ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

ОТЧЕТ

Лабораторная работа №3

по дисциплине “Моделирование”

Выполнили
студенты Брескун И. М., Селиванов В. В.
Ф.И.О.

Группы ИВ-022

Работу принял _____ ст. преп. кафедры ПМиК Бублей Д. А.
подпись

Защищена _____ Оценка _____

Новосибирск – 2024

СОДЕРЖАНИЕ

ПОСТАНОВКА ЗАДАЧИ

Реализовать генератор случайной дискретной величины. Реализовать выборку с возвратом и без возвратов. Проверить корректность работы, задав собственную случайную величину и собрав статистику выборок с возвратом и производя выборку без возврата

Теоретическая справка

Дискретное распределение – это распределение вероятностей, которое моделирует вероятности дискретных значений случайной величины. Дискретная случайная величина принимает значения из конечного или счетного множества возможных значений, то есть может принимать только целочисленные значения

Дискретное распределение с возвратом (с повторениями) – это дискретное распределение, при котором выбор каждого элемента производится независимо от предыдущих выборов, т.е. элементы могут быть выбраны более одного раза. Таким образом, вероятность выбора каждого элемента остается неизменной на протяжении всей выборки

Например, имеется множество {1, 2, 3}. Требуется сгенерировать выборку из 3-х элементов с возвратом. Тогда выбирается первый элемент, например, «2», затем этот элемент возвращается во множество, и выборка продолжается также из 3-х элементов. Таким образом, вероятность выбора каждого элемента равна 1/3 на каждой выборке, независимо от предыдущих выборок

Дискретное распределение без возврата (без повторений) – это дискретное распределение, при котором каждый элемент выбирается только один раз, то есть выбранные элементы не могут быть выбраны повторно. При каждой выборке элемент удаляется из множества, таким образом, вероятность выбора каждого элемента зависит от предыдущих выборок

Например, имеется множество {1, 2, 3}. Требуется сгенерировать выборку из 2-х элементов без возврата. Тогда выбирается первый элемент, например, «1», затем этот элемент удаляется из множества и выборка продолжается из оставшихся 2-х элементов {2, 3}. Таким образом, вероятность выбора каждого элемента зависит от предыдущих выборок

Гипергеометрическое распределение в теории вероятностей моделирует количество удачных выборок без возвращения из конечной совокупности

$$p_k = \frac{C_K^k C_{N-K}^{n-k}}{C_N^n}$$

Схемой Бернулли называется последовательность независимых в совокупности испытаний, в каждом из которых возможны лишь два исхода — «успех» и «неудача», при этом успех в одном испытании происходит с вероятностью $p \in (0, 1)$, а неудача — с вероятностью $q=1-p$. При любом $k = 0, 1, \dots, n$ имеет место равенство:

$$P_n(k) = C_n^k \cdot p^k \cdot q^{n-k}$$

Ход работы

1.1 Формулировка задачи для случайной величины

Из урны, в которой 20 белых и 10 черных шаров, наудачу вынимают 4 шара. Xi — вероятность того что попадется Xi белых шаров. Тогда закон распределения дискретной случайной величины для выборки с повторениями можно определить следующим образом:

Xi	0	1	2	3	4
p_i	0.01265	0.09876	0.296	0.39506	0.19753

Вероятности посчитаны с помощью формулы Бернулли:

$P_4(0) = C_4^0 \cdot p_0 \cdot q_4 \approx 0.01265$

$P_4(1) = C_4^1 \cdot p_1 \cdot q_3 \approx 0.09876$

$P_4(2) = C_4^2 \cdot p_2 \cdot q_2 \approx 0.296$

$P_4(3) = C_4^3 \cdot p_3 \cdot q_1 \approx 0.39506$

$P_4(4) = C_4^4 \cdot p_4 \cdot q_0 \approx 0.19753$

Для выборки без повторений закон распределения величины определяется следующим образом (вероятности рассчитаны по формуле гипергеометрической вероятности):

Xi	0	1	2	3	4
p_i	0.00766	0.08757	0.311995	0.415982	0.176793

$p_0 = \frac{C_{20}^0 \cdot C_{10}^4}{C_{30}^4} \approx 0.00766$

$p_0 = \frac{C_{20}^1 \cdot C_{10}^3}{C_{30}^4} \approx 0.08757$

$p_0 = \frac{C_{20}^2 \cdot C_{10}^2}{C_{30}^4} \approx 0.311995$

$p_0 = \frac{C_{20}^3 \cdot C_{10}^1}{C_{30}^4} \approx 0.415982$

$p_0 = \frac{C_{20}^4 \cdot C_{10}^0}{C_{30}^4} \approx 0.176793$

1.2 Результат работы программы

В результате работы программы построены диаграммы статистики (теоретической и практической) для выборок с возвратом и без возвратов:

