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

Звіт
Лабораторна робота № 3

з дисципліни
«ЙОПІ»

Виконав:	Ільницький Олександр Віталійович	**Перевірила**:	
Група	ІПЗ-24(1)	Дата перевірки	
Форма навчання	денна	Оцінка	
Спеціальність	121		

Завдання

1. Намалюйте діаграму розсіювання для даних. Укажіть, чи існує тренд у даних. Якщо так, то вкажіть, чи є це негативним трендом, чи позитивним.
2. Знайдіть центр ваги і коваріацію.
3. Знайти рівняння лінії регресії у від x.
4. Розрахуйте коефіцієнт кореляції між даними.
5. Зробити висновок про залежності.

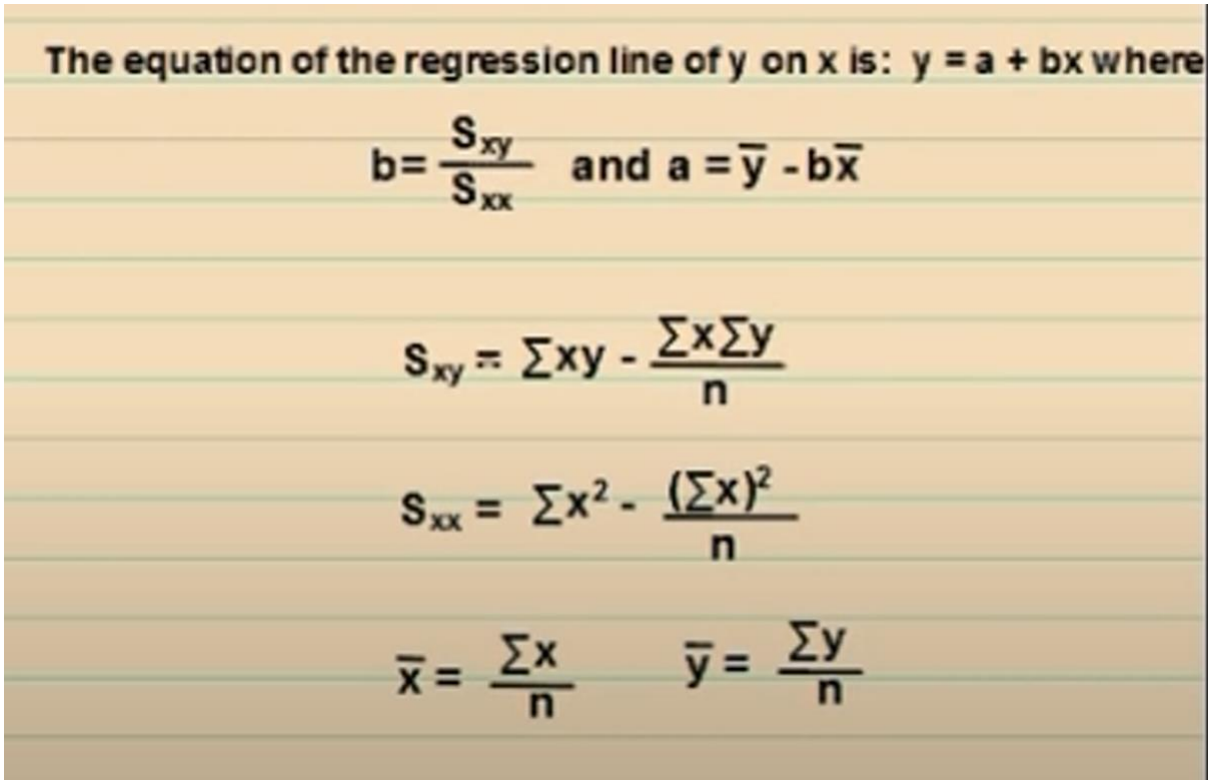
Мета: навчитись використовувати на практиці набуті знання про міри в двовимірній статистиці.
Тема: ДВОВИМІРНА СТАТИСТИКА

