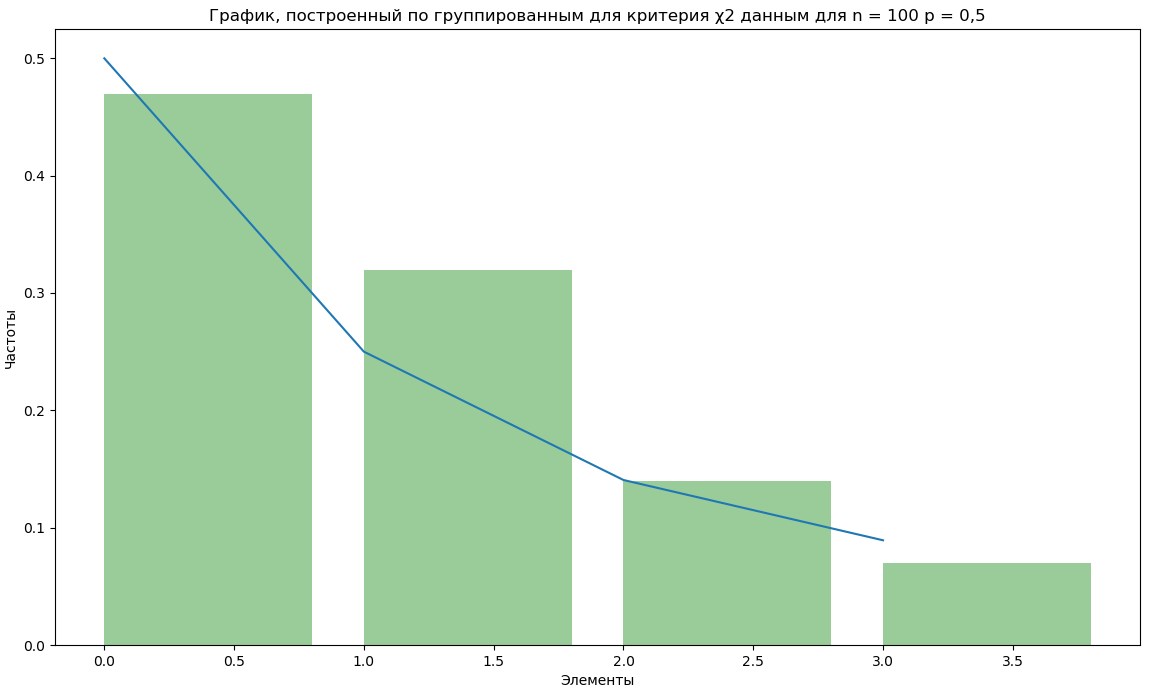


**Для N = 100.**

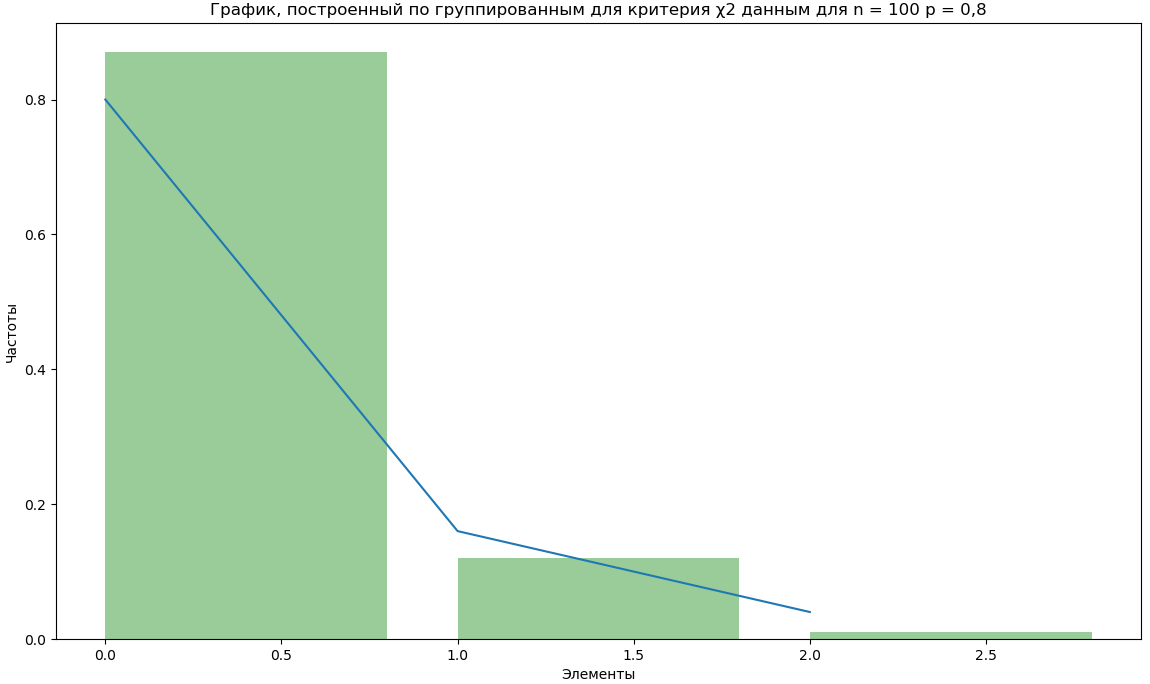
**P = 0.2**



**P = 0.5**



**P = 0.8**



1. **Код программы**

**import** math **as** mh  
**import** pandas **as** pd  
**import** numpy **as** np  
**import** matplotlib.pyplot **as** plt  
**from** pylab **import** \*  
**import** scipy  
**import** scipy.stats  
**from** scipy **import** integrate  
**from** scipy.stats **import** gamma  
  
pd.set\_option(**'display.max\_columns'**, **None**)  
  
  
**def** read\_data(): *# считывание данных из файла* data = []  
 **with** open(**"data.txt"**) **as** f:  
 **for** line **in** f:  
 data.append([float(x) **for** x **in** line.split()])  
 **return** data  
  
  
**def** seq\_gen(a, b, c, x\_t, N, m): *# генератор псевдослучайной последовательности* x = []  
 x\_n = 0  
 x\_p = 0  
  
 **for** i **in** range(N):  
 x\_n = (a \* x\_t + b \* x\_p + c) % m  
