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.

Методології програмування»

«Успадкування та поліморфізм»

Варіант 31

Виконав студент ІП-11 Трикош Іван Володимирович

Перевірила Вітковська Ірина Іванівна

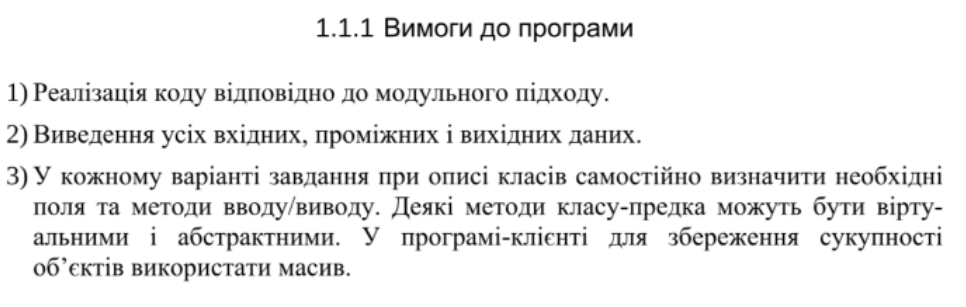
Київ 2022

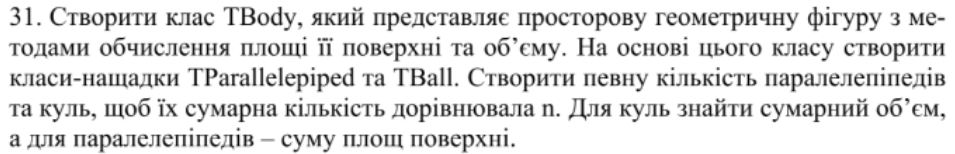
**Лабораторна робота №5**

**Успадкування та поліморфізм**

**Мета –** вивчити механізми створення і використання класів та об’єктів.

Варіант 31.





**Постановка задачі.** Спочатку просимо користувача ввести загальну кількість фігур та кількість куль. Описуємо класи TBody, TParallelepiped та TBall. Два останні класи унаслідуємо від першого. У класах описуємо атрибути сторін, перевантажуємо методи отримання площі та об’єму. Далі створюємо масиви об’єктів класів TBall та TParallelepiped за вказаними сторонами. Після цього обчислюємо загальних об’єм куль та загальну площу поверхонь паралелепіпедів. Виводимо результат.

**Код та результат програми на С++:**

**Function.h**

#pragma once

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <Windows.h>

#include <string>

#include <cmath>

using namespace std;

class TBody

{

public:

TBody(); // Конструктор без параметрів

TBody(string, double\*); // Конструктор для створення та ініціалізації об'єктів

virtual double get\_area(); // Метод отримання площі поверхні фігури

virtual double get\_volume(); // Метод отримання об'єму фігури

string get\_name(); // Метод отримання назви фігури

protected:

string name; // Назва фігури

private:

double\* arrayofsides; // Покажчик на масив сторін фігури

};

class TParallelepiped : public TBody

{

public:

TParallelepiped(); // Конструктор без параметрів

TParallelepiped(double, double, double); // Конструктор для створення та ініціалізації об'єктів

void operator = (const TParallelepiped&); // Перевантажуємо "="

double get\_area() override; // Метод отримання площі поверхні прямокутного паралелепіпеда

double get\_volume() override; // Метод отримання об'єму прямокутного паралелепіпеда

private:

double a, b, c; // Сторони паралелепіпеда

};

class TBall : public TBody

{

public:

TBall(); // Конструктор без параметрів

TBall(double); // Конструктор для створення та ініціалізації об'єктів

void operator = (const TBall&); // Перевантажуємо "="

double get\_area() override; // Метод отримання площі поверхні кулі

double get\_volume() override; // Метод отримання об'єму кулі

private:

double r; // Радіус кулі

};

void input\_number(int&, int&); // Вводимо кількість фігур

TBall\* input\_ball(int&); // Вводимо кулі

TParallelepiped\* input\_parallelepiped(int); // Вводимо прямокутні паралелепіпеди

void find\_area\_volume(TBall\*, TParallelepiped\*, int, int); // Знаходимо та виводимо суму об'ємів та площ поверхонь фігур

**Laba-C++.cpp**

#include "Function.h"

