**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ** НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП’ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

КУРСОВА РОБОТА

з дисципліни «Моделювання та аналіз програмного забезпечення»

Тема: «Датчик випадкових величин»

Варіант 25

**Виконав студент 3 курсу 322 гр.**

**кафедри інженерії програмного**

**забезпечення:**

Царук С.О.

**Прийняла:**

Волкогон В.О.

Київ 2021

**ЗМІСТ**

[1. ЗАВДАННЯ 3](#_Toc71983727)

[2. МЕТА КУРСОВОЇ РОБОТИ 4](#_Toc71983728)

[3. ВИКОНАННЯ РОБОТИ 5](#_Toc71983729)

[3.1. Визначення коефіцієнта С 5](#_Toc71983730)

[3.2. Знаходження інтегральної функції розподілу 6](#_Toc71983731)

[3.3. Знаходження зворотної до інтегральної функції 7](#_Toc71983732)

[3.4. Розробка коду датчика 7](#_Toc71983733)

[3.5. Розрахунок теоретичної гістограми 9](#_Toc71983734)

[3.6. Розробка програми тестування 10](#_Toc71983735)

[3.7. Обробка результатів тестування 11](#_Toc71983736)

1. ЗАВДАННЯ
2. Визначити коефіцієнт С.
3. Знайти інтегральну функцію розподілу.
4. Знайти зворотну до інтегральної функцію.
5. Розробити код датчика.
6. Розрахувати теоретичну гістограму.
7. Розробити програму тестування.
8. Обробити результати тестування.

|  |  |
| --- | --- |
| Варіант | Функція |
| 25 |  |

1. МЕТА КУРСОВОЇ РОБОТИ

Курсова робота має мету вивчити та засвоїти:

* процес розробки датчиків випадкових величин із заданим законом розподілу методом зворотної функції;
* методику перевірки датчиків випадкових чисел на основі критерію Пірсона.

Вихідними даними для розробки датчика є задана функція щільності розподілу випадкової величини f(x) та інтервал [a, b] на якому вона визначена.

Виконання курсової роботи складається з двох частин – розробки датчика і його перевірки і включає 7 етапів.

1. ВИКОНАННЯ РОБОТИ
   1. Визначення коефіцієнта С

Значення коефіцієнту С знаходиться з умови нормування:

Використовуючи формулу:

Тоді:

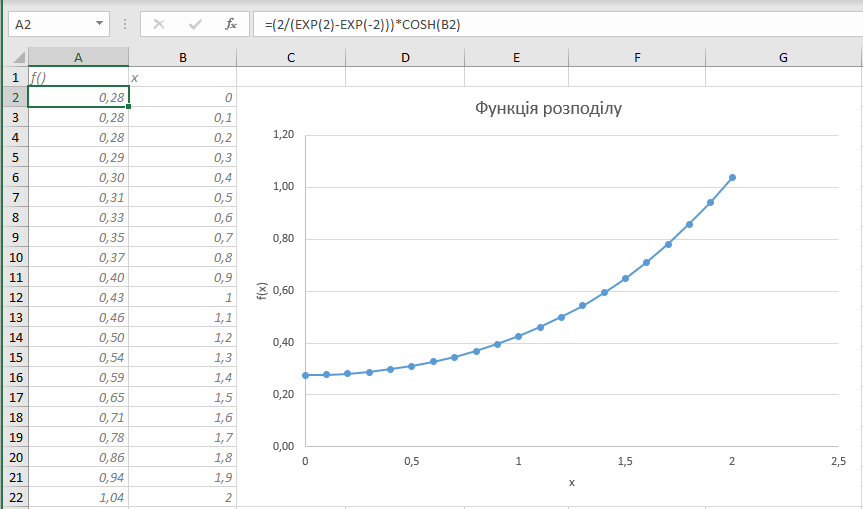


Рис.1. Графік функції розподілу випадкової величини.

* 1. Знаходження інтегральної функції розподілу

Інтегральна функція розподілу знаходиться за наступною формулою:

Використовуючи формулу:

Проведемо перевірку основних властивостей отриманої функції розподілу.

