|  |
| --- |
| МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ **Кафедра програмних систем і технологій**  **Звіт**  **до лабораторної роботи № 2**  з дисципліни  **«Ймовірнісні основи програмної інженерії»**  **Студента групи ІПЗ-22 групи**  **Мукан Дмитра Сергійовича** |

**Мета роботи:**

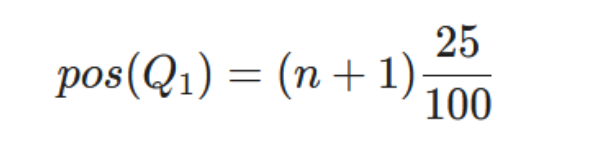
Навчитись використовувати на практиці набуті знання про лінійні перетворення та графічне зображення даних.

**Завдання:**

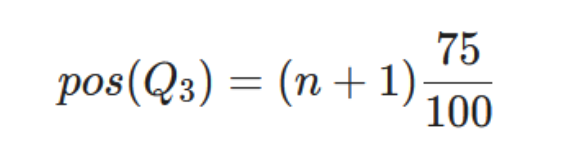
# Изображение выглядит как текст Автоматически созданное описание

**Математична модель:**

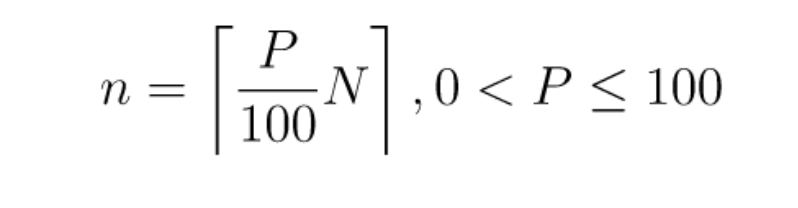
Формула для 𝑄1:



Формула для 𝑄3:



Формула розрахунку порядкового номера n для заданого процентиля *P*:

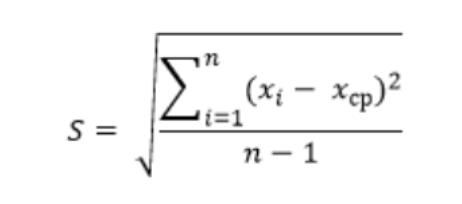


Формула пошуку середнього квадратичного відхилення:

Изображение выглядит как текст, часы

Автоматически созданное описание

Формула пошуку стандартного відхилення:



Для зміни середнього значення за лінійною трансформацією використовувалась формула:

Изображение выглядит как текст

Автоматически созданное описание

**Код алгоритму:**

import math  
import matplotlib.pyplot as plt  
import numpy as np  
  
def mean(l:list):  
 n = len(l)  
 mean = sum(l) / n  
 return mean  
  
def perc\_and\_quart(list:list,perc:int):  
 a = (perc/100)\*(len(list)+1)  
 b = a - math.floor(a)  
 c = list[math.floor(a)]  
 c1 = list[math.floor(a-1)]  
 d = c1+b\*(c-c1)  
 return d  
  
def average\_vidhil(l:list):  
 n = len(l)  
 mean = sum(l) / n  
 deviations = [math.fabs((mean-x)) for x in l]  
 variance = sum(deviations) / n  
 return variance  
  
def despertion(l:list):  
 n = len(l)  
 mean = sum(l) / n  
 deviations = [(x - mean) \*\* 2 for x in l]  
 variance = sum(deviations) / n  
 return variance  
def standart\_vidhil(data):  
 var = despertion(data)  
 std\_dev = math.sqrt(var)  
 return std\_dev  
  
def hz(l:list):  
 # 100 = 100\*a+b  
 # 90 = mean \* a +b  
 mean\_1 = mean(l)  
 a = np.array([[100,1],  
 [mean\_1,1]])  
 b = np.array([100,90])  
 res = np.linalg.solve(a, b)  
 a1 = str(res).replace('[', '').replace(']', '').replace(' ', ' ').strip().split(' ')  
 a, b = a1  
 temp\_list = []  
 for i in l:  
 if i == 100:  
 temp\_list.append(i)  
 else:  
 temp = float(a)\*float(i)+float(b)  
 temp\_list.append(math.ceil(temp))  
 return temp\_list  
  