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n; // Кількість фігур

int n\_ball; // Кількість куль

input\_number(n, n\_ball); // Вводимо кількість фігур

TBall\* arrayofballs = input\_ball(n\_ball); // Вводимо кулі

TParallelepiped\* arrayofparallelepiped = input\_parallelepiped(n - n\_ball); // Вводимо прямокутні паралелепіпеди

find\_area\_volume(arrayofballs, arrayofparallelepiped, n\_ball, n - n\_ball); // Знаходимо суму об'ємів та площ поверхонь фігур

// Очищуємо пам'ять

delete[] arrayofballs;

delete[] arrayofparallelepiped;

return 0;

}

**Function.cpp**

#include "Function.h"

// Методи класу TBody

TBody::TBody() // У цьому конструкторі ініціалізуємо атрибути за умовчуванням

{

name = "Без імені";

arrayofsides = nullptr;

}

TBody::TBody(string name, double\* arrayofsides) // Ініціалізуємо атрибути за параметрами

{

this->name = name;

this->arrayofsides = arrayofsides;

}

double TBody::get\_area()

{

double area; // Площа поверхні фігури

// Для фігури визначаємо площу поверхні за відповідною формулою

if (name == "Прямокутний паралелепіпед")

{

area = 2 \* (arrayofsides[0] \* arrayofsides[1] + arrayofsides[1] \* arrayofsides[2] + arrayofsides[0] \* arrayofsides[2]);

}

else if (name == "Куля")

{

area = 4 \* M\_PI \* pow(arrayofsides[0], 2);

}

else if (name == "Куб")

{

area = 6 \* pow(arrayofsides[0], 2);

}

else if (name == "Циліндр")

{

area = 2 \* M\_PI \* arrayofsides[0] \* (arrayofsides[0] + arrayofsides[1]);

}

else if (name == "Конус")

{

area = M\_PI \* arrayofsides[0] \* (arrayofsides[0] + sqrt(pow(arrayofsides[0], 2) + pow(arrayofsides[1], 2)));

}

else

{

area = -1;

}

return area;

}

double TBody::get\_volume()

{

double volume; // Об'єм фігури

// Для фігури визначаємо об'єм за відповідною формулою

if (name == "Прямокутний паралелепіпед")

{

volume = arrayofsides[0] \* arrayofsides[1] \* arrayofsides[2];

}

else if (name == "Куля")

{

volume = 4. / 3 \* M\_PI \* pow(arrayofsides[0], 3);

}

else if (name == "Куб")

{

volume = pow(arrayofsides[0], 3);

}

else if (name == "Циліндр")

{

volume = M\_PI \* pow(arrayofsides[0], 2) \* arrayofsides[1];

}

else if (name == "Конус")

{

volume = 1. / 3 \* M\_PI \* pow(arrayofsides[0], 2) \* arrayofsides[1];

}

else

{

volume = -1;

}

return volume;

}

string TBody::get\_name()

{

return name;

}

// Методи класу TParallelepiped

TParallelepiped::TParallelepiped() : TBody() // У цьому конструкторі ініціалізуємо лише назву фігури та атрибути за умовчуванням (0)

{

name = "Прямокутний паралелепіпед";

a = b = c = 0;

}

TParallelepiped::TParallelepiped(double a, double b, double c) : TBody() // Ініціалізуємо атрибути за параметрами

{

name = "Прямокутний паралелепіпед";

this->a = a;

this->b = b;

this->c = c;

}

void TParallelepiped::operator = (const TParallelepiped& value) // Копіємо довжини сторін фігури

{

a = value.a;

b = value.b;

c = value.c;

}

double TParallelepiped::get\_area() // Обчислюємо площу поверхні за формулою

{

return 2 \* (a \* b + b \* c + a \* c);

}

double TParallelepiped::get\_volume() // Обчислюємо об'єм за формулою

{

return a \* b \* c;

}

// Методи класу TBall

TBall::TBall() : TBody() // У цьому конструкторі ініціалізуємо лише назву фігури та атрибут за умовчуванням (0)

{

name = "Куля";

r = 0;

}

TBall::TBall(double r) : TBody() // Ініціалізуємо атрибут за параметром

{

name = "Куля";

this->r = r;

}

void TBall::operator = (const TBall& value) // Копіюємо радіус кулі

{

r = value.r;

}

double TBall::get\_area() // Обчислюємо площу поверхні за формулою

{

return 4 \* M\_PI \* pow(r, 2);

}

double TBall::get\_volume() // Обчислюємо об'єм за формулою

{

return 4. / 3 \* M\_PI \* pow(r, 3);