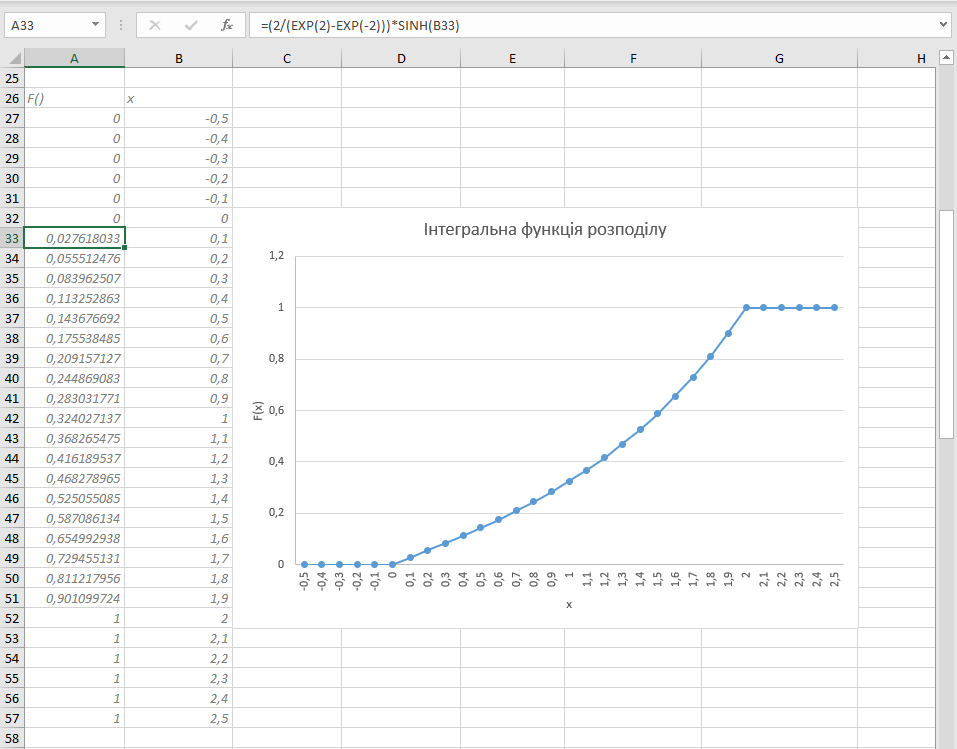


Рис.2. Графік інтегральної функції розподілу випадкової величини.

* 1. Знаходження зворотної до інтегральної функції

Зворотна функція відносно F має наступне формулювання:

Використовуючи формулу:

Виразимо х:

Z – випадкова величина, яку генерує датчик випадкових чисел і яка рівномірно розподілена в діапазоні [0, 1]. По закону розподілу f (x) випадкова величина x буде змінюватися в наступному діапазоні:

* 1. Розробка коду датчика

Код створеного класу:

package com.company;  
import java.util.concurrent.atomic.AtomicLong;  
  
public class Random\_number {  
  
 private final AtomicLong seed;//початкове зерно генератора випадкових чисел  
 private static final long *multiplier* = 0x5DEECE66DL;//множник  
 private static final long *addend* = 0xBL;//доданок  
 private static final long *mask* = (1L << 48) - 1;//маска  
 private static final double *DOUBLE\_UNIT* = 0x1.0p-53;  
  
 //Створення нового генератору випадкових чисел  
 public Random\_number() {  
 this(*seedUnique*() ^ System.*nanoTime*());  
 }  
  
 // Створення нового початкового зерна генератора випадкових чисел.  
 private static long seedUnique() {  
 for (;;) {  
 long current = *seedUnique*.get();  
 long next = current \* 181783497276652981L;  
  
 if (*seedUnique*.compareAndSet(current, next))  
 return next;  
 }  
 }  
  
 private static final AtomicLong *seedUnique* = new AtomicLong(8682522807148012L);  
  