Математична модель

1. Центр ваги шукається за формулою середнього значення
2. Коваріацію за допомогою таких формул

$$cov(x;y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_{\text{ср}})(y_i - y_{\text{ср}})$$

Де
Xi, Yi – i-тий елемент
Xср, Yср – середнє значення по X та Y

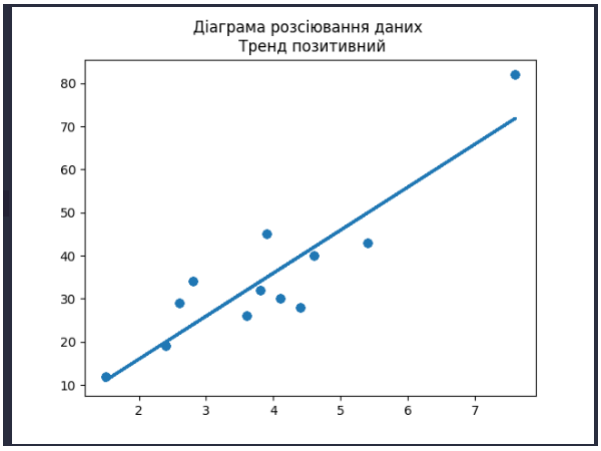
3. 

4. Coorelation = cov(x,y)/(Sx*Sy) де
   Cov(x,y) – коваріація
   Sx Sy – стандартне відхилення x, y

1. Намалюйте діаграму розсіювання для даних. Укажіть, чи існує тренд у даних. Якщо так, то вкажіть, чи є це негативним трендом, чи позитивним

```
import numpy as np
import matplotlib.pyplot as plt


def scatter_diagram(x, y):
    plt.scatter(x, y)
    plt.title("Діаграма розсіювання даних \n Тренд позитивний")
    z = np.polyfit(x, y, 1)
    p = np.poly1d(z)
    plt.plot(x, p(x),  linewidth=2)
    plt.show()
```

Result



Посилання на використані функції
Np.polyfit(args) - https://numpy.org/doc/stable/reference/generated/numpy.polyfit.html
Np.poly1d(args) - https://numpy.org/doc/stable/reference/generated/numpy.poly1d.html

PSEUDOCODE:

```
Function diagram(x,y){

plt.scatter(x, y)
plt.title("Діаграма розсіювання даних \n Тренд позитивний")
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x, p(x),  linewidth=2)
plt.show()
}
```

2. Знайдіть центр ваги і коваріацію

Формула

$$cov(x;y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_{\text{ср}})(y_i - y_{\text{ср}})$$

Де
Xi, Yi – і-тий елемент
Xcp, Ycp – середнє значення по X та Y

```
def task_2(x, y):
    avg_x = sum(x) / float(len(x))
    avg_y = sum(y) / float(len(y))
```

```
    Xi = [i - avg_x for i in x]
    Yi = [i - avg_y for i in y]
    numerator = sum([Xi[i] * Yi[i] for i in range(len(Xi))])
    denom = len(x) - 1
    covariance_res = numerator / denom
    print(f"Covariance: {covariance_res} \nCenter of X: {avg_x}\nCenter of
Y: {avg_y}")
    return covariance_res
```

