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. Модульне програмування»

«Успадкування та поліморфізм»

Варіант 8

Виконав студент ІП-15, Дацьо Іван Іванович

(шифр, прізвище, ім'я, по батькові)

Перевірила Вєчерковська Анастасія Сергіївна

(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 5

Успадкування та поліморфізм

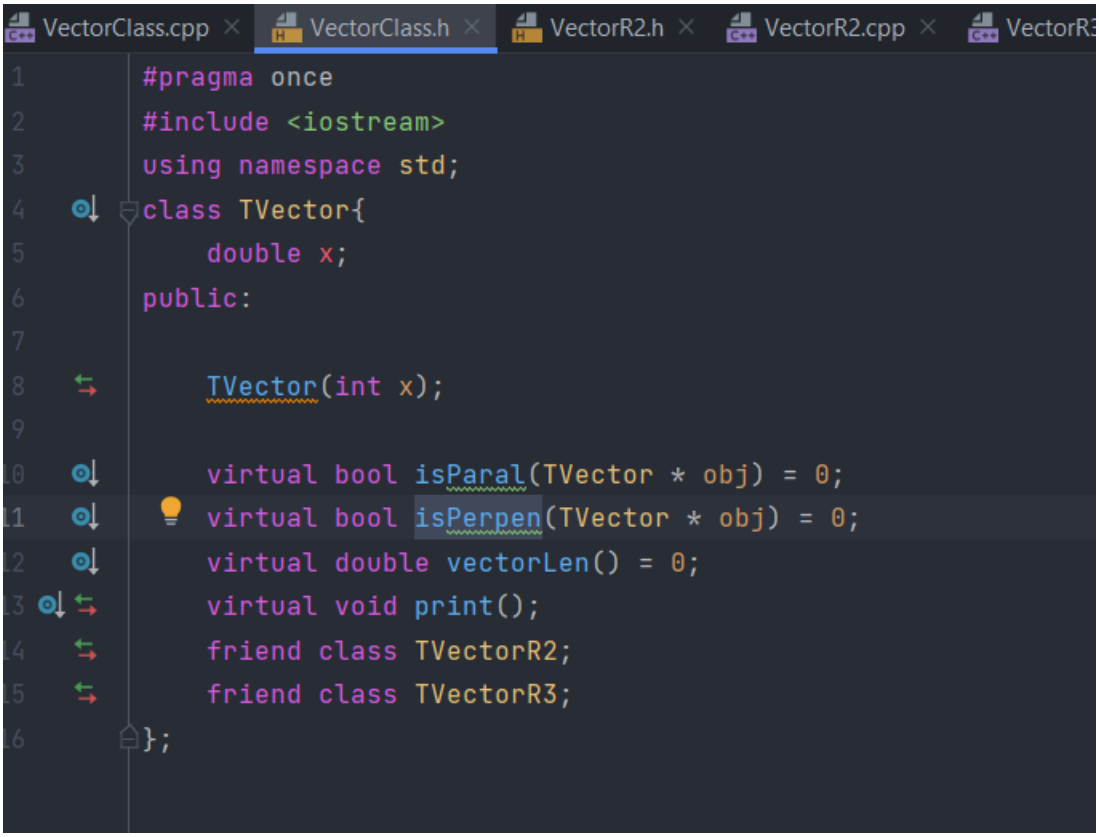
Індивідуальне завдання

Варіант 8

8. Створити клас TVector, який представляє вектор і містить методи для визначення того, чи є інший вектор паралельним / перпендикулярним до нього та метод знаходження довжини вектора. На основі цього класу створити класи-нащадки, які представляють вектори з просторів R^2 та R^3. Створити 3 двовимірні та 4 тривимірні вектори. Знайти суму довжин векторів, паралельних до першого по порядку двовимірного вектора, та суму векторів, перпендикулярних до першого по порядку тривимірного вектора.

Код C++

```cpp
#include "VectorClass.h"
#include <cmath>

TVector::TVector(int x) {
    this->x = x;
}
void TVector::print() {
    cout << "X = "<<x;
}

```


VectorClass.cpp
VectorClass.h
VectorR2.h
VectorR2.cpp
#pragma once
#include <iostream>
using namespace std;
class TVector{
double x;
public:
TVector(int x);
virtual bool isParal(TVector * obj) = 0;
virtual bool isPerpen(TVector * obj) = 0;
virtual double vectorLen() = 0;
virtual void print();
friend class TVectorR2;
friend class TVectorR3;
};