def stovbur(list:list):  
 temp = 0  
 a = ''  
 for i in list:  
 if 10 > i > 0 and temp != -999:  
 temp = 0  
 a = a + '0|'  
 temp = -999  
 if 10 > i > 0 and temp == -999:  
 a = a + f'{str(i)},'  
 elif 9 < i < 100:  
 n = str(abs(i))[0]  
 if temp == n:  
 a = a + f'{str(abs(i))[1:]},'  
 elif temp != n:  
 a = a + f'\n{n}|{str(abs(i))[1:]},'  
 temp = n  
 elif i > 99:  
 n = str(abs(i))[:2]  
 if temp == n:  
 a = a + f'{str(abs(i))[2:]},'  
 elif temp != n:  
 a = a + f'\n{n}|{str(abs(i))[2:]},'  
 temp = n  
 return(a)  
  
def boxplot(list:list):  
 plt.figure(figsize=(10, 8))  
 plt.style.use('ggplot')  
 plt.rcParams['axes.unicode\_minus'] = False  
  
 plt.boxplot(x=list,  
 patch\_artist=True,  
 showmeans=True, # Показать означает как точки  
 widths=0.2,  
 boxprops={'color': 'black', 'facecolor': '#9999ff'},  
 # Установить свойства поля, цвет заливки и цвет рамки  
 flierprops={'marker': 'o', 'markerfacecolor': 'red', 'color': 'black'},  
 # Установите атрибуты выброса, форму точки, цвет заливки и цвет границы  
 meanprops={'marker': 'D', 'markerfacecolor': 'indianred'},  
 # Установите свойства средней точки, формы и цвета заливки точки  
 medianprops={'linestyle': '--',  
 'color': 'orange'}) # Установить срединные свойства линии, тип и цвет линии  
 plt.show()  
  
  
def all(f,res):  
 list = []  
 for i in f:  
 list.append(int(i))  
 list.sort()  
 ###################################################  
 res.write(f'Percentile90 : {perc\_and\_quart(list, 90)}\n')  
 res.write(f'Q3 : {perc\_and\_quart(list, 75)}\n')  
 res.write(f'Q1 : {perc\_and\_quart(list, 25)}\n')  
 ###################################################  
 res.write(f'Standart Vidhil: {standart\_vidhil(list)}\n')  
 res.write(f'Average Vidhil: {average\_vidhil(list)}\n')  
 ###################################################  
 res.write(f'{list}\n')  
  
 res.write(f'{hz(list)}\n')  
 ###################################################  
 res.write(f'{stovbur(list)}\n')  
 ###################################################  
 boxplot(list)  
 ###################################################  
  
print('1(end)/10/100 ?:', end='')  
q = int(input())  
while q != 1:  
 if q == 10:  
 print('10 is done')  
 f = open('input\_10.txt', 'r')  
 res = open('res\_10.txt', 'w')  
  
 all(f,res)  
  
 f.close()  
 res.close()  
 elif q == 100:  
 print('100 is done')  
 f = open('input\_100.txt', 'r')  
 res = open('res\_100.txt', 'w')  
  
 all(f,res)  
  
 f.close()  
 res.close()  
  
 else:  
 print('I don`t know this type')  
 print('1(end)/10/100/1000 ?:', end='')  
 q = int(input())

Середнє значення

def mean(l:list):  
 n = len(l)  
 mean = sum(l) / n  
 return mean

персентилі та квартилі

def perc\_and\_quart(list:list,perc:int):  
 a = (perc/100)\*(len(list)+1)  
 b = a - math.floor(a)  
 c = list[math.floor(a)]  
 c1 = list[math.floor(a-1)]  
 d = c1+b\*(c-c1)  
 return d

Сер відхіл

def average\_vidhil(l:list):  
 n = len(l)  
 mean = sum(l) / n  
 deviations = [math.fabs((mean-x)) for x in l]  
 variance = sum(deviations) / n  
 return variance

Дисперсія

def despertion(l:list):  
 n = len(l)  
 mean = sum(l) / n  
 deviations = [(x - mean) \*\* 2 for x in l]  
 variance = sum(deviations) / n  
 return variance