 x\_p = x\_t  
 x\_t = x\_n  
 x.insert(i, x\_n)  
 **return** x  
  
  
**def** C(n, k): *# сочетания из n по k* c = mh.factorial(n) / ((mh.factorial(k)) \* mh.factorial(n - k))  
 **return** c  
  
  
**def** Negative\_binomial(s,  
 p): *# генерация последовательности вероятностей по негативному биномиальному закону распределения* P = []  
 ef = 0  
 out = []  
 k = 0  
 deg\_f = 0  
 buf = C(s + k - 1, k) \* p \*\* s \* (1 - p) \*\* k  
 **if** buf >= 0.001:  
 deg\_f += 1  
 P.insert(0, buf)  
 ef += 10  
 **while** sum(P) <= 1:  
 **if** k > 1000:  
 **break** k += 1  
 buf = C(s + k - 1, k) \* P[k - 1] \*\* s \* ((1 - P[k - 1]) \*\* k)  
 **if** buf >= 0.001:  
 deg\_f += 1  
 P.insert(k, buf)  
 ef += 10  
 **if** sum(P) > 1:  
 **for** i **in** range(len(P) - 1):  
 out.insert(i, P[i])  
 out.insert(len(P), 1 - sum(out))  
 **return** out, ef, deg\_f  
 **return** P, ef, deg\_f  
  
