*Додаток 1*

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 5
з дисципліни «Основи програмування – 2.
Метидології програмування»

«Успадкування та поліморфізм»

Варіант 9

Виконав студент ІП-13 Григоренко Родіон Ярославович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вечерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

# Лабораторна робота

# 5 Успадкування та поліморфізм

9. Створити клас TMatrix, який представляє матрицю і містить методи для обчислення детермінанта та суми елементів матриці. На основі цього класу створити класи-нащадки, які представляють квадратні матриці 2-го та 3-го порядку. За допомогою цих класів обчислити вираз

$$S = \left( \sum_{i=1}^{3} \sum_{j=1}^{3} a_{ij} \right) + |A| + |B|,$$

де $A = \|a_{ij}\|_1^3$ – матриця 3-го порядку, а $B = \|b_{ij}\|_1^2$ – матриця 2-го порядку.

## Варіант 9

## Код програми

**C++**

## Lab5_second_semestr.cpp

```
#include "Header.h"
int main()
{
    srand(time(NULL));
    DoubleMatrix db;
    cout << "Double Matrix : \n";
    db.show_matrix();
    cout  << "Sum of elements: " << db.Sum() << "  Determinant: " << db.determinant() << "\n\n";
    TripleMatrix tp;
    cout << "Triple Matrix : \n";
    tp.show_matrix();
    cout << "Sum of elements: " << tp.Sum() << "  Determinant: " << tp.determinant() << "\n\n";
    cout << "The result of function is: " << result(db, tp);
}
```

## Header.h

```
#pragma once
#include <iostream>
#include <string>
#include <vector>
#include <stdlib.h>

using namespace std;

class TMatrix {
protected:
int** Matrix;
int size;
public:
TMatrix();
int** get_matrix();
int get_size();
int Sum();
virtual int determinant() = 0;
void show_matrix();
};
```

```
class DoubleMatrix : public TMatrix {
public:
DoubleMatrix();
int determinant() override ;
};

class TripleMatrix : public TMatrix {
public:
TripleMatrix();
int determinant() override ;
};

int result(DoubleMatrix, TripleMatrix);

Person* create_array(int len);

vector<Person> find_happy_dates(Person*, int);

bool ben(Person);

void output(vector<Person>);

vector<string> split(string, char);
```

**Sourse.cpp**

```
#include "Header.h"

TMatrix::TMatrix() {}

//int TMatrix::Sum() { return 0; }
int** TMatrix::get_matrix() { return Matrix; }

int TMatrix::get_size() { return size; }

DoubleMatrix::DoubleMatrix() {
size = 2;
Matrix = new int* [size];
for (int i = 0; i < size; i++) {
Matrix[i] = new int[size];
for (int j = 0; j < size; j++) {
Matrix[i][j] = rand() % 20 - 10;
}
}
}

TripleMatrix::TripleMatrix() {
size = 3;
Matrix = new int* [size];
for (int i = 0; i < size; i++) {
Matrix[i] = new int[size];
for (int j = 0; j < size; j++) {
Matrix[i][j] = rand() % 20 - 10;
}
}
}

int TMatrix::Sum() {
int sum = 0;
for (int i = 0; i < size; i++) {
for (int j = 0; j < size; j++) {
sum += Matrix[i][j];
}
}
```

```
return sum;
}


int DoubleMatrix::determinant(){
return Matrix[0][0] * Matrix[1][1] - Matrix[1][0] * Matrix[0][1];
}


void TMatrix::show_matrix() {
cout << "\n";
for (size_t i = 0; i < size; i++)
{
for (size_t j = 0; j < size; j++) {
cout << Matrix[i][j] << "  ";
}
cout << "\n\n";
}
cout << "\n";
}

int TripleMatrix::determinant() {
return Matrix[0][0] * Matrix[1][1]*Matrix[2][2] + Matrix[0][1] * Matrix[2][0] *
Matrix[1][2] + Matrix[1][0] * Matrix[2][1] * Matrix[0][2] - Matrix[2][0] *
Matrix[1][1] * Matrix[0][2] - Matrix[0][0] * Matrix[1][2] * Matrix[2][1] -
Matrix[2][2] * Matrix[1][0] * Matrix[0][1];
}

int result(DoubleMatrix db, TripleMatrix tp) {
int res = tp.Sum() + tp.determinant() + db.determinant();
return res;
}
```

**Тестування:**

```
Double Matrix :

5   -5

9   -10


Sum of elements: -1   Determinant: -5

Triple Matrix :

-8   -6   5

-7   -7   -7

2   -2   8


Sum of elements: -22   Determinant: 448

The result of function is: 421
```

## PYTHON

### main.py

```
from Functions import *

db = DoubleMatrix()
print("Double Matrix : \n")
db.show_matrix()
print("Sum of elements: " + str(db.Sum()) + "  Determinant: " +
str(db.determinant()) + "\n\n")

tp = TripleMatrix()
print("Triple Matrix : \n")
tp.show_matrix()
print("Sum of elements: " + str(tp.Sum()) + "  Determinant: " +
str(tp.determinant()) + "\n\n")
print("The result of function is: " + str(result(db, tp)))
```

### Functions.py

```
import random
```

```
class TMatrix:
    def __init__(self):
        self.size = 0
        Matrix = []

    def show_matrix(self):
        string = ""
        for i in range(self.size):
            for j in range(self.size):
                string += str(self.Matrix[i][j]) + '  '
            string += '\n'
        print(string)

    def Sum(self):
        res = 0
        for i in range(self.size):
            for j in range(self.size):
                res += self.Matrix[i][j]
        return res

    def determinant(self):
        pass


class DoubleMatrix(TMatrix):
    #Matrix = [[0, 0], [0, 0]]
    def __init__(self):
        self.size = 2
        self.Matrix = [[0, 0], [0, 0]]
        for i in range(self.size):
            for j in range(self.size):
                self.Matrix[i][j] = random.randint(-10, 10)

    def determinant(db):
        return db.Matrix[0][0] * db.Matrix[1][1] - db.Matrix[1][0]
* db.Matrix[0][1]


class TripleMatrix(TMatrix):
    size = 3
    Matrix = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
    def __init__(self):
        for i in range(self.size):
            for j in range(self.size):
                self.Matrix[i][j] = random.randint(-10, 10)

    def determinant(self):
```

```
        return self.Matrix[0][0] * self.Matrix[1][1] *
self.Matrix[2][2] + self.Matrix[0][1] * self.Matrix[2][0] * \
               self.Matrix[1][2] + self.Matrix[1][0] * \
               self.Matrix[2][1] * self.Matrix[0][2] -
self.Matrix[2][0] * self.Matrix[1][1] * self.Matrix[0][2] - \
               self.Matrix[0][0] * self.Matrix[1][2] * \
               self.Matrix[2][1] - self.Matrix[2][2] *
self.Matrix[1][0] * self.Matrix[0][1]


def result(db, tp):
    res = tp.Sum() + tp.determinant() + db.determinant()
    return res
```

**Тестування:**

```
Double Matrix :

10  -3
10  7

Sum of elements: 24  Determinant: 100


Triple Matrix :

-1  -3  8
0  -10  2
1  0  -3

Sum of elements: -6  Determinant: 44


The result of function is: 138
```

**Висновки:**

Я вивчив особливості успадкування та поліморфізму. Застосував ці навички на практиці.