

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра програмних систем і технологій

Дисципліна

Лабораторна робота № 2

«Ймовірнісні основи програмної інженерії»

| | | | |
|-----------------|--|--------------------|--|
| Виконав: | Ільницький Олександр Віталійович | Перевірила: | |
| Група | ІПЗ-24 | Дата перевірки | |

| | | | |
|----------------|-------|--------|--|
| Форма навчання | денна | Оцінка | |
| Спеціальність | 121 | | |
| 2022 | | | |

Завдання

1. Знайдіть 1Q, 3Q, P90
2. Знайдіть середнє та стандартне відхилення цих оцінок.
3. Через незадоволення низькими оцінками викладач вирішив використати шкалу форми $y = ax + b$, щоб відредувати оцінки. Він хотів, щоб середнє значення масштабних оцінок становило 95, а оцінка 100, щоб залишалася рівною 100.
4. Показати дані за допомогою діаграми "стовбур – листя".
5. Відобразити дані за допомогою коробкового графіка.
6. Зробити висновок.

Математична модель

1. Для розрахунку 1-ого та 3-ого квантилю та 90-ого персантилю будемо використовувати формулу:

$$\frac{k}{100}(N+1)^{\text{th}}$$

Тепер для знаходження його значення нам потрібно використати наступну формулу:

$$P_n = x_n + x_d * (x_{n+1} - x_n), \text{ де}$$

2. Для знаходження середнього відхилення використаємо формулу:

$$\frac{1}{n} \sum_{i=1}^n |x_i - m(X)|$$

, де

стандартне відхилення:

$$s = \sqrt{\frac{\sum_{i=1}^n (x_i - x_{cp})^2}{n - 1}}$$

Для виконання завдання потрібно знайти значення а та b, розв'язавши систему рівнянь

$$\begin{cases} 100 = 100a + b \\ 95 = \bar{x}_i a + b \end{cases}$$

$$y = ax + b$$

Pseudocode:

FUNCTION writeFile(file,data):

```
file.write(str(data))
```

FUNCTION percentile(k, array_len ,file)

```
Pk = float(k / 100) * (array_len + 1)
file.write(f'\n{k} Percentile: {Pk}')
return Pk
```

FUCNTION quartile(I, array_len, file):

```
Qk = float(i / 4) * (array_len + 1)
file.write(f'\n{i} Quartile: {Qk}')
return Qk
```

Псевдокод алгоритму знаходження середнього та стандартного відхилення оцінок.

FUNCTION avg(array):

```
sum = 0
for num in array:
    sum += num
return sum / len(array)
```

FUNCTION variance(array):

```
avg_n = avg(array)
frequency_sum = 0
sum_ = 0
for num in array:
    sum_ += array.count(num) * (num - avg_n) ** 2
```

```
frequency_sum += array.count(num)
return sum_ / frequency_sum
```

FUNCTION standart_dev(array, file):

```
var_x = variance(array)
writeFile(file, f"Standard Deviation:{math.sqrt(var_x)}\n")
```

FUNCTION middle_dev(array, file):

```
sum_ = 0
avg_n = avg(array)
array_len = len(array)
for i in range(array_len - 1):
    sum_ += math.fabs(array[i] - avg_n)
res = sum_ / array_len
writeFile(file, f"Middle Deviation:{res}")
```

Псевдокод алгоритму обчислення відредагованих оцінок

FUNCTION task3(array, file):

```
avg_n = avg(array)
res_array = []
array_len = len(array)
A = np.array([[100, 1], [avg_n, 1]])
X = np.linalg.solve(A, np.array([100, 95]))

for i in range(array_len):
    res_array.append((X[0] * array[i] + X[1]))

writeFile(file, f"\nA = {X[0]}\nB = {X[1]}")
writeFile(file, f"\nMarks:{res_array}")
```

Псевдокод алгоритму побудови діаграми "стовбур – листя".

