ДНІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ОЛЕСЯ ГОНЧАРА

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА КОМП’ЮТЕРНИХ ТЕХНОЛОГІЙ

**Лабораторна робота №1**

**«Генерування випадкових чисел»**

**з курсу «Інженерія надійності систем»**

**Варіант №14**

Виконав:

студент групи ПК-21м-1

Панасенко Єгор Сергійович

Дніпро, 2022

# Постановка задачі

**Створити програмне забезпечення**, в якому реалізувати:

1. Генерування псевдовипадкових чисел з перевіркою їх якості. Для цього передбачити:
   1. моделювання послідовності обсягу *N* рівномірно розподілених випадкових чисел з проміжку [0; 1] за допомогою заданого генератора (див. індивідуальний варіант);
   2. перевірку якості змодельованої послідовності за допомогою чотирьох тестів: аперіодичності, збігу моментів, коваріації та критерію згоди з індивідуального варіанта;
   3. оцінку обсягу *N*, за якого досягається задана точність моделювання.
2. Генерування випадкових чисел, розподілених за заданим законом, з перевіркою їх якості. Для цього передбачити:
   1. моделювання послідовності обсягу *N* випадкових чисел, розподілених за заданим законом (див. індивідуальний варіант);
   2. перевірку якості одержаної послідовності за допомогою критерію згоди (див. індивідуальний варіант);
   3. перевірку якості одержаної послідовності шляхом перевірки гіпотези про збіг параметрів моделювання з оцінками, обчисленими за послідовністю.

**Зробити висновки про якість генерування** випадкових чисел реалізованими датчиками.

За результатом виконання лабораторної роботи **оформити звіт**.

**Індивідуальні варіанти завдань**

Номер індивідуального варіанта відповідає порядковому номеру студента у журналі відвідувань групи.

Генератор псевдовипадкових чисел обрати відповідно варіанту з таблиці:

|  |  |
| --- | --- |
| Варіант | Генератор |
| 1, 7, 13, 19 | , |
| 2, 8, 14, 20 | , , |
| 3, 9, 15, 21 | , |
| 4, 10, 16, 22 | , |
| 5, 11, 17, 23 | , |
| 6, 12, 18, 24 | , , |

Початкове значення  обирати, використовуючи системний час.

Тип закону розподілу та критерій згоди обрати відповідно варіанту з таблиці:

|  |  |  |
| --- | --- | --- |
| Варіант | Тип розподілу | Критерій згоди |
| 1 | Релея | Колмогорова |
| 2 | Логарифмічно-нормальний (функціональних перетворень) | Пірсона |
| 3 | Експоненціальний | Колмогорова |
| 4 | Вейбулла | Пірсона |
| 5 | Нормальний (метод зворотньої функції) | Колмогорова |
| 6 | Нормальний (метод додавання) | Пірсона |
| 7 | Найбільшого значення | Колмогорова |
| 8 | Логарифмічно-нормальний (метод полярних координат) | Пірсона |
| 9 | Лапласа | Колмогорова |
| 10 | З класу екстремальних | Пірсона |
| 11 | Експоненціальний | Колмогорова |
| 12 | Нормальний (функціональних перетворень) | Колмогорова |
| 13 | Вейбулла | Колмогорова |
| 14 | Нормальний (метод полярних координат) | Пірсона |
| 15 | Логарифмічно-нормальний (метод додавання) | Колмогорова |
| 16 | Лапласа | Пірсона |
| 17 | З класу екстремальних | Колмогорова |
| 18 | Експоненціальний | Пірсона |
| 19 | Нормальний (метод зворотньої функції) | Пірсона |
| 20 | Нормальний (метод полярних координат) | Колмогорова |
| 21 | Релея | Пірсона |
| 22 | Нормальний (метод додавання) | Колмогорова |
| 23 | Вейбулла | Пірсона |
| 24 | Паретто | Колмогорова |
| 25 | Логарифмічно-нормальний (метод зворотньої функції) | Пірсона |
| 26 | Найбільшого значення | Колмогорова |
| 27 | Релея | Колмогорова |
| 28 | Експоненціальний | Колмогорова |
| 29 | Нормальний (функціональних перетворень) | Пірсона |
| 30 | Найбільшого значення | Пірсона |

# Опис програми

У программі використовуються модулі для роботи з випадковими числами (random), математикою (math та numpy), часом (time), графіками (matplotlib).

Є глобальна змінна MAX\_N, яка відповідає за максимальний розмір послідовності.