  
**def** s\_algorithm(a, b, c, x, m, n, s, p): *# стандартный алгоритм получения дискретно распределённых случайных величин* seq = [] *# последовательность* ro = seq\_gen(a, b, c, x, n, m) *# набор псевдослучайных чисел* deg\_f = 0 *# степень свободы (количество вероятностей P >= 0.001)* ef = 0 *# эффективность алгоритма (число операций)* P, ef, deg\_f = Negative\_binomial(s, p) *# получаем вероятности случайных величин* **for** i **in** range(n):  
 M = ro[i]  
 j = 0  
 **while** j < len(P):  
 M -= P[j]  
 **if** M >= 0:  
 j += 1  
 ef += 1  
 **else**:  
 seq.insert(i, j)  
 **break** ef += 1  
 ef += 1  
 **return** seq, ef, P, deg\_f  
  
  
**def** Poisson\_dist(lda): *# генерация последовательности вероятностей по закону распределения Пуассона* P = []  
 Q = 0  
 k = 0  
 deg\_f = 0 *# степень свободы (количество вероятностей P >= 0.001)* buf = (lda \*\* k \* mh.e \*\* (-lda)) / mh.factorial(k)  
 **if** buf >= 0.001:  
 deg\_f += 1  
 P.insert(0, buf)  
 ef = 5 *# эффективность алгоритма (число операций)* **while** sum(P) <= 1:  
 k += 1  
 buf = (lda \*\* k \* mh.e \*\* (-lda)) / mh.factorial(k)  
 **if** buf >= 0.001:  
 deg\_f += 1  
 P.insert(k, buf)  
 **if** k == 18:  
 Q = sum(P)  
 ef += 2  
 ef += 6  
 **return** P, Q, ef, deg\_f  
  
  
**def** un\_algorithm(a, b, c, x, m, n,  
 lda): *# не стандартный алгоритм получения дискретно распределённых случайных величин (распределение Пуассона)* seq = []  
 ro = seq\_gen(a, b, c, x, n, m) *# набор псевдослучайных чисел* deg\_f = 0 *# степень свободы (количество вероятностей P >= 0.001)* ef = 0 *# эффективность алгоритма (число операций)* P, Q, ef, deg\_f = Poisson\_dist(lda) *# получаем вероятности случайных величин* ef = 0 *# эффективность алгоритма (число операций)  
 # print(P)* **for** i **in** range(n):  
 ef += 1  
 M = ro[i] - Q  
 *# print(M)* j = lda  
 **if** M < 0:  
 **while** 1:  
 M += P[j]  
 **if** M < 0:  
 j -= 1  
 ef += 2  
 **else**:  
 ef += 1  
 seq.insert(i, j)  
 **break** ef += 1  
 **else**:  
 **while** 1:  
 M -= P[j]  
 **if** M >= 0:  
 j += 1  
 ef += 1  
 **else**:  
 seq.insert(i, j)  
 **break** ef += 1  
 **return** seq, ef, P, deg\_f  
  
  
**def** find(x, y): *# проверка принадлежности элемента y выборке x* **for** i **in** range(len(x)):  
 **if** y == x[i]:  
 **return** 1  
 **return** 0  
  
  
**def** unique(x): *# выделение последовательности уникальных элементов из выборки x* un\_seq = [] *# набор уникальных элементов последовательности* un\_seq.insert(0, x[0])  
 i = 1  
 j = 1  
 **for** i **in** range(len(x)):  
 fl = find(un\_seq, x[i])  
 **if** fl == 0:  
 un\_seq.insert(j, x[i])  
 j += 1  
 **return** un\_seq  
  
  
**def** P\_crit(Sk, r): *# Вычисление значения статистики* coef = 1 / (np.power(2, (r / 2)) \* mh.gamma(r / 2))  
 func = **lambda** S: np.power(S, (r / 2 - 1)) \* (np.power(mh.e, (- S / 2)))  
 P = integrate.quad(func, Sk, np.inf)  
 **return** P[0] \* coef  
  
  
**def** f\_it(x, xi): *# посчёт числа встречи элемента xi в выборке x* n = 0  
 **for** i **in** range(len(x)):  
 **if** x[i] == xi:  
 n += 1  
 **return** n  
  
  
*# Проверка гипотезы по критерию хи квадрат***def** hi\_2(x, alpha, N, p, lda, P, deg\_f):  
 print(**'Проверка критерия χ2:\n'**)  
 sum = 0 *# Значение статистики S* v = []  
  