Станд відхіл

def standart\_vidhil(data):  
 var = despertion(data)  
 std\_dev = math.sqrt(var)  
 return std\_dev

Стовбур-листя

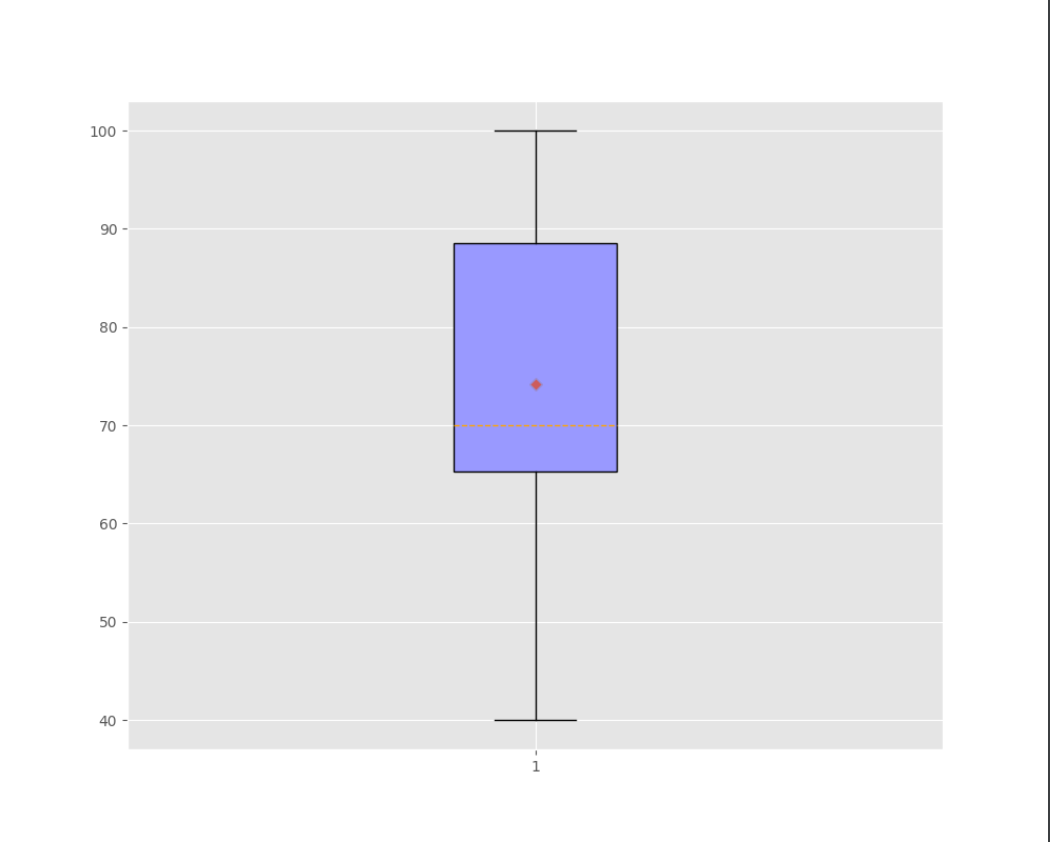
def stovbur(list:list):  
 temp = 0  
 a = ''  
 for i in list:  
 if 10 > i > 0 and temp != -999:  
 temp = 0  
 a = a + '0|'  
 temp = -999  
 if 10 > i > 0 and temp == -999:  
 a = a + f'{str(i)},'  
 elif 9 < i < 100:  
 n = str(abs(i))[0]  
 if temp == n:  
 a = a + f'{str(abs(i))[1:]},'  
 elif temp != n:  
 a = a + f'\n{n}|{str(abs(i))[1:]},'  
 temp = n  
 elif i > 99:  
 n = str(abs(i))[:2]  
 if temp == n:  
 a = a + f'{str(abs(i))[2:]},'  
 elif temp != n:  
 a = a + f'\n{n}|{str(abs(i))[2:]},'  
 temp = n  
 return(a)

