**Лабораторная работа 2.**

**Условие:**

Смоделировать дискретную случайную величину (задания на стр. 18-22). Исследовать точность моделирования.

1) Осуществить моделирование n = 1000 реализаций СВ из заданных дискретных распределений.

2) Вывести на экран несмещенные оценки математического ожидания и дисперсии, сравнить их с истинными значениями.

3) Для каждой из случайных величин построить свой χ2-критерий Пирсона с уровнем значимости ε=0.05. Проверить, что вероятность ошибки I рода стремится к 0.05.

4) Осуществить проверку каждой из сгенерированных выборок каждым из построенных критериев.

**Теория:**

**Отрицательное биномиальное распределение (с параметрами ):**

Случайная величина принимает только целые неотрицательные значения, причем . Параметры распределения: *m* - натуральное число,

В данной работе, сначала моделировалась последовательность БСВ, а потом по каждой БСВ строился соответствующий элемент выборки отрицательного биномиального распределения: отрезок [0;1] разбивался на интервалы длин и проверялось, в какой интервал попадает элемент последовательности БСВ.

**Распределение Пуассона (с параметром ):**

Случайная величина принимает только целые неотрицательные значения, причем .

В данной работе, сначала моделировалась последовательность БСВ, а потом по каждой БСВ строился соответствующий элемент выборки распределения Пуассона: отрезок [0;1] разбивался на интервалы длин и проверялось, в какой интервал попадает элемент последовательности БСВ.

**Геометрическое распределение (с параметром):**

Случайная величина принимает только целые неотрицательные значения, причем . Параметр распределения

В данной работе, сначала моделировалась последовательность БСВ, а потом по каждой БСВ строился соответствующий элемент выборки геометрического распределения: отрезок [0;1] разбивался на интервалы длин и проверялось, в какой интервал попадает элемент последовательности БСВ.

**Код программы:**

#include <iostream>

#include <time.h>

#include <algorithm>

**using** **namespace** std;

**void** NegBi(**int**\* sequence, **int** n, **int** r, **double** p) {

srand(time(0));

**double** q = 1 - p;

**double** p0 = pow(p, r);

**for** (**int** i = 0; i < n; i++) {

p = p0;

**int** z = 0;

**double** a = (**double**)rand() / RAND\_MAX;

a -= p;

**while** (a >= 0) {

z++;

p = p \* q \* (z - 1 + r) / z;

a -= p;

}

sequence[i] = z;

}

}

**void** P(**int**\* sequence, **int** n, **int** lambda) {

srand(time(0));

**double** p0 = exp(-lambda), p;

**for** (**int** i = 0; i < n; i++) {

p = p0;

**int** z = 0;

**double** a = (**double**)rand() / RAND\_MAX;

a -= p;

**while** (a >= 0) {

z++;

p = p \* lambda / z;

a -= p;

}

sequence[i] = z;

}

}

**void** G(**int**\* sequence, **int** n, **double** p) {

srand(time(0));

**double** p0 = p;

**double** q = 1 - p;

**for** (**int** i = 0; i < n; i++) {

p = p0;

**int** z = 0;

**double** a = (**double**)rand() / RAND\_MAX;

a -= p;

**while** (a >= 0) {

z++;

p \*= q;

a -= p;

}

sequence[i] = z;

}

}

**double** E(**int**\* sequence, **int** n) {

**double** res = 0;

**for** (**int** i = 0; i < n; i++) {

res += (**double**)sequence[i];

}

**return** res / n;

}

**double** D(**int**\* sequence, **int** n, **double** e) {

**double** res = 0;

**for** (**int** i = 0; i < n; i++) {

**double** k = ((**double**)sequence[i] - e);

res += k \* k;

}

**return** res / (n - 1);

}

**void** Pirson(**int**\* sequence, **int** n, **double**(\*f)(**int**)) {

**int**\* sortedSequence = **new** **int**[n];

**for** (**int** i = 0; i < n; i++) {

sortedSequence[i] = sequence[i];

}

sort(sortedSequence, sortedSequence + n);

**int** max = sortedSequence[n - 1];

**double** x = 0;

**int** count = 0;

**int** j = 0;

**for** (**int** i = 0; i <= max; i++) {

count = 0;

**while** (j < n && sortedSequence[j] == i) {

count++;

j++;

}

**double** p = f(i) \* n;

x += pow((count - p), 2) / p;

}

cout << "Критерий Пирсона (количество степеней свободы - " << max - 1 << ")\n";

cout << "Результат - " << x << "\n";

**delete**[] sortedSequence;

}

**double** fBi(**int** x) {

**double** p = pow(0.25, 6);

**double** q = pow(0.75, x);

**double** C = 1;

**for** (**int** i = 1; i <= x; i++) {

C \*= (**double**)(5 + i);

C /= i;

}

**return** C \* p \* q;

}

**double** fP(**int** x) {

**int** lambda = 3;

**double** res = pow(lambda, x) \* exp(-lambda);

**for** (**int** i = 2; i <= x; i++) {

res /= i;

}

**return** res;

}

**double** fG(**int** x) {

**double** p = 0.25;

**return** p \* pow(1 - p, x);

}

**int** main()

{

setlocale(LC\_ALL, "Russian");

**int** n = 1000;

**int**\* sequenceBi = **new** **int**[n];

**int**\* sequenceP = **new** **int**[n];

**int**\* sequenceG = **new** **int**[n];

NegBi(sequenceBi, n, 6, 0.25);

P(sequenceP, n, 3);

G(sequenceG, n, 0.25);

**double** e = E(sequenceBi, n);

**double** d = D(sequenceBi, n, e);

cout << "Отрицательное биномиальное распределение (с параметрами 6 и 0.25):\n";

cout << "Несмещенная оценка математического ожидания (истинное значение - 18): " << e <<"\n";

cout << "Несмещенная оценка дисперсии (истинное значение - 72): " << d << "\n";

Pirson(sequenceBi, n, fBi);

cout << "\n";

e = E(sequenceP, n);

d = D(sequenceP, n, e);

cout << "Распределение Пуассона (с параметром 3):\n";

cout << "Несмещенная оценка математического ожидания (истинное значение - 3): " << e << "\n";

cout << "Несмещенная оценка дисперсии (истинное значение - 3): " << d << "\n";

Pirson(sequenceP, n, fP);

cout << "\n";

e = E(sequenceG, n);

d = D(sequenceG, n, e);

cout << Геометрическое распределение (с параметром 0.25):\n";

cout << "Несмещенная оценка математического ожидания (истинное значение - 3): " << e << "\n";

cout << "Несмещенная оценка дисперсии (истинное значение - 12): " << d << "\n";

Pirson(sequenceG, n, fG);

**delete**[] sequenceBi, sequenceP, sequenceG;

**return** 0;

}

**Результаты:**

Отрицательное биномиальное распределение:

Несмещенная оценка математического ожидания (истинное значение - 18): 18.138

Несмещенная оценка дисперсии (истинное значение - 72): 73.3203

Критерий Пирсона (количество степеней свободы - 53)

Результат - 49.5619

Распределение Пуассона:

Несмещенная оценка математического ожидания (истинное значение - 3): 3.039

Несмещенная оценка дисперсии (истинное значение - 3): 2.96444

Критерий Пирсона (количество степеней свободы - 9)

Результат - 10.0552

Геометрическое распределение:

Несмещенная оценка математического ожидания (истинное значение - 3): 3.011

Несмещенная оценка дисперсии (истинное значение - 12): 12.4173

Критерий Пирсона (количество степеней свободы - 23)

Результат - 27.5227