 *# определяем число интервалов K* un\_seq = unique(x) *# массив уникальных элементов последовательности* K = deg\_f  
  
 un\_sort = sorted(un\_seq) *# сортировка уникальных элементов по возростанию  
  
 # подсчет частоты встречи уникльных элементов в последовательности* **for** i **in** range(len(un\_seq)):  
 v.append(f\_it(x, un\_sort[i]))  
 print(**'Уникальные элементы'**, un\_sort)  
 print(**"V = "**, v)  
  
 v\_n = []  
 **for** i **in** range(len(v)):  
 v\_n.append(v[i] / len(x)) *# отностельные частоты элементов* print(**"V = "**, v\_n)  
 P\_theor = [] *# теоритическая вероятность* **for** i **in** un\_sort:  
 P\_theor.append(i)  
 P\_u = [] *# полученная вероятность встречи элемента в последовательности* **for** i **in** un\_sort:  
 P\_u.append(P[i])  
 print(P\_u)  
 *# график, построенный по группированным для критерия χ2 данным* fig = plt.figure(figsize=(14, 8))  
 ax = fig.add\_subplot()  
 plt.bar(un\_sort, v\_n, align=**'edge'**, color=**'green'**, alpha=0.4) *# Гистограмма* ax.plot(P\_theor[:len(v)], P\_u[:len(v)]) *# График* plt.xlabel(**'Элементы'**)  
 plt.ylabel(**'Частоты'**)  
 **if** N == 40:  
 **if** p == 0.2:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 40 p = 0,2'**)  
 **if** p == 0.5:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 40 p = 0,5'**)  
 **if** p == 0.8:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 40 p = 0,8'**)  
 **else**:  
 **if** p == 0.2:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 100 p = 0,2'**)  
 **if** p == 0.5:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 100 p = 0,5'**)  
 **if** p == 0.8:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 100 p = 0,8'**)  
 **if** lda != -1:  
 **if** N == 40:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 40'**)  
 **else**:  
 ax.set\_title(**'График, построенный по группированным для критерия χ2 данным для n = 100'**)  
 plt.show()  
  
 *# вычисляем сумму в формуле S(hi2)* **if** deg\_f > len(un\_seq):  
 d = len(un\_seq)  
 **else**:  
 d = deg\_f  
 **for** i **in** range(d):  
 sum += mh.pow(((v[i] / len(x)) - P\_u[i]), 2) / P\_u[i]  
 S = len(x) \* sum *# умножаем сумму на объем выборки n* print(**'S = '**, S)  
 r = K - 1 *# число степеней свободы* print(**"R = "**, r)  
 S\_star = P\_crit(S, r)  
 **if** S\_star > alpha:  
 print(**'P {Sχ2 > Sχ2\*}(достигнутый уровень значимости) = '**, S\_star)  
 print(**'Проверка критерия χ2 окончена.\n'**)  
 **return** 1, S, S\_star *# гипотеза отвергается* **else**:  
 print(**'P {Sχ2 > Sχ2\*}(достигнутый уровень значимости) = '**, S\_star)  
 print(**'Проверка критерия χ2 окончена.\n'**)  
 **return** 0, S, S\_star *# гипотеза не отвергается*data = read\_data()  
  
a = int(data[0][0])  
b = int(data[1][0])  
c = int(data[2][0])  
m = data[3][0]  
x = int(data[4][0])  
n1 = int(data[5][0])  
n2 = int(data[6][0])  
lda = int(data[7][0])  
s = int(data[8][0])  
p1 = data[9][0]  
p2 = data[10][0]  
p3 = data[11][0]  
alpha = data[12][0]  
  