VectorClass.cpp | VectorClass.h | VectorR2.h | VectorR2.cpp | Vecto

```cpp
#pragma once
#include "VectorClass.h"
#include <cmath>
class TVectorR2 :public TVector
{
    double y;
public:
    TVectorR2(double x , double y);
    void print() override;
    bool isPerpen(TVector * obj) override;
    bool isParal(TVector * obj) override;
    double vectorLen() override;


};
```

VectorClass.cpp | VectorClass.h | VectorR2.h | VectorR2.cpp | VectorR3.cpp | VectorR3.h

```cpp

#include "VectorR2.h"

TVectorR2::TVectorR2(double x, double y): TVector(x)
{
    this->y = y;
}
bool TVectorR2::isParal(TVector *obj) {
    double tmp = x * ((TVectorR2*)obj)->y - y * obj->x;
    return !tmp;
}
bool TVectorR2::isPerpen(TVector *obj) {
    double tmp = x *obj->x + y * ((TVectorR2 *)obj)->y;
return !tmp;
}

double TVectorR2::vectorLen() {
    return sqrt( x: x*x + y*y);
}
void TVectorR2::print() {
    TVector::print();
    cout <<" Y = "<<y<<endl;
}


```

VectorClass.cpp | VectorClass.h | VectorR2.h | VectorR2.cpp | VectorR3.cpp | VectorR3.h | foo.cpp | foo.h | main.cpp

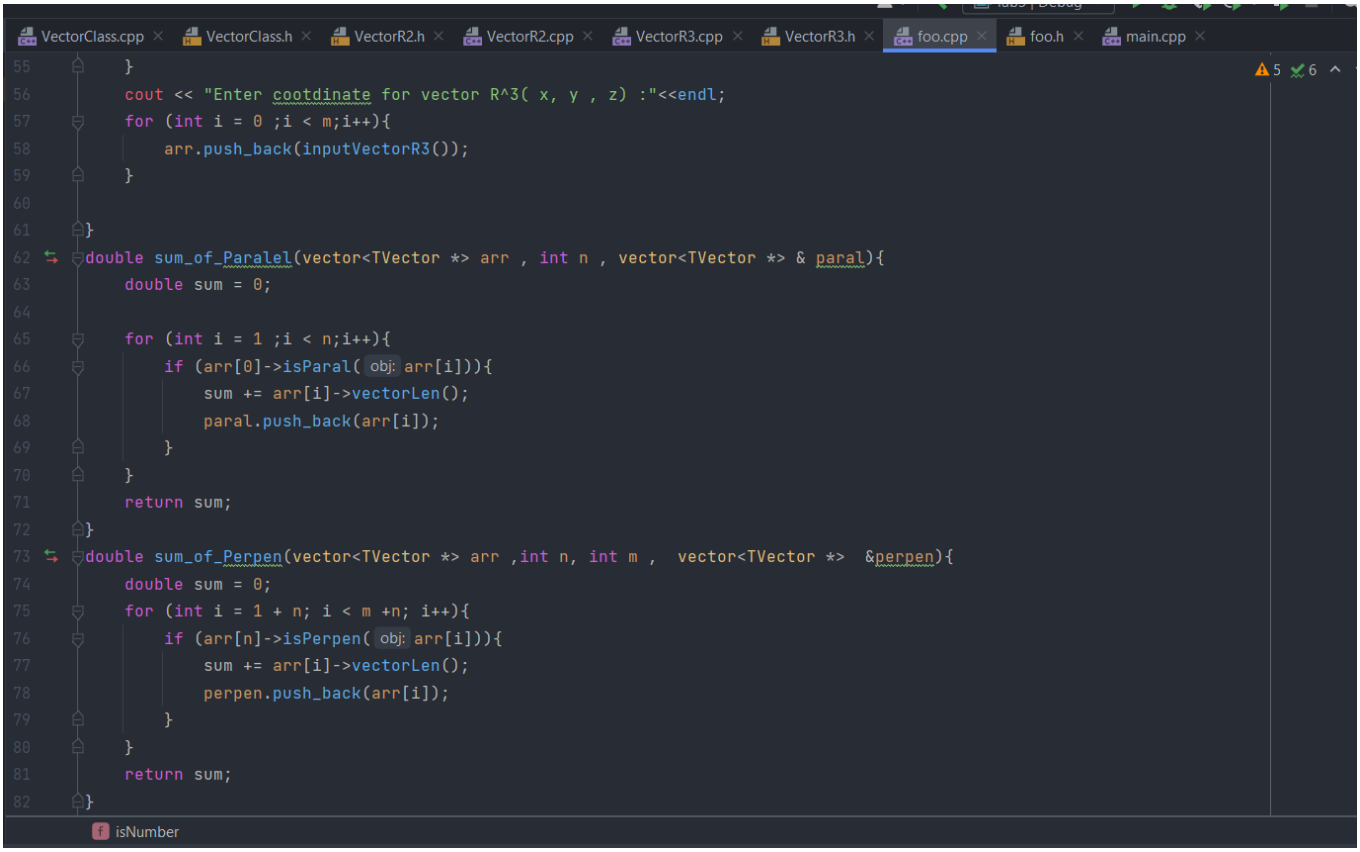
```cpp
#include "VectorR3.h"


TVectorR3::TVectorR3(double x, double y , double z): TVector(x)
{
    this->y = y;
    this->z = z;
}
bool TVectorR3::isParal(TVector *obj) {
    double tmp = (y * ((TVectorR3*)obj)->z - ((TVectorR3*)obj)->y * z) - (x * ((TVectorR3*)obj)->z - obj->x * z) + (x * ((TVectorR3
    return !tmp;
}
bool TVectorR3::isPerpen(TVector *obj) {
    double tmp = x * obj->x + y * ((TVectorR3*)obj)->y + z * ((TVectorR3*)obj)->z;
    return !tmp;
}

double TVectorR3::vectorLen() {
    return sqrt( x: x*x + y*y + z*z);
}
void TVectorR3::print() {
    TVector::print();
    cout <<" Y = "<<y;
    cout << " Z = "<<z<<endl;
}

