Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**ЛАБОРАТОРНА РОБОТА № 1**

з дисципліни «МОПЕ» на тему

«ЗАГАЛЬНІ ПРИНЦИПИ ОРГАНІЗАЦІЇ ЕКСПЕРИМЕНТІВ З ДОВІЛЬНИМИ ЗНАЧЕННЯМИ ФАКТОРІВ»

ВИКОНАВ:

студент ІІ курсу ФІОТ

групи ІВ-82

Захарчук Даниїл

8210

ПЕРЕВІРИВ:

вик. Регіда П. Г.

Київ – 2020

**Мета:** Вивчити основні поняття, визначення, принципи теорії планування експерименту, на основі яких вивчити побудову формалізованих алгоритмів проведення експерименту і отримання формалізованої моделі об’єкта. Закріпити отримані знання практичним їх використанням при написанні програми, що реалізує завдання на лабораторну роботу.

**Завдання:**

1. Використовуючи програму генерації випадкових чисел, провести трьохфакторний експеримент в восьми точках (три стовбці і вісім рядків в матриці планування – заповнити її випадковими числами). Рекомендовано взяти обмеження до 20 при генерації випадкових чисел, але врахувати можливість зміни обмеження на вимогу викладача. Програма створюється на основі будь-якої мови високого рівня.
2. Визначити значення функції відгукув для кожної точки плану за формулою лінійної регресії: Y =a0 + a1 X1 + a2 X2 + a3 X3, де a0, a1, a2, a3 довільно вибрані (для кожного студента різні) коефіцієнти, постійні протягом усього часу проведення експерименту.
3. Виконати нормування факторів. Визначити значення нульових рівнів факторів. Знайти значення відгуку для нульових рівнів факторів і прийняти його за еталонне Уэт.
4. Знайти точку плану, що задовольняє критерію вибору оптимальності (див. табл.1). Варіанти обираються по номеру в списку в журналі викладача.

**Варіант завдання:**



**Роздруківка тексту програми:**

import random

from prettytable import PrettyTable

from numpy import mean

from typing import List

class FactorExperiment:

def \_\_init\_\_(self, n: int, a: List[float]) -> None:

self.n = n # Num of dotes

self.a = a # List of coefs

# Factors

self.X\_1 = self.random\_list()

self.X\_2 = self.random\_list()

self.X\_3 = self.random\_list()

self.factors = [self.X\_1, self.X\_2, self.X\_3]

# Normalized Factors

self.Xn\_1 = self.get\_xn(self.X\_1)

self.Xn\_2 = self.get\_xn(self.X\_2)

self.Xn\_3 = self.get\_xn(self.X\_3)

# Form regression and sort Y[i]

self.Y = self.form\_regression()

self.sorted\_Y = sorted(self.Y)

# Form zero Factors

self.x0 = self.form\_x0()

# Form Factors change interval

self.dx = self.form\_dx()

# Form Y etalon

self.Y\_et = self.form\_Y\_et()

# Form answer by variant |<-mean(Y)|

self.answear = self.form\_answear()

# Form Plans point

self.point = self.form\_point()

def random\_list(self) -> List[float]:

''' Gen Factor vector[n] '''

return [random.randint(0, 20) for \_ in range(self.n)]

def get\_xn(self, X: List[float]) -> List[float]:

''' Normalize Factor vector X '''

a = sorted(X)

x0 = (a[-1] + a[0]) / 2

dx = x0 - a[0]

return [round((X[i] - x0) / dx, 2) for i in range(self.n)]

def form\_regression(self) -> List[float]:

''' Get regression with given Factors '''

return [self.a[0] + self.a[1] \* self.X\_1[i] +

self.a[2] \* self.X\_2[i] + self.a[3] \* self.X\_3[i]

for i in range(self.n)]

def form\_x0(self) -> List[float]:

''' Calculate x0 by given Factors '''

return [round((X[0] + X[-1]) / 2, 2)

for X in self.factors]

def form\_dx(self) -> List[float]:

''' Calculate dx by given Factors '''

return [(self.x0[i] - sorted(self.factors[i])[0])

for i in range(len(self.factors))]

def form\_Y\_et(self) -> float:

''' Criteria by variant '''

return mean(self.a[0] + self.a[1] \* self.x0[0] +

self.a[2] \* self.x0[1] + self.a[3] \* self.x0[2])

def form\_answear(self) -> float:

''' Get answear by criteria Y\_et '''

return [self.sorted\_Y[i] for i in range(self.n)

if self.sorted\_Y[i] > self.Y\_et][0]

def form\_point(self) -> List[float]:

''' Get point which satisfied criteria answear '''

return [(self.X\_1[self.Y.index(self.sorted\_Y[i])],

self.X\_2[self.Y.index(self.sorted\_Y[i])],

self.X\_3[self.Y.index(self.sorted\_Y[i])])

for i in range(self.n) if self.sorted\_Y[i] > self.Y\_et][0]

def check\_results(self) -> None:

''' Print results to console '''

experiment = PrettyTable()

experiment.field\_names = ['№', 'X1', 'X2', 'X3', 'Y',

'Хn1', 'Хn2', 'Хn3']

for i in range(self.n):

experiment.add\_row([i+1, self.X\_1[i], self.X\_2[i], self.X\_3[i],

self.Y[i], self.Xn\_1[i], self.Xn\_2[i],

self.Xn\_3[i]])

characteristic = PrettyTable()

characteristic.field\_names = ['Characteristic', 'X1', 'X2', 'X3']

characteristic.add\_row(['x0', self.x0[0], self.x0[1], self.x0[2]])

characteristic.add\_row(['dx', self.dx[0], self.dx[1], self.dx[2]])

result = PrettyTable()

result.field\_names = ['Result', 'Value']

result.add\_row(['Y', f'{self.a[0]} + {self.a[1]}\*x1 + {self.a[2]}\*x2 + {self.a[3]}\*x3'])

result.add\_row(['Y\_etalon', f'{self.answear}'])

result.add\_row(['Point', f'{self.point}'])

print(experiment)

print(characteristic)

print(result)

if \_\_name\_\_ == '\_\_main\_\_':

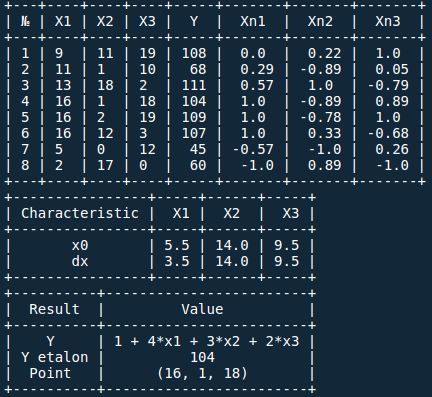
n = 8

a = [1, 5, 3, 2]

lab\_1 = FactorExperiment(n, a)

lab\_1.check\_results()

**Результат:**



**При a = [1, 4, 3, 2]**

**Висновки:**

Я вивчив основні поняття, визначення, принципи теорії планування експерименту, на основі яких вивчив побудову формалізованих алгоритмів проведення експерименту і отримання формалізованої моделі об’єкта. Закріпив отримані знання практичним їх використанням при написанні програми, що реалізує завдання на лабораторну роботу.