print(**"Стаднартный алгоритм"**)  
print(**"N = 40"**)  
seq\_n40\_1, ef40\_1, P40\_1, deg40\_1 = s\_algorithm(a, b, c, x, m, n1, s, p1)  
h40\_1, s40\_1, st40\_1 = hi\_2(seq\_n40\_1, 0.05, n1, p1, -1, P40\_1, deg40\_1)  
print(**"P = 0,2: "**, seq\_n40\_1, **'Число действий'**, ef40\_1, **'Вероятности'**, P40\_1)  
print(**"Hi^2: "**, h40\_1, s40\_1, st40\_1)  
  
seq\_n40\_2, ef40\_2, P40\_2, deg40\_2 = s\_algorithm(a, b, c, x, m, n1, s, p2)  
h40\_2, s40\_2, st40\_2 = hi\_2(seq\_n40\_2, 0.05, n1, p2, -1, P40\_2, deg40\_2)  
print(**"P = 0,5: "**, seq\_n40\_2, **'Число действий'**, ef40\_2, **'Вероятности'**, P40\_2)  
print(**"Hi^2: "**, h40\_2, s40\_2, st40\_2)  
  
seq\_n40\_3, ef40\_3, P40\_3, deg40\_3 = s\_algorithm(a, b, c, x, m, n1, s, p3)  
h40\_3, s40\_3, st40\_3 = hi\_2(seq\_n40\_3, 0.05, n1, p3, -1, P40\_3, deg40\_3)  
print(**"P = 0,8: "**, seq\_n40\_3, **'Число действий'**, ef40\_3, **'Вероятности'**, P40\_3)  
print(**"Hi^2: "**, h40\_3, s40\_3, st40\_3)  
  
print(**"N = 100"**)  
seq\_n100\_1, ef100\_1, P100\_1, deg100\_1 = s\_algorithm(a, b, c, x, m, n2, s, p1)  
h100\_1, s100\_1, st100\_1 = hi\_2(seq\_n100\_1, 0.05, n2, p1, -1, P100\_1, deg100\_1)  
print(**"P = 0,2: "**, seq\_n100\_1, **'Число действий'**, ef100\_1, **'Вероятности'**, P100\_1)  
print(**"Hi^2: "**, h100\_1, s100\_1, st100\_1)  
  
seq\_n100\_2, ef100\_2, P100\_2, deg100\_2 = s\_algorithm(a, b, c, x, m, n2, s, p2)  
h100\_2, s100\_2, st100\_2 = hi\_2(seq\_n100\_2, 0.05, n2, p2, -1, P100\_2, deg100\_2)  
print(**"P = 0,5: "**, seq\_n100\_2, **'Число действий'**, ef100\_2, **'Вероятности'**, P100\_2)  
print(**"Hi^2: "**, h100\_2, s100\_2, st100\_2)  
  
seq\_n100\_3, ef100\_3, P100\_3, deg100\_3 = s\_algorithm(a, b, c, x, m, n2, s, p3)  
h100\_3, s100\_3, st100\_3 = hi\_2(seq\_n100\_3, 0.05, n2, p3, -1, P100\_3, deg100\_3)  
print(**"P = 0,8: "**, seq\_n100\_3, **'Число действий'**, ef100\_3, **'Вероятности'**, P100\_3)  
print(**"Hi^2: "**, h100\_3, s100\_3, st100\_3)  
  
print(**"Нестаднартный алгоритм"**)  
print(**"N = 40"**)  
seq\_n40, ef40, P40, deg40 = un\_algorithm(a, b, c, x, m, n1, lda)  
h40, s40, st40 = hi\_2(seq\_n40, 0.05, n1, p1, lda, P40, deg40)  
print(**'Последовательность'**, seq\_n40, **'Число действий'**, ef40, **'Вероятности'**, P40)  
print(**"Hi^2: "**, h40, s40, st40)  
  
print(**"N = 100"**)  
seq\_n100, ef100, P100, deg100 = un\_algorithm(a, b, c, x, m, n2, lda)  
h100, s100, st100 = hi\_2(seq\_n100, 0.05, n2, p1, lda, P100, deg100)  
print(**'Последовательность'**, seq\_n100, **'Число действий'**, ef100, **'Вероятности'**, P100)  
print(**"Hi^2: "**, h100, s100, st100)  
  