Result

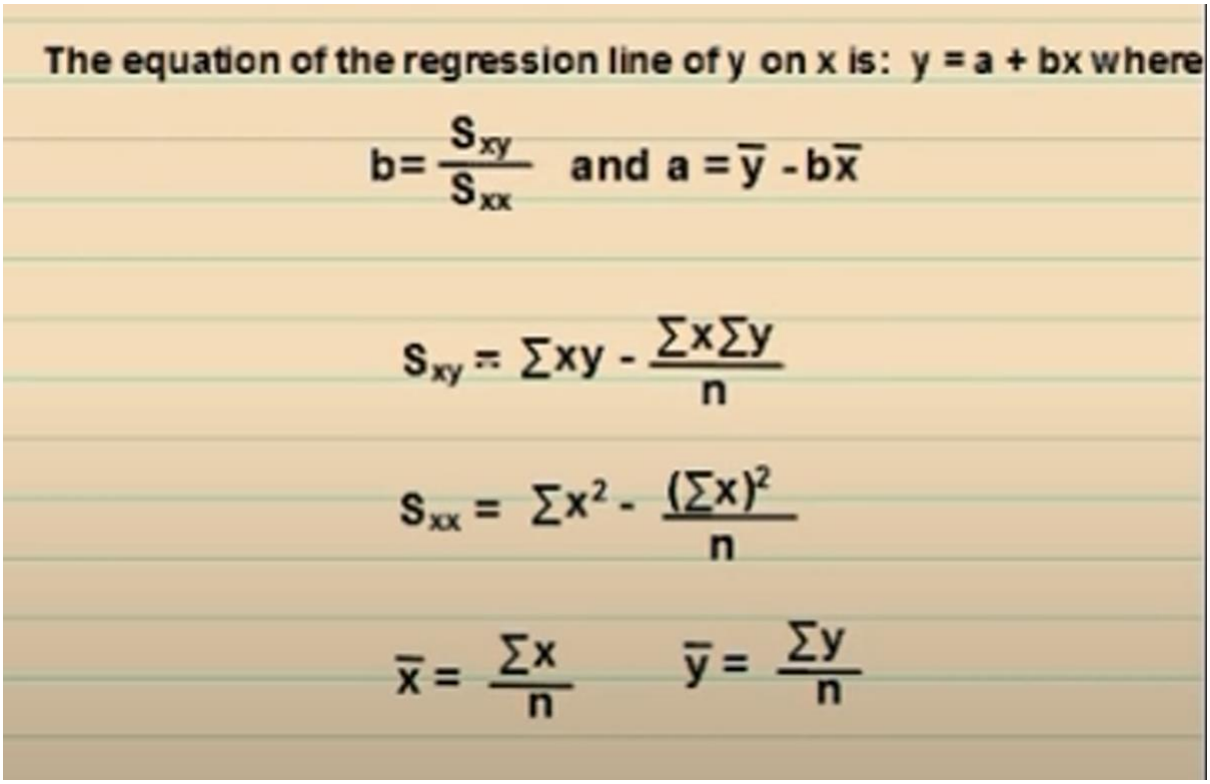
```
Task 2

Covariance: 22.820202020202014

Center of X: 3.856000000000001

Center of Y: 34.5
```

Pseudocode:

```
Function Cov(x, y){
avg x = sum(x) / float(len(x))
avg_y = sum(y) / float(len(y))
Xi = [i - avg_x for i in x]
Yi = [i - avg_y for i in y]
numerator = sum([Xi[i] * Yi[i] for i in range(len(Xi))])
denom = len(x) - 1
covariance_res = numerator / denom
print(f"Covariance: {covariance_res} \nCenter of X: {avg_x}\nCenter of Y: {avg_y}")
return covariance_res
}
```

3. Знайти рівняння лінії регресії y від x

Формула




```
def task_3(x , y , x_or_y = ''):
    x = np.array(x)
    y = np.array(y)
    avg_x = float(avg(x))
    avg_y = float(avg(y))
    # Sxy = sum XY - (sumX)(sumY)/n
    Sxy = np.sum(x * y) - avg_x * avg_y
    #Sxx = sum X^2 - (sumX)^2/n
    Sxx = np.sum(x * x) - avg_x * avg_x
    b = Sxy / Sxx
    a = avg_y - b * avg_x
    print( f"{x_or_y}: {a} + {b}x")
```

Result

```
Task 3


Y: -0.5318246322997737 + 9.085016761488527x
X: 0.19889132514517804 + 0.10600314999579198x
```

Pseudocode:

Function reg_line(x,y){

```
x = np.array(x)
y = np.array(y)
avg_x = float(avg(x))
avg_y = float(avg(y))
# Sxy = sum XY - (sumX)(sumY)/n
Sxy = np.sum(x * y) - avg_x * avg_y
#Sxx = sum X^2 - (sumX)^2/n
Sxx = np.sum(x * x) - avg_x * avg_x
b = Sxy / Sxx
a = avg_y - b * avg_x
print( f"{x_or_y}: {a} + {b}x")
```

}

4. Розрахуйте коефіцієнт кореляції між даними.