}

// Функції

void input\_number(int& n, int& n\_ball) // Вводимо загальну кількість фігур та кількість куль

{

cout << "Кількість фігур = ";

cin >> n;

cout << "Кількість куль = ";

cin >> n\_ball;

cout << "Кількість прямокутних паралелепіпедів = " << n - n\_ball << endl;

}

TBall\* input\_ball(int& n\_ball) // Вводимо кулі

{

TBall\* arrayofballs = new TBall[n\_ball]; // Створюємо масив об'єктів

cout << endl;

for (int i = 0; i < n\_ball; i++)

{

double r; // Радіус кулі

cout << "Радіус " << i + 1 << "-ї кулі = ";

cin >> r; // Вводимо радіус кожної кулі

TBall temp(r); // Створюємо тимчасовий об'єкт

arrayofballs[i] = temp; // Зберігаємо радіус в елемент масиву

}

return arrayofballs;

}

TParallelepiped\* input\_parallelepiped(int n\_parallelepiped) // Вводимо прямокутні паралелепіпеди

{

TParallelepiped\* arrayofparallelepipeds = new TParallelepiped[n\_parallelepiped]; // Створюємо масив об'єктів

cout << endl;

for (int i = 0; i < n\_parallelepiped; i++)

{

double a, b, c; // Сторони прямокутного паралелепіпеда

cout << "Сторони " << i + 1 << "-го паралелепіпеда = ";

cin >> a >> b >> c; // Вводимо сторони

TParallelepiped temp(a, b, c); // Створюємо тимчасовий об'єкт

arrayofparallelepipeds[i] = temp; // Зберігаємо сторони в елемент масиву

}

return arrayofparallelepipeds;

}

void find\_area\_volume(TBall\* arrayofballs, TParallelepiped\* arrayofparallelepipeds, int n\_balls, int n\_parallelepipeds)

{

double sum\_volume = 0; // Сума об'ємів куль

double sum\_area = 0; // Сума площ поверхонь прямокутних паралелепіпедів

// Знаходимо суму об'ємів куль

for (int i = 0; i < n\_balls; i++)

{

sum\_volume += arrayofballs[i].get\_volume();

}

// Знаходимо суму площ поверхонь прямокутних паралелепіпедів

for (int i = 0; i < n\_parallelepipeds; i++)

{

sum\_area += arrayofparallelepipeds[i].get\_area();

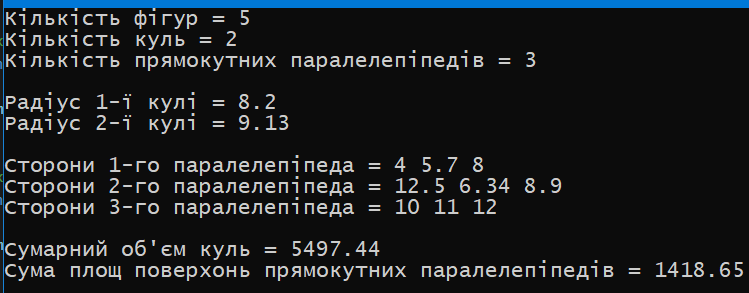
}

// Виводимо результат

cout << "\nСумарний об'єм куль = " << sum\_volume << endl;

cout << "Сума площ поверхонь прямокутних паралелепіпедів = " << sum\_area << endl;

}

****

**Код та результат програми на Python:**

**functions.py**

from classes import \*

def input\_number(): # Вводимо загальну кількість фігур та кількість куль

n = int(input("Кількість фігур = "))

n\_ball = int(input("Кількість куль = "))

print("Кількість прямокутних паралелепіпедів =", n - n\_ball, end = "\n\n")

return n, n\_ball

def input\_ball(n\_ball): # Вводимо кулі

listofballs = [] # Список куль

for i in range(n\_ball):

print("Радіус ", i + 1, "-ї кулі = ", sep = "", end = "")

r = float(input()) # Вводимо радіус

temp = TBall(r) # Створюємо об'єкт

listofballs += [temp] # Додаємо об'єкт у список

return listofballs

def input\_parallelepiped(n\_parallelepiped): # Вводимо прямокутні паралелепіпеди

listofparallelepipeds = [] # Список прямокутних параледепіпедів

print()

for i in range(n\_parallelepiped):

print("Сторони ", i + 1, "-го паралелепіпеда:", sep = "")