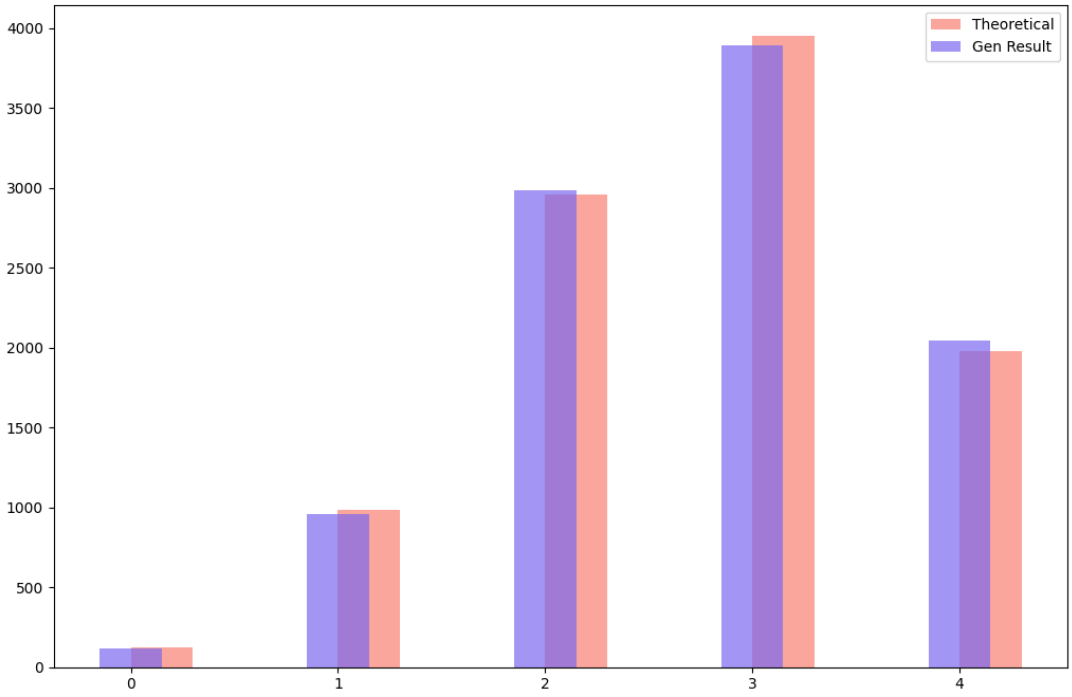

Рисунок 1 – Диаграмма для выборки с повторениями

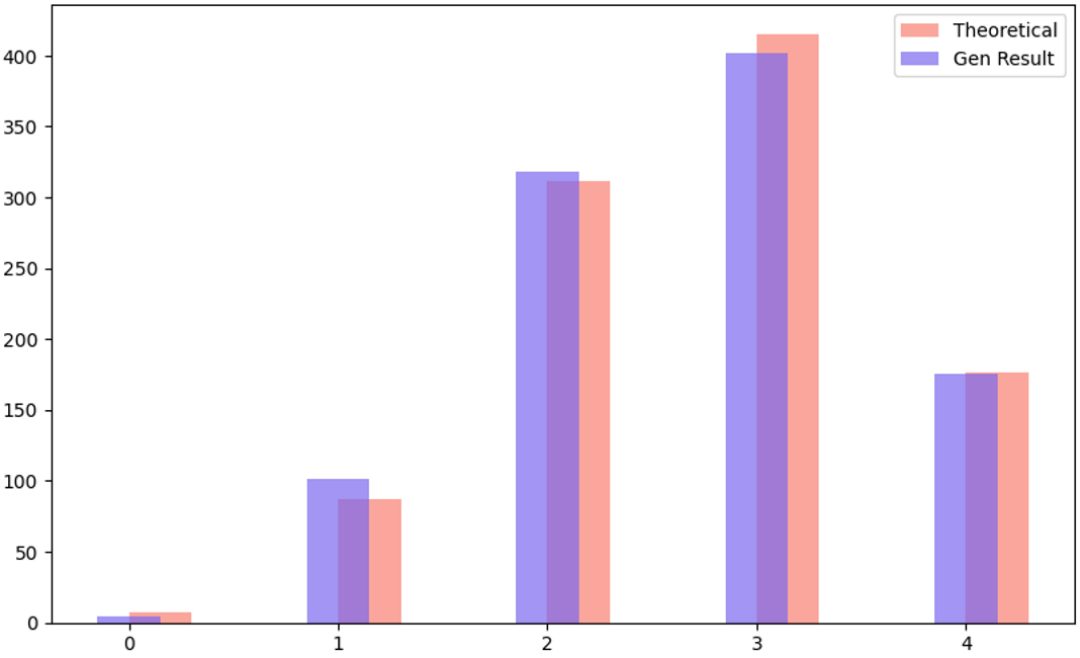

Рисунок 2 – Диаграмма для выборки без повторений

ЗАКЛЮЧЕНИЕ

Полученные в результате выполнения работы диаграммы распределения дискретной случайной величины (для выборки с возвратом и без возврата) соответствуют законам распределения, из чего следует, что генератор случайной дискретной величины реализован корректно

ПРИЛОЖЕНИЕ

Код программы

```
import numpy as np
from random import randint
import matplotlib.pyplot as plt
import math
from collections import Counter
import argparse
import sys

# Число испытаний
_N = 1000

# Размерность корзины
_n = 30

# Task
# Из урны, в которой 20 белых и 10 черных шаров, наудачу вынимают 4 шара
# Xi — вероятность того что попадется Xi белых шаров

theory_rep_selection = Counter()



def plot_data(x_t, y_t, x_pi, y_pi, title):
    plt.bar(x_t,    y_t,    width=0.3,    align='edge',    color    =    '#FA8072',
label="Theoretical", alpha=0.7)
    plt.bar(x_pi,  y_pi,  width=0.3,  color  =  '#7B68EE',  label="Gen  Result",
alpha=0.7)
    plt.title(title)
    plt.legend()
    plt.show()



# Вычисление теоретической выборки
def calc_theory_selection(mode, pi):
    global theory_rep_selection

    # Для выборки с повторениями
    tselect = {}
    for i in range(0, len(pi)):
```

```
        tselect[i] = int(_N * pi[i])

    # Приведение к объекту Counter
    theory_rep_selection = Counter(tselect)
    print(theory_rep_selection)


# m - размер выборки для алгоритма "без возвратов"
def get_selection(mode, pi, m=5):
    selection = []

    match mode:
        case 0:
        # Массив, хранящий суммы вероятностей (вектор накопленных вероятностей до
каждой категории)
            sum_pi = np.cumsum(pi)
            print(sum_pi)

            for i in range(_N):
                # Генерация случайного числа от 0 до 1
                rand_value = np.random.rand()
                # Нахождение индекса категории. Чем больше вероятность, тем больше
интервал
                category = np.searchsorted(sum_pi, rand_value)
                # Добавление категории в выборку
                selection.append(category)

            return Counter(selection)