```

VectorClass.cpp | VectorClass.h | VectorR2.h | VectorR2.cpp | VectorR3.cpp | VectorR3.h | foo.cpp

```cpp
#pragma once
#include "VectorClass.h"
#include <cmath>
class TVectorR3 :public TVector
{
    double y;
    double z;
public:
    TVectorR3(double x , double y , double z);
    void print() override;
    bool isPerpen(TVector * obj) override;
    bool isParal(TVector * obj) override;
    double vectorLen() override;


};
```

VectorClass.cpp | VectorClass.h | VectorR2.h | VectorR2.cpp | VectorR3.cpp | VectorR3.h | foo.cpp | foo.h | main.cpp

```cpp
#include "foo.h"
TVectorR2* inputVectorR2(){

    double x , y;
    cout <<"Enter x:"<<endl;
    //cin >> x;
    x = valNumber();
    cout <<"Enter y:"<<endl;
    y = valNumber();
    TVectorR2 * tmp = new TVectorR2(x ,y);
    return tmp;
}
TVectorR3* inputVectorR3(){

    double x , y ,z;
    cout <<"Enter x:"<<endl;
    x = valNumber();
    cout <<"Enter y:"<<endl;
    y = valNumber();
    cout <<"Enter z:"<<endl;
    z = valNumber();
    TVectorR3 * tmp = new TVectorR3(x ,y ,z);
    return tmp;
}
double valNumber(){// Enter add number
    string n;
    cin >> n;

    while (!isNumber( s: n) ){
        cout <<"Enter correct number :"<<endl;
        cin >> n;
    }
    return stod( str: n);
}
bool isNumber(string s)// check if it is int number
{
    int count = 0;
    for (char ch : s) {
        if (ch =='.'){
            count ++;
            if(count > 1){
                return false;
            }
        }
        else if (!isdigit( c: ch) && ch != '-') return false;
    }

    return true;
}

void create_arr(vector<TVector *> & arr , int n,int m ){
    cout << "Enter cootdinate for vector R^2 :"<<endl;
    for (int i = 0 ;i < n;i++){
        arr.push_back(inputVectorR2());
    }
```


VectorClass.cpp
VectorClass.h
VectorR2.h
VectorR2.cpp
VectorR3.cpp
VectorR3.h
foo.cpp
foo.h
main.cpp
}
cout << "Enter cootdinate for vector R^3(x, y , z) :"<<endl;
for (int i = 0 ;i < m;i++){
arr.push_back(inputVectorR3());
}
}
double sum_of_Paralel(vector<TVector *> arr , int n , vector<TVector *> & paral){
double sum = 0;
for (int i = 1 ;i < n;i++){
if (arr[0]->isParal(obj: arr[i])){
sum += arr[i]->vectorLen();
paral.push_back(arr[i]);
}
}
return sum;
}
double sum_of_Perpen(vector<TVector *> arr ,int n, int m , vector<TVector *> &perpen){
double sum = 0;
for (int i = 1 + n; i < m +n; i++){
if (arr[n]->isPerpen(obj: arr[i])){
sum += arr[i]->vectorLen();
perpen.push_back(arr[i]);
}
}
return sum;
}
isNumber

```
                sum += arr[i]->VectorLen();
                perpen.push_back(arr[i]);
            }
        }
        return sum;
    }
    void out_All(vector<TVector *> arr , int n ,int  m){
        cout << "Vectors R^2 :"<<endl;
        for (int i = 0; i < n; i++){
            arr[i]->print();
        }
        cout <<"Vectors R^3 :"<<endl;
        for(int i = n ; i < m+n; i++){
            arr[i]->print();
        }
    }
    void out_to_zero(vector<TVector *> arr , string text){
        cout <<text<<endl;
        for (int i = 0; i < arr.size(); i++){
            arr[i]->print();
        }
    }
```

