МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет електроніки і комп’ютерних технологій

Кафедра системного проєктування

**Звіт**

про виконання лабораторної роботи № 3

з дисципліни «Прикладна статистика та ймовірнісні процеси»

«Моделювання випадкових чисел із заданим законом розподілу»

**Виконав:**

студент 2 курсу

групи Феп-22

Линва В.А.

**Перевірив:**

Доцент кафедри РКИ, к.ф.-м.н

Сас Н. Б.

**Львів 2022**

**Мета:**  Ознайомитись із темою та теоретичними відомостями. Виконати подані завдання.

**Хід роботи**

1. Змоделював послідовність із n=100 значень дискретної випадкової величини X, заданої одним із варіантів в таблиці. Номер варіанту – 3.

|  |
| --- |
|  |

1. Визначив вибіркове математичне сподівання та вибіркову дисперсію отриманої дискретної випадкової величини та порівняв їх з теоретичними значеннями.
2. Побудував частотну таблицю, вивів її на екран.
3. Побудував гістограму та оцінив за її допомогою закон розподілу випадкової величини X
4. Повторив виконання роботи для n=1000.

Результат виконання:

1. Вибірка із 100 значень.

|  |
| --- |
|  |

Значення збігаються із визначеними теоретичним значенннями заданої вибірки з досить не великою похибкою.

1. Вибірка із 1000 значень.

|  |
| --- |
|  |

Діаграми:

|  |
| --- |
|  |
|  |

**Висновок:** Я ознайомився із темо: « Моделювання випадкових чисел із заданим законом розподілу». Виконав усі поставлені практичні завдання.

**Додаток**

#include <iostream>

#include <cmath>

#include <time.h>

using namespace std;

const int SIZE(1000);

double frequency(double\* mass, double k, int size) // Обрахунок частоти

{

int count\_of\_k = 0;

for (int i = 0; i < size; i++)

{

if (mass[i] == k) { count\_of\_k++; }

}

return double(count\_of\_k) / double(size);

}

double vib\_ser(double\* mass, int size) // Математичне вибіркове середнє

{

double result = 0;

for (int i = 1; i <= SIZE; i++)

{

result += i \* (frequency(mass, i, size) \* size);

}

return result / size;

}

double variacy(double\* mass, int size) // Математична дисперсія

{

double temp = 0;

double result = 0;

double total = 0;

for (int i = 1; i <= SIZE; i++)

{

result += pow(i, 2) \* (frequency(mass, i, size) \* size);

}

result = result / size;

total = result - pow(vib\_ser(mass, size), 2);

return total;

}

double expectation(double\* mass, int size) // Математичне сподівання

{

double expectation = 0;

for (int i = 1; i <= SIZE; i++)

{

expectation += frequency(mass, i, size) \* i;

}

return expectation;

}

int main()

{

srand(time(NULL));

double x[7] = { 2.000 , 3.000 , 5.000, 12.000, 21.000, 33.000, 44.000 };

double p[7] = { 0.100, 0.250, 0.450, 0.500, 0.520, 0.850, 1.000 };

double randoms[SIZE];

for (int i = 0; i < SIZE; i++) // Заповнення масиву

{

randoms[i] = rand() % 1000 + 1.000;

randoms[i] = randoms[i] / 1000;

cout << randoms[i] << endl;

}

for (int i = 0; i < SIZE; i++)

{

if (randoms[i] > 0 && randoms[i] <= p[0])

randoms[i] = x[0];

else if (randoms[i] > p[0] && randoms[i] <= p[1])

randoms[i] = x[1];

else if (randoms[i] > p[1] && randoms[i] <= p[2])

randoms[i] = x[2];

else if (randoms[i] > p[2] && randoms[i] <= p[3])

randoms[i] = x[3];

else if (randoms[i] > p[3] && randoms[i] <= p[4])

randoms[i] = x[4];

else if (randoms[i] > p[4] && randoms[i] <= p[5])

randoms[i] = x[5];

else if (randoms[i] > p[5] && randoms[i] <= p[6])

randoms[i] = x[6];

}

cout << endl;

for (int i = 0; i < SIZE; i++)

{

cout << randoms[i] << " ";

}

cout << "\n" << endl;

cout << "Expectation = " << expectation(randoms, SIZE) << " |" << "20.16" << endl;

cout << "Variacy = " << variacy(randoms, SIZE) << " |" << "266.1144" << endl;

cout << endl << "[Interval]\t[Frequency]\t[Relative frequency]" << endl;

cout << "[" << "0 " << "- " << p[0] << "]" << "\t" << frequency(randoms, double(x[0]), SIZE) \* SIZE << "\t\t" << (frequency(randoms, double(x[0]), SIZE) \* SIZE) / 120 << endl;

cout << "[" << p[0] << " - " << p[1] << "]" << "\t" << frequency(randoms, double(x[1]), SIZE) \* SIZE << "\t\t" << (frequency(randoms, double(x[1]), SIZE) \* SIZE) / 120 << endl;

cout << "[" << p[1] << " - " << p[2] << "]" << "\t" << frequency(randoms, double(x[2]), SIZE) \* SIZE << "\t\t" << (frequency(randoms, double(x[2]), SIZE) \* SIZE) / 120 << endl;

cout << "[" << p[2] << " - " << p[3] << "]" << "\t" << frequency(randoms, double(x[3]), SIZE) \* SIZE << "\t\t" << (frequency(randoms, double(x[3]), SIZE) \* SIZE) / 120 << endl;

cout << "[" << p[3] << " - " << p[4] << "]" << "\t" << frequency(randoms, double(x[4]), SIZE) \* SIZE << "\t\t" << (frequency(randoms, double(x[4]), SIZE) \* SIZE) / 120 << endl;

cout << "[" << p[4] << " - " << p[5] << "]" << "\t" << frequency(randoms, double(x[5]), SIZE) \* SIZE << "\t\t" << (frequency(randoms, double(x[5]), SIZE) \* SIZE) / 120 << endl;

cout << "[" << p[5] << " - " << p[6] << "]" << "\t" << frequency(randoms, double(x[6]), SIZE) \* SIZE << "\t\t" << (frequency(randoms, double(x[6]), SIZE) \* SIZE) / 120 << endl;

}