 //Створює новий генератор випадкових чисел, використовуючи початкове зерно  
 public Random\_number(long seed) {  
 if (getClass() == Random\_number.class)  
 this.seed = new AtomicLong(*initialScramble*(seed));  
 else {  
 this.seed = new AtomicLong();  
 setSeed(seed);  
 }  
 }  
  
 private static long initialScramble(long seed) {  
 return (seed ^ *multiplier*) & *mask*;  
 }  
  
 //Встановлює зерно генератора випадкових  
 synchronized public void setSeed(long seed) {  
 this.seed.set(*initialScramble*(seed));  
 }  
  
 //Генерує псевдовипадкове число.  
 protected int next(int bits) {  
 AtomicLong seed = this.seed;  
 long nextSeed;  
 long oldSeed;  
  
 do {  
 oldSeed = seed.get();  
 nextSeed = (oldSeed \* *multiplier* + *addend*) & *mask*;  
 } while (!seed.compareAndSet(oldSeed, nextSeed));  
  
 return (int)(nextSeed >>> (48 - bits));  
 }  
  
 //Повертає число між значеннями 0 та 1  
 public double nextDouble() {  
 return (((long)(next(26)) << 27) + next(27)) \* *DOUBLE\_UNIT*;  
 }  
  
 //Повертає рівномірно розподілене по заданому закону розподілу f(x) число  
 //між значеннями 0 та 2  
 public double function() {  
 return Math.*log*((nextDouble()\*(Math.*exp*(2) - Math.*exp*(-2)))/2 +  
 Math.*sqrt*(Math.*pow*((nextDouble()\*(Math.*exp*(2) - Math.*exp*(-2)))/2,2) + 1));  
 }  
}

Даний клас створено на основі класу Random. Клас генерує випадкове значення z (метод nextDouble()) на проміжку [0;1]. Після цього дане значення підставляється у функцію F-1(z).

* 1. Розрахунок теоретичної гістограми

Ймовірність того, що випадкова величина X попаде в інтервал [a, b] розраховується по формулі:

Число попадань в інтервал [a, b]:

Нехай кількість випробувань N = 1000, число інтервалів m = 10.

Інтервал для датчика з діапазоном зміни випадкової величини на інтервалі [0, 2]:

Таблиця 1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № інтервалу | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Ліва гр. | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 | 1.2 | 1.4 | 1.6 | 1.8 |
| Права гр. | 0.2 | 0.4 | 0.6 | 0.8 | 1 | 1.2 | 1.4 | 1.6 | 1.8 | 2 |

Розрахуємо ймовірність попадань випадкової величини в кожен інтервал та виведемо теоретичне значення числа попадань випадкової величини в інтервали при N = 1000.

Формула:

Обчислення проведемо в програмі Excel:

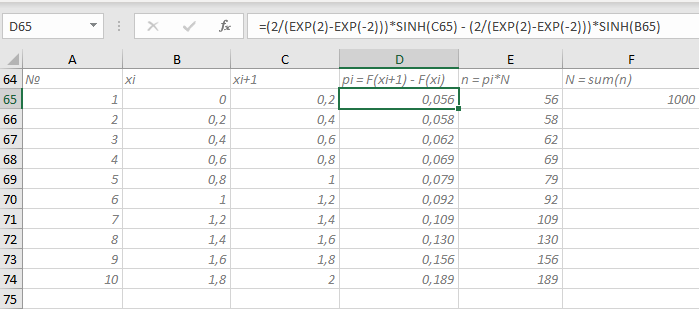


Рис.3. Результат проведення обчислення.

* 1. Розробка програми тестування

Код програми тестування:

package com.company;  
  
public class Main {  
 public static void main(String[] args) {  
 int m = 10;//Кількість інтервалів  
 int[] arr = new int[m];//Масив з кількістю попадань випадкової величини в кожен інтервал  
 Random\_number x = new Random\_number();//Створюємо новий генератор  
  
 //Проведення випробувань заданої кількості (1000)  
 for(int i = 0; i < 1000; i++){  
 double res = x.function();//Виклик генерації числа  
 *interval*(res,arr);//Перевірка результату  
 }  
  
 System.*out*.println("Кількість попаданнь в інтервали");  
 double left\_int = 0;//Ліва межа  
  