  
f = open(**"out\_seq\_40\_0.2\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n40\_1:  
 p = str(i) + **','** f.write(p)  
  
p = **'\n'** + **'Число операций: '** + str(ef40\_1)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef40\_1)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s40\_1) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st40\_1) + **'\n'**f.write(p)  
**if** h40\_1:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()  
  
f = open(**"out\_seq\_40\_0.5\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n40\_2:  
 p = str(i) + **','** f.write(p)  
p = **'\n'** + **'Число операций: '** + str(ef40\_2)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef40\_2)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s40\_2) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st40\_2) + **'\n'**f.write(p)  
**if** h40\_2:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()  
  
f = open(**"out\_seq\_40\_0.8\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n40\_3:  
 p = str(i) + **','** f.write(p)  
   
p = **'\n'** + **'Число операций: '** + str(ef40\_3)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef40\_3)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s40\_3) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st40\_3) + **'\n'**f.write(p)  
**if** h40\_3:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()  
  
f = open(**"out\_seq\_100\_0.2\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n100\_1:  
 p = str(i) + **','** f.write(p)  
p = **'\n'** + **'Число операций: '** + str(ef100\_1)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef100\_1)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s100\_1) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st100\_1) + **'\n'**f.write(p)  
**if** h100\_1:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()  
  
f = open(**"out\_seq\_100\_0.5\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n100\_2:  
 p = str(i) + **','** f.write(p)  
p = **'\n'** + **'Число операций: '** + str(ef100\_2)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef100\_2)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s100\_2) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st100\_2) + **'\n'**f.write(p)  
**if** h100\_2:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()  
  
f = open(**"out\_seq\_100\_0.8\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n100\_3:  
 p = str(i) + **','** f.write(p)  
p = **'\n'** + **'Число операций: '** + str(ef100\_3)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef100\_3)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s100\_3) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st100\_3) + **'\n'**f.write(p)  
**if** h100\_3:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()  
  
f = open(**"out\_seq\_40\_un\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n40:  
 p = str(i) + **','** f.write(p)  
p = **'\n'** + **'Число операций: '** + str(ef40)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef40)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s40) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st40) + **'\n'**f.write(p)  
**if** h40:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()  
  
f = open(**"out\_seq\_100\_un\_standart.txt"**, **"w"**)  
p = **'Последовательность: '**f.write(p)  
**for** i **in** seq\_n100:  
 p = str(i) + **','** f.write(p)  
p = **'\n'** + **'Число операций: '** + str(ef100)  
f.write(p)  
p = **'\n'** + **'Эффективноесть: '** + str(1 / ef100)  
f.write(p)  
p = **'\n'** + **'Значение статистики S: '** + str(s100) + **'\n'**f.write(p)  
p = **'Достигнутый уровень значимости: '** + str(st100) + **'\n'**f.write(p)  
**if** h100:  
 p = **'Критери Xi^2 пройден, следовательно не причин для отклонения гипотезы о согласии распределения сгенерированной последовательности с равномерным распределением'** + **'\n'** f.write(p)  
**else**:  
 p = **'Критери Xi^2 не пройден, следовательно гипотеза о согласии распределения сгенерированной последовательности с равномерным распределением отвергается'** + **'\n'** f.write(p)  
f.close()

1. **Выводы**

В ходе проведённой работы, было разработано программное средство для генерации последовательностей псевдослучайных дискретных величин.

Исходя из результатов тестирования программы на разных входных данных можно сделать выводы:

1. Нестандартный алгоритм с законом распределения Пуассона даёт более качественные последовательности чем стандартный алгоритм с отрицательным биномиальным распределением.
2. Эффективнее по количеству операций использовать стандартный алгоритм с отрицательным биномиальным распределением при параметре распределения P не выше 0.5.
3. При уменьшении начальной вероятности P в отрицательным биномиальным распределении число операций растёт.
4. При этом, при приближении значения P качество последовательности падает.

Таким образом, выгоднее использовать стандартный алгоритм с отрицательным биномиальным распределением.