        case 1:
            # Статистика по белым шарам (0, 1, 2, 3, или 4 были в выборке)
            # Логика будет в selection

            # Инициализация корзины с шарами (индексация шаров)
            basket = [None] * _n

            # Проводим N экспериментов
            for i in range(0, _N):
                white_count = 0

                local_selection = []
                # Индексы шаров
```

```
                for i in range(1, _n + 1):
                    basket[i - 1] = i

                # [n, n-1, n-2, n-3]
                # Произвести выборку
                for s in range(_n, _n - m, -1):
                    index = randint(0, s - 1)
                    print(f's: {s}, index: {index}')
                    # Добавить шар с индексом в выборку
                    local_selection.append(basket[index])
                    # Исключение из выборки выбранного значения
                    basket[index] = basket[s - 1]

                for index in local_selection:
                    if index <= 20:
                        white_count += 1

                # whites[white_count] += 1
                # print(indexes)
                print(white_count)
                selection.append(white_count)
            return Counter(selection)


def main():
    parser = argparse.ArgumentParser(description="Command line arguments parser")

    parser.add_argument('--mode', type=int, default=1, help='Режим 0 - выборка с
повторениями, 1 - без повторений')

    args = parser.parse_args()

    # Вероятности, расчитанные по формуле Бернулли и формуле гипергеометрической
вероятности соответственно
    pi = [0.01265, 0.09876, 0.296, 0.39506, 0.19753] if not args.mode else
[0.00766, 0.08757, 0.311995, 0.415982, 0.176793]
    print(pi)
    calc_theory_selection(args.mode, pi)

    # Генерация выборки с повторениями с заданными вероятностями
    selection_count = get_selection(args.mode, pi, len(pi) - 1)
```

```
    print(selection_count)


    plot_data(theory_rep_selection.keys(),        theory_rep_selection.values(),
selection_count.keys(),  selection_count.values(),  "Гистограмма  для  выборки  с
повторениями" if not args.mode else "Гистограмма для выборки без повторений")


if __name__ == '__main__':
    main()
```