 //Виведення кількості попадань  
 for (int i : arr) {  
 System.*out*.print("Від " + (Math.*ceil*(left\_int\*10)/10) +  
 " до " + (Math.*ceil*((left\_int+0.2)\*10)/10) + ": ");  
 System.*out*.println(i);  
 left\_int += 0.2;  
 }  
 }  
  
 //Метод для запису попадання числа в певний інтервал  
 public static void interval(double x, int[] arr) {  
 double left\_int = 0;//Ліва межа  
 double d = 0.2;//Крок інтервалу  
  
 for(int i = 0; i < arr.length; i++){  
 if(i == arr.length - 1 && x >= left\_int && x <= left\_int+d){  
 arr[i] = arr[i] + 1;  
 break;  
 }  
  
 if(x >= left\_int && x < left\_int+d){  
 arr[i] = arr[i] + 1;  
 break;  
 }  
  
 left\_int = Math.*ceil*((left\_int+d)\*10)/10;//Округлення  
 }  
 }  
}

Результат виконання програми:

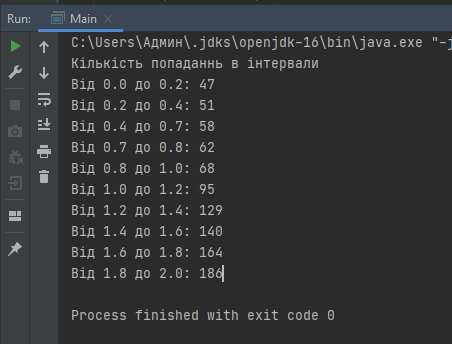


Рис.4. Результат тестування.

Гістограма для результатів тестування:



Рис.5. Гістограма.

* 1. Обробка результатів тестування

Перший крок – розрахунок відстані між емпіричним та теоретичним розподілами випадкової величини. В методі Пірсона в якості міри розходження U береться величина χ2 («хи-квадрат»):

Другий крок – знаходження порога (або критичного значення відстані). Поріг залежить від двох параметрів – рівня значимості критерію α і числа ступенів вільності k.

Третій крок – прийняття рішення. Якщо фактично спостерігаємо значення χ2 більше критичного, тобто χ2 > гіпотеза відкидається, якщо χ2, то гіпотеза не суперечить дослідним даним.

Знайдемо значення χ2 :

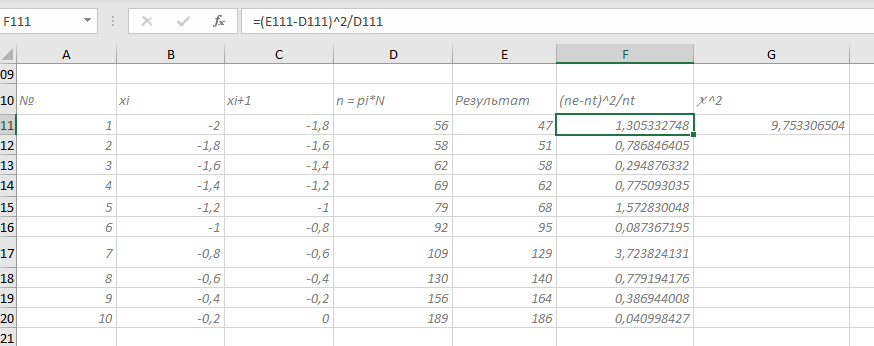


Рис.6. Обчислення значення χ2.

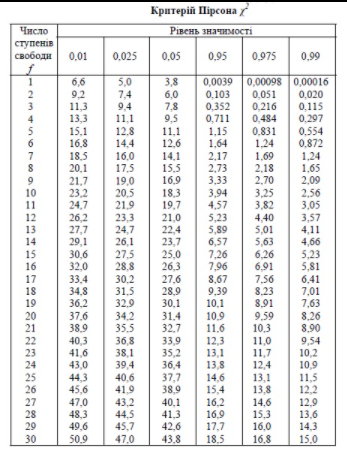


Рис.7. Таблиця критерія Пірсона.

При рівні значимості 0,05 та ступені свободи 9, = 16,9.

Отже χ2 - гіпотеза не суперечить дослідним даним.