VectorClass.cpp | VectorClass.h | VectorR2.h | VectorR2.cpp | VectorR3.cpp | VectorR3.h | foo.cpp | foo.h | main.cpp

```
#pragma once
#include "VectorR3.h"
#include "VectorR2.h"
#include "VectorClass.h"
#include <iostream>
#include <string>
#include <vector>
TVectorR2* inputVectorR2();
TVectorR3* inputVectorR3();
double valNumber();
bool isNumber(string s);
void create_arr(vector<TVector *> & arr , int n,int m );
double sum_of_Paralel(vector<TVector *> arr , int n , vector<TVector *> & paral);
double sum_of_Perpen(vector<TVector *> arr , int n, int m ,  vector<TVector *>  &perpen);
void out_All(vector<TVector *> arr , int n ,int  m);
void out_to_zero(vector<TVector *> arr , string text);
```

```cpp
#include <iostream>
#include "foo.h"
int main () {
    vector<TVector *> arr;
    vector<TVector *> paral;
    vector<TVector *> perpen;
    create_arr( &: arr , n: 3, m: 4);
    out_All(arr , n: 3 , m: 4);
    cout << "sum of parallel vectors to 0-vec :"<<sum_of_Paralel(arr, n: 3 , &: paral)<<endl;
    cout << "sum of perpendicular vectors to 0-vec:"<<sum_of_Perpen(arr , n: 3, m: 4 , &: perpen)<<endl;
    out_to_zero( arr: paral , text: "All vectors that paral. to 0-vec :");
    out_to_zero( arr: perpen , text: "All vectors that perpen. to 0-vec :");
}
```

Код Python

```python
from foo import *
n = 3
m =4
arr = create_arr(n, m)
out_vectors(arr, n, m)
sum_of_par, par = sum_of_Paralel(arr, n)
sum_of_per, per = sum_of_Perpen(arr, n, m)
print(f"sum of parallel vectors to 0-vec R^2: {sum_of_par}")
print(f"sum of parallel vectors to 0-vec R^3: {sum_of_per}")
out_to_zero(par,"All vectors that paral. to 0-vec :")
out_to_zero(per, "All vectors that perpen. to 0-vec :")
```

main.py | tvector.py | tvector2R.py | tvectorR3.py | foo.py

```python
from abc import ABC, abstractmethod

class TVector(ABC):

    def __init__(self, x):
        self.x = x

    def out(self):
        print(f"X ={self.x}", end='')

    @abstractmethod
    def isParal(self, vector):
        pass

    @abstractmethod
    def isPerpen(self, vector):
        pass

    @abstractmethod
    def vectorLen(self):
        pass
```

main.py | tvector.py | tvector2R.py | tvectorR3.py | foo.py

```python
from tvector import TVector
from math import sqrt


class TVectorR2(TVector):
    def __init__(self, x, y):
        super().__init__(x)
        self.__y = y

    def out(self):
        super().out()
        print(f" Y ={self.__y}\n", end_='')

    def isParal(self, vector):
        tmp = self.x * vector.__y - self.__y * vector.x
        if tmp == 0:
            return True
        return False

    def isPerpen(self, vector):
        tmp = self.x * vector.x + self.__y * vector.__y
        if tmp == 0:
            return True
        return False

    def vectorLen(self):
        return sqrt(self.x ** 2 + self.__y ** 2)
```