FUNCTION task4(array, file):

```
array_len = len(array)
file.write("\nList diagram\n")
file.write("-" * 20)
sorted_array = sorted(array)

prev_n = int(sorted_array[0]/10)

print(sorted_array)
file.write(f"\n{prev_n}\t|")
for i in range(array_len):
    if int(prev_n) == int(sorted_array[i] / 10):
        file.write(f"{sorted_array[i] % 10} ")
    else:
        prev_n = int(sorted_array[i] / 10)
        file.write(f"\n{prev_n}\t|")
        file.write(f"{sorted_array[i] % 10} ")
```

FUNCTION show_diagram(array):

```
plt.boxplot(array)
plt.show()
```

Код програми

```
from Deviation import standart_dev, middle_dev
from array_funcs import data_append
from Percantile_Quartile import percentile, quartile
from task3 import task3
from task4 import task4
from Diagram import show_diagram

file = open('./task_02_data/input_10.txt')
answer_file = open('task.txt', 'w')
array = []

data_append(array, file)

standart_dev(array, answer_file)
middle_dev(array, answer_file)
percentile(90, len(array), answer_file)
quartile(1, len(array), answer_file)
quartile(3, len(array), answer_file)
task3(array, answer_file)
task4(array, answer_file)
show_diagram(array)
```

```
def data_append(array, file):
    for i, line in enumerate(file):
        if i != 0:
            array.append(int(line.strip()))
```

```
import math

def writeFile(file, data):
    file.write(str(data))

def avg(array):
    sum = 0
    for num in array:
        sum += num
    return sum / len(array)
```

```

def variance(array):
    avg_n = avg(array)
    frequency_sum = 0
    sum_ = 0
    for num in array:
        sum_ += array.count(num) * (num - avg_n) ** 2
        frequency_sum += array.count(num)
    return sum_ / frequency_sum

def standart_dev(array, file):
    var_x = variance(array)
    writeFile(file, f"Standard Deviation:{math.sqrt(var_x)}\n")

def middle_dev(array, file):
    sum_ = 0
    avg_n = avg(array)
    array_len = len(array)
    for i in range(array_len - 1):
        sum_ += math.fabs(array[i] - avg_n)
    res = sum_ / array_len
    writeFile(file, f"Middle Deviation:{res}")

```

```

import matplotlib.pyplot as plt

```

```

def show_diagram(array):
    plt.boxplot(array)
    plt.show()

```

```

def percentile(k, array_len, file):
    Pk = float(k / 100) * (array_len + 1)
    file.write(f'\n{k} Percentile: {Pk}')
    return Pk

def quartile(i, array_len, file):
    Qk = float(i / 4) * (array_len + 1)
    file.write(f'\n{i} Quartile: {Qk}')
    return Qk

```

```

import numpy as np
from Deviation import avg
from Deviation import writeFile

def task3(array, file):
    avg_n = avg(array)
    res_array = []
    array_len = len(array)
    A = np.array([[100, 1], [avg_n, 1]])
    X = np.linalg.solve(A, np.array([100, 95]))

    for i in range(array_len):

```

```

        res_array.append((X[0] * array[i] + X[1]))

writeFile(file, f"\nA = {X[0]}\nB = {X[1]}")
writeFile(file, f"\nMarks:{res_array}")

```

```

def task4(array, file):
    array_len = len(array)
    file.write("\nList diagram\n")
    file.write("-" * 20)
    sorted_array = sorted(array)

    prev_n = int(sorted_array[0]/10)

    print(sorted_array)
    file.write(f"\n{prev_n}\t|")
    for i in range(array_len):
        if int(prev_n) == int(sorted_array[i] / 10):
            file.write(f"{sorted_array[i] % 10} ")
        else:
            prev_n = int(sorted_array[i] / 10)
            file.write(f"\n{prev_n}\t|")
            file.write(f"{sorted_array[i] % 10} ")

    file.write("\n")
    file.write("-" * 20)

```

Випробування алгоритму

```

Standard Deviation:15.771493271088822
Middle Deviation:13.460000000000003
90 Percentile: 9.9
1 Quartile: 2.75
3 Quartile: 8.25
A = 0.19379844961240322
B = 80.62015503875968
Marks:[88.3720930232558, 93.21705426356588, 92.63565891472868, 94.18604651162791, 100.0, 98.06201550387597, 93.41085271317829, 94.18604651162791, 99.03100775193798, 96.89922480620154]
List diagram
-----
4  |0
6  |2 5 6
7  |0 0 |
8  |4
9  |0 5
10 |0
-----

```

Завдання 5



Висновки: навчилися використовувати та знаходити данні для лінійних перетворень та для зображення цих даних.