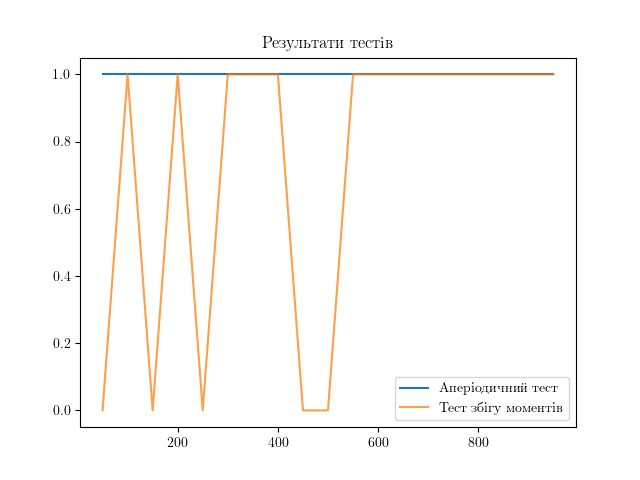
Реалізовані функції:

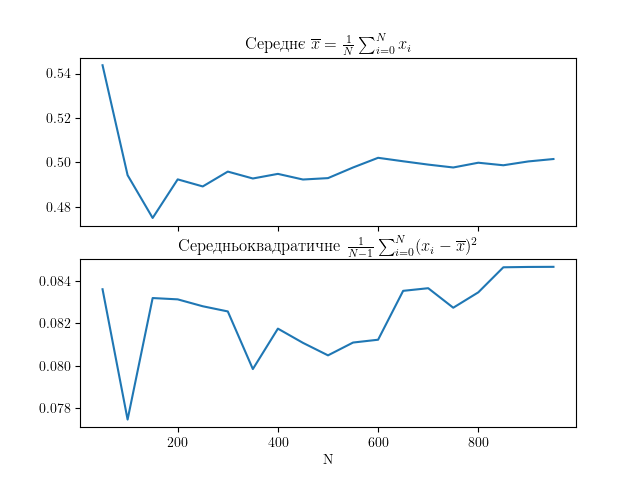
* calc\_b — обчислює 
* aperiodic\_test\_i — перевіряє чи є періодичність заданого розміру
* aperiodic\_test — перевіряє чи є періодичність від 1 до N
* avg\_calc — обчислює середнє і середнєквадратичне
* coincidence\_of\_moments\_test — тест збігу моментів
* pseudo — генератор 
* analyze\_arr — аналіз послідовності та вивід графіків

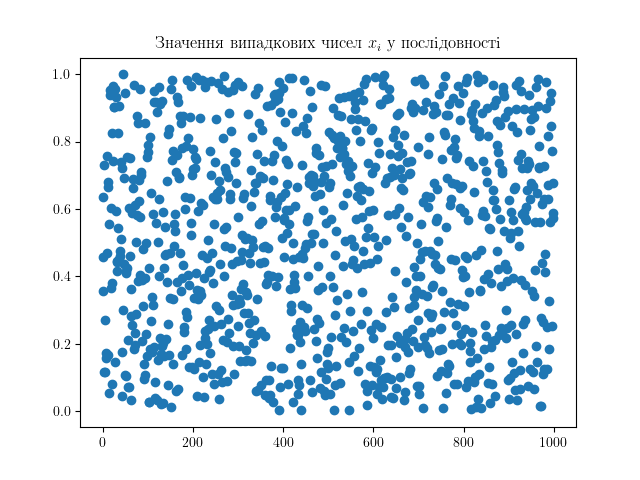
# Тестування

Генератор випадкових чисел за варіантом:

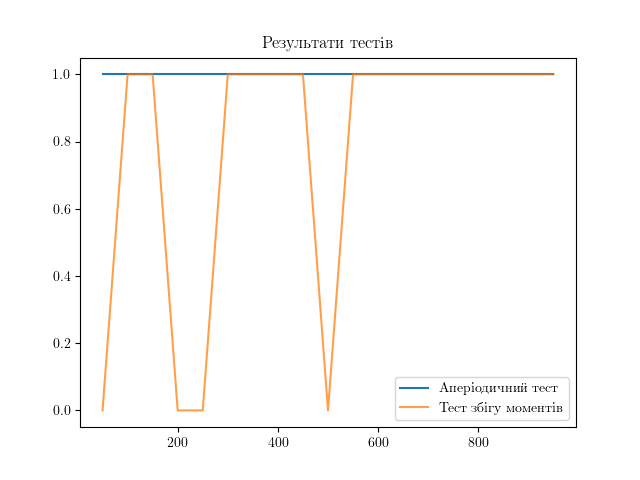
|  |
| --- |
| , , |

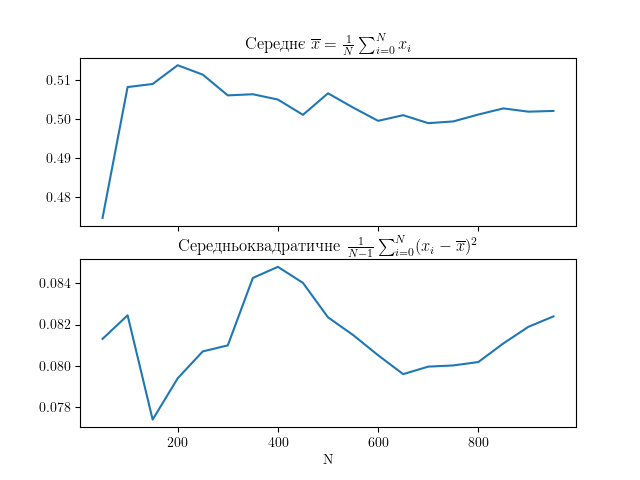


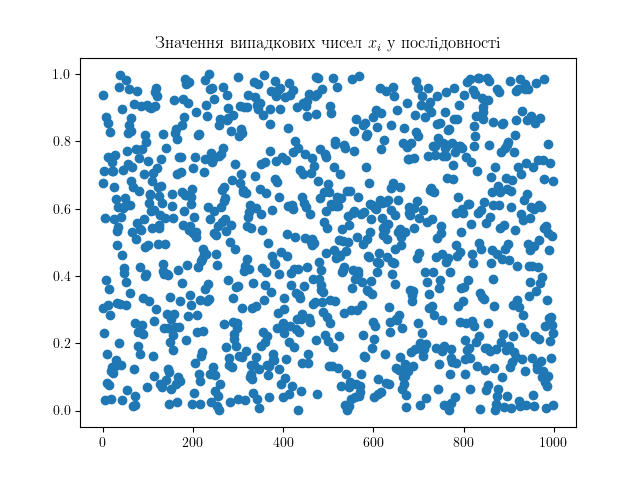




Вбудований генератор випадкових чисел:







# Висновки

Реалізовано за варіантом генерування псевдовипадкових чисел з перевіркою їх якості. Перевірено якості змодельованої послідовності за допомогою тестів: аперіодичності, збігу моментів. Також згенеровано послідовність псевдовипадкових чисел за допомогою вбодованих функції та перевірено їх якість.

# Код

#!/usr/bin/env python3

import random

import math

import time

import matplotlib.pyplot as plt

import numpy as np

from matplotlib import rc

plt.rc('text', usetex=True)

plt.rc('font', \*\*{'family': 'verdana'})

# ~ plt.rc('text.latex', unicode=True)

#plt.rc('text.latex', preamble=r'\usepackage[utf8x]{inputenc},\usepackage[russian]{babel},\usepackage{amsmath}')

plt.rcParams['text.latex.preamble'] = r'\usepackage[ukrainian]{babel}'

MAX\_N = 1000

T = 10\*\*11

def calc\_b(bj):

k = 2

return ((7\*\*(4\*k+1)\*bj)%T)

def aperiodic\_test\_i(a, x, N):

arr = a[:x]

for i in range(N):

if not math.isclose(a[i%x], a[i]):

return False

return True

def aperiodic\_test(arr):

N = len(arr)

x = 1

cycle = False

while not cycle and x < N:

cycle = aperiodic\_test\_i(arr, x, N)

x += 1

return not cycle

def avg\_calc(arr):

N = len(arr)

a=sum(arr)/N

a2 = sum([(i-a)\*\*2 for i in arr])/(N-1)

return a, a2

def coincidence\_of\_moments\_test(arr):

N = len(arr)

x=sum(arr)/N

S2 = sum([(arr[i] - x)\*\*2 for i in range(N)])/(N-1)

C1 = x-1/2

C2 = S2-1/12

if math.sqrt(12\*N)\*abs(C1) > 1.96:

return False

if (N-1)/N\*abs(C1)/math.sqrt(0.0056/N+0.0028/N\*\*2-0.0083/N\*\*3) > 1.96:

return False

return True

def pseudo(N):

arr = []

bj = time.time()\*1000

for i in range(N):

bj = calc\_b(bj)

rj = (bj/T)

arr.append(rj)

return arr

def analyze\_arr(arr):

res = []

for i in range(50, MAX\_N, 50):

myarr = arr[:i]

test1 = aperiodic\_test(myarr)

test2 = coincidence\_of\_moments\_test(myarr)

a, a2 = avg\_calc(arr[:i])

res.append((i, a, a2, int(test1), int(test2)))

res = np.array(res)

plt.figure()

plt.scatter(range(MAX\_N), arr)

plt.title(r"Значення випадкових чисел $x\_i$ у послідовності")

fig, axs = plt.subplots(2)

axs[0].plot(res[:,0], res[:,1], label="Середнє")

axs[0].set\_title(r"Середнє $\overline{x}=\frac{1}{N}\sum\_{i=0}^Nx\_i$ ")

axs[1].plot(res[:,0], res[:,2], label="Середньоквадратичне")

axs[1].set\_title(r"Середньоквадратичне $\frac{1}{N-1}\sum\_{i=0}^N(x\_i-\overline{x})^2$")

for ax in axs.flat:

ax.set(xlabel='N', ylabel='')

ax.label\_outer()

plt.figure()

plt.plot(res[:,0], res[:,3], label="Аперіодичний тест")

plt.plot(res[:,0], res[:,4], label="Тест збігу моментів", alpha=0.75)

plt.legend()

plt.title(r"Результати тестів")

def main():

arr = pseudo(MAX\_N)

analyze\_arr(arr)

arr = np.random.rand(MAX\_N)

analyze\_arr(arr)

plt.show()

main()