# Вводимо сторони прямокутного паралелепіпеда

a = float(input(" Перша сторона = "))

b = float(input(" Друга сторона = "))

c = float(input(" Третя сторона = "))

temp = TParallelepiped(a, b, c) # Створюємо об'єкт

listofparallelepipeds += [temp] # Додаємо об'єкт до списку

return listofparallelepipeds

def find\_area\_volume(listofballs, listofparallelepipeds, n\_ball, n\_parallelepiped):

sum\_volume = 0.0 # Сума об'ємів куль

sum\_area = 0.0 # Сума площ поверхонь прямокутних паралелепіпедів

for i in range(n\_ball):

sum\_volume += listofballs[i].get\_volume() # Отримуємо об'єм кулі та додаємо його до суми

for i in range(n\_parallelepiped):

sum\_area += listofparallelepipeds[i].get\_area() # Отримуємо площу поверхні прямокутного паралелепіпеда та додаємо її до суми

# Виводимо результат

print("\nСумарний об'єм куль =", sum\_volume)

print("Сума площ поверхонь прямокутних паралелепіпедів =", sum\_area, end = "\n\n")

**classes.py**

import math

class TBody:

def \_\_init\_\_(self, name, listofsides):

self.\_name = name # Ім'я фігури

self.\_\_sides = listofsides # Сторони фігури

def get\_area(self): # Для фігури знаходимо площу поверхні за окремою формулою

area = 0.0

if self.\_name == "Прямокутний паралелепіпед":

area = 2 \* (self.\_\_sides[0] \* self.\_\_sides[1] + self.\_\_sides[1] \* self.\_\_sides[2] + self.\_\_sides[0] \* self.\_\_sides[2])

elif self.\_name == "Куля":

area = 4 \* math.pi \* self.\_\_sides[0] \*\* 2

elif self.\_name == "Куб":

area = 6 \* self.\_\_sides[0] \*\* 2

elif self.\_name == "Циліндр":

area = 2 \* math.pi \* self.\_\_sides[0] \* (self.\_\_sides[0] + self.\_\_sides[1])

elif self.\_name == "Конус":

area = math.pi \* self.\_\_sides[0] \* (self.\_\_sides[0] + math.sqrt(self.\_\_sides[0] \*\* 2 + self.\_\_sides[1] \*\* 2))

else:

area = -1.0

return area

def get\_volume(self): # Для фігури знаходимо об'єм за оркемою формулою

volume = 0.0

if self.\_name == "Прямокутний паралелепіпед":

volume = self.\_\_sides[0] \* self.\_\_sides[1] \* self.\_\_sides[2]

elif self.\_name == "Куля":

volume = 4 / 3 \* math.pi \* self.\_\_sides[0] \*\* 3

elif self.\_name == "Куб":

volume = self.\_\_sides[0] \*\* 3

elif self.\_name == "Циліндр":

volume = 2 \* math.pi \* self.\_\_sides[0] \* (self.\_\_sides[0] + self.\_\_sides[1])

elif self.\_name == "Конус":

volume = math.pi \* self.\_\_sides[0] \* (self.\_\_sides[0] + math.sqrt(self.\_\_sides[0] \*\* 2 + self.\_\_sides[1] \*\* 2))

else:

volume = -1.0

return volume

def get\_name(self): # Повертаємо ім'я фігури

return self.\_name

class TBall(TBody):

def \_\_init\_\_(self, r):

self.\_name = "Куля" # Ім'я фігури ("Куля" за умовчуванням)

self.\_\_r = r # Радіус кулі

def get\_area(self): # Знаходимо площу поверхні за формулою

return 4 \* math.pi \* self.\_\_r \*\* 2

def get\_volume(self): # Знаходимо об'єм за формулою

return 4 / 3 \* math.pi \* self.\_\_r \*\* 3

class TParallelepiped(TBody):

def \_\_init\_\_(self, a, b, c):

self.\_name = "Прямокутний паралелепіпед" # Ім'я фігури ("Прямокутний паралелепіпед" за умовчуванням)

self.\_\_a = a # Перша сторона

self.\_\_b = b # Друга сторона

self.\_\_c = c # Третя сторона

def get\_area(self): # Знаходимо площу поверхні за формулою

return 2 \* (self.\_\_a \* self.\_\_b + self.\_\_b \* self.\_\_c + self.\_\_a \* self.\_\_c)

def get\_volume(self): # Знаходимо об'єм за формулою

return self.\_\_a \* self.\_\_b \* self.\_\_c

**Laba\_Python.py**

from classes import \*

from functions import \*

def main():

n, n\_ball = input\_number() # Вводимо кількість фігур