Формула

Coorelation = cov(x,y)/(Sx*Sy) де

Cov(x,y) – коваріація

Sx Sy – стандартне відхилення x, y

```
def task_4(x, y, file):
    cov = task_2(x, y)
    standart_x = standart_dev(x, file)
    standart_y = standart_dev(y, file)
    Pxy = cov/(standart_x*standart_y)
    print(f"Correlation: {Pxy}")
```

Pseudocode:

Function Coorelation(x,y):

{

```
cov = cov(x, y)
standart_x = standart_dev(x, file)
standart_y = standart_dev(y, file)
Pxy = cov/(standart_x*standart_y)
print(f"Correlation: {Pxy}")
```

}

Program:

```
def data_append(array, file):
    for i, line in enumerate(file):
        if i != 0:
            line_ = line.strip().replace(',', '.').split('\t')
            array.append([float(num) for num in line_])

def x_y_fill(array, array_to_fill, index):
    for num in (array[i][index] for i in range(len(array))):
        array_to_fill.append(num)
```

```
from array_functin import data_append, x_y_fill
from Diagram import scatter_diagram
from task_2 import task_2
from task_3 import task_3
from task_4 import task_4

file = open('task_03_data/input_100.txt')
answer_file = open('task.txt', 'w')
array = []
x = []
y = []
data_append(array=array, file=file)
x_y_fill(array=array, array_to_fill=x, index=0)
x_y_fill(array=array, array_to_fill=y, index=1)

scatter_diagram(x, y)
print('Task 2')
task_2(x, y)
print('\n\nTask 3\n')
task_3(x, y, 'Y')
print('\n\nTask 4\n')
task_4(x, y, answer_file)
```

```
import numpy as np
import matplotlib.pyplot as plt


def scatter_diagram(x, y):
    plt.scatter(x, y)
    plt.title("Діаграма розсіювання даних \n Тренд позитивний")
    z = np.polyfit(x, y, 1)
    p = np.poly1d(z)
    plt.plot(x, p(x),  linewidth=2)
    plt.show()
```

```
def task_2(x, y):
    avg_x = sum(x) / float(len(x))
    avg_y = sum(y) / float(len(y))
    Xi = [i - avg_x for i in x]
    Yi = [i - avg_y for i in y]
    numerator = sum([Xi[i] * Yi[i] for i in range(len(Xi))])
    denom = len(x) - 1
    covariance_res = numerator / denom
    print(f"Covariance: {covariance_res} \nCenter of X: {avg_x}\nCenter of
```

```
Y: {avg_y}")
    return covariance_res



from usable_funcs import avg
import numpy as np


def task_3(x , y , x_or_y = ''):
    x = np.array(x)
    y = np.array(y)
    avg_x = float(avg(x))
    avg_y = float(avg(y))
    # Sxy = sum XY - (sumX)(sumY)/n
    Sxy = np.sum(x * y) - avg_x * avg_y
    #Sxx = sum X^2 - (sumX)^2/n
    Sxx = np.sum(x * x) - avg_x * avg_x
    b = Sxy / Sxx
    a = avg_y - b * avg_x
    print( f"{x_or_y}: {a} + {b}x")
```

```
from task_2 import task_2
from LAB2.Deviation import standart_dev


def task_4(x, y, file):
    cov = task_2(x, y)
    standart_x = standart_dev(x, file)
    standart_y = standart_dev(y, file)
    Pxy = cov/(standart_x*standart_y)
    print(f"Correlation: {Pxy}")
```

```
def avg(array):
    return sum(array) / len(array)
```

Висновки:

Навчилися будувати діаграму розсіювання та знаходити коефіцієнт кореляції та коваріації