Коробковий графік

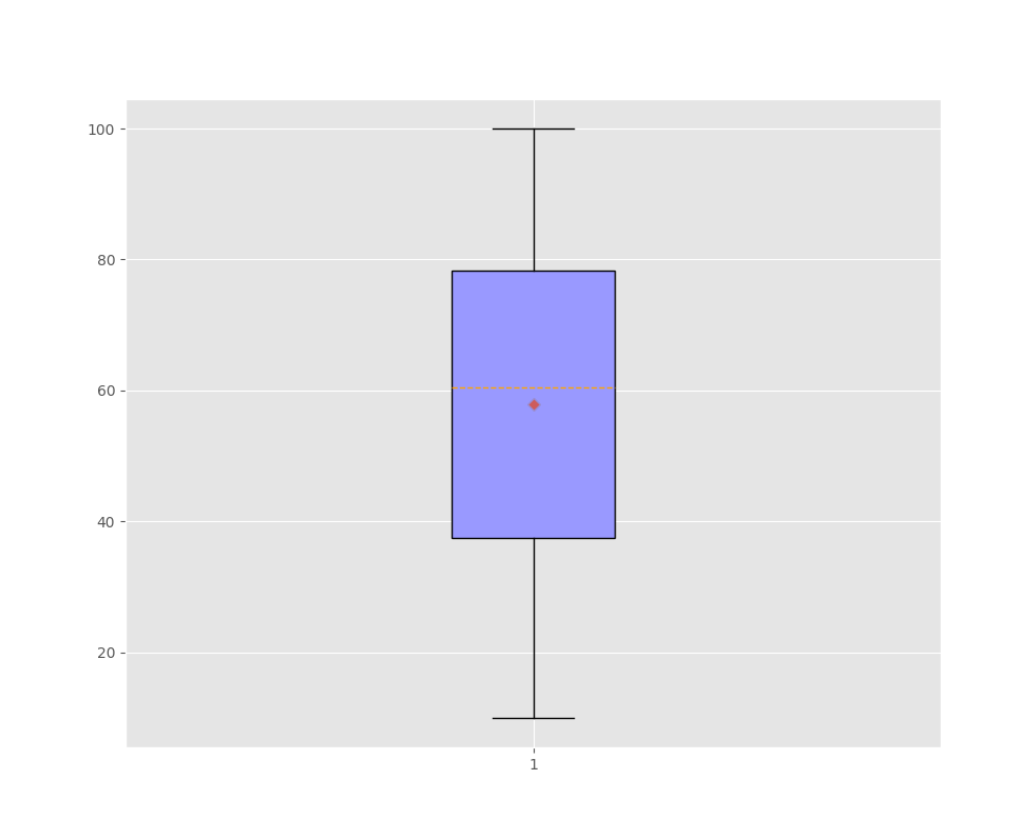
def boxplot(list:list):  
 plt.figure(figsize=(10, 8))  
 plt.style.use('ggplot')  
 plt.rcParams['axes.unicode\_minus'] = False  
  
 plt.boxplot(x=list,  
 patch\_artist=True,  
 showmeans=True, # Показать означает как точки  
 widths=0.2,  
 boxprops={'color': 'black', 'facecolor': '#9999ff'},  
 # Установить свойства поля, цвет заливки и цвет рамки  
 flierprops={'marker': 'o', 'markerfacecolor': 'red', 'color': 'black'},  
 # Установите атрибуты выброса, форму точки, цвет заливки и цвет границы  
 meanprops={'marker': 'D', 'markerfacecolor': 'indianred'},  
 # Установите свойства средней точки, формы и цвета заливки точки  
 medianprops={'linestyle': '--',  
 'color': 'orange'}) # Установить срединные свойства линии, тип и цвет линии  
 plt.show()

Результат для 10

Изображение выглядит как текст

Автоматически созданное описание

Результат для 100

Изображение выглядит как текст

Автоматически созданное описание

**Висновок**

Протягом даної лабораторної роботи було використано на практиці набуті знання про лінійні перетворення та графічне зображення даних. Було знайдено значення квартилів, персентиля, відхилення за заданими даними. Також було побудовано за інформацією з вхідних файлів діаграму стовбур-листя та коробкову діаграму.