main.py | tvector.py | tvector2R.py | tvectorR3.py | foo.py

```python
from tvector import TVector
from math import sqrt
class TVectorR3(TVector):
    def __init__(self, x, y, z):
        super().__init__(x)
        self.__y = y
        self.__z = z

    def out(self):
        super().out()
        print(f" Y ={self.__y} Z = {self.__z}\n", end = '')

    def isParal(self, vector):
        tmp = (self.__y * vector.__z - vector.__y * self.__z) - (self.x * vector.__z - vector.x * self.__z) + (self.x * vector.__
        if tmp == 0:
            return True
        return False

    def isPerpen(self, vector):
        tmp = self.x * vector.x + self.__y * vector.__y + self.__z * vector.__z
        if tmp == 0:
            return True
        return False

    def vectorLen(self):
        return sqrt(self.x ** 2 + self.__y ** 2 + self.__z ** 2)
```

main.py | tvector.py | tvector2R.py | tvectorR3.py | foo.py

```python
from tvector2R import TVectorR2
from tvectorR3 import TVectorR3


def create_arr(n, m):
    arr = []
    print("Enter vector R^2 :")
    for i in range(0, n):
        print(f"Enter {i+1} vector coordinates:")
        x = valNumber("Enter X :")
        y = valNumber("Enter Y :")
        arr.append(TVectorR2(x, y))
    print("Enter vector R^3 :")
    for i in range(0, m):
        print(" ")
        x = valNumber("Enter X :")
        y = valNumber("Enter Y :")
        z = valNumber("Enter Z :")
        arr.append(TVectorR3(x, y, z))
    return arr


def valNumber(text):
    isValid = False
    while not isValid:
        try:
            number = float(input(text))
        except ValueError:
            print("Enter correct number :")
```

create_arr() › for i in range(0, n)

main.py | tvector.py | tvector2R.py | tvectorR3.py | foo.py

```
            except ValueError:
                print("Enter correct number :")
            else:
                isValid = True
        return number


def sum_of_Paralel(arr, n):
    par = []
    sum = 0
    for i in range(1, n):
        if arr[0].isParal(arr[i]):
            sum += arr[i].vectorLen()
            par.append(arr[i])
    return sum, par


def sum_of_Perpen(arr, n, m):
    per = []
    sum = 0
    for i in range(n + 1, n + m):
        if arr[n].isPerpen(arr[i]):
            per.append(arr[i])
            sum += arr[i].vectorLen()
    return sum, per


def out_vectors(arr, n, m):
    print("Vectors R^2")
```

```
    return sum, per


def out_vectors(arr, n, m):
    print("Vectors R^2")
    for i in range(0, n):
        arr[i].out()
    print("Vectors R^3 :")
    for i in range(n, m + n):
        arr[i].out()


def out_to_zero(arr, text):
    print(text)
    for i in range(0, len(arr)):
        arr[i].out()
```

Тестування C++

```
C:\Op-1.2\2-semlab\lab5\cmake-build-debug\lab5.exe
Enter cootdinate for vector R^2 :
Enter x:
2
Enter y:
4
Enter x:
1
Enter y:
2
Enter x:
-4
Enter y:
9
Enter cootdinate for vector R^3( x, y , z) :
Enter x:
1
Enter y:
0
Enter z:
-3
Enter x:
7
Enter y:
8
Enter z:
-8
Enter x:
0
Enter y:
```

```
1
Vectors R^2 :
X = 2 Y = 4
X = 1 Y = 2
X = -4 Y = 9
Vectors R^3 :
X = 1 Y = 0 Z = -3
X = 7 Y = 8 Z = -8
X = 0 Y = 2 Z = 0
X = 3 Y = 0 Z = 1
sum of parallel vectors to 0-vec :2.23607
sum of perpendicular vectors to 0-vec:5.16228
All vectors that paral. to 0-vec :
X = 1 Y = 2
All vectors that perpen. to 0-vec :
X = 0 Y = 2 Z = 0
X = 3 Y = 0 Z = 1
```

Тестування Python

```
Enter X :7
Enter Y :8
Enter Z :-8

Enter X :0
Enter Y :2
Enter Z :0

Enter X :3
Enter Y :0
Enter Z :1
Vectors R^2
X =2.0 Y =4.0
X =1.0 Y =2.0
X =-4.0 Y =9.0
Vectors R^3 :
X =1.0 Y =0.0 Z = -3.0
X =7.0 Y =8.0 Z = -8.0
X =0.0 Y =2.0 Z = 0.0
X =3.0 Y =0.0 Z = 1.0
sum of parallel vectors to 0-vec R^2: 2.23606797749979
sum of parallel vectors to 0-vec R^3: 5.16227766016838
All vectors that paral. to 0-vec :
X =1.0 Y =2.0
All vectors that perpen. to 0-vec :
X =0.0 Y =2.0 Z = 0.0
X =3.0 Y =0.0 Z = 1.0

Process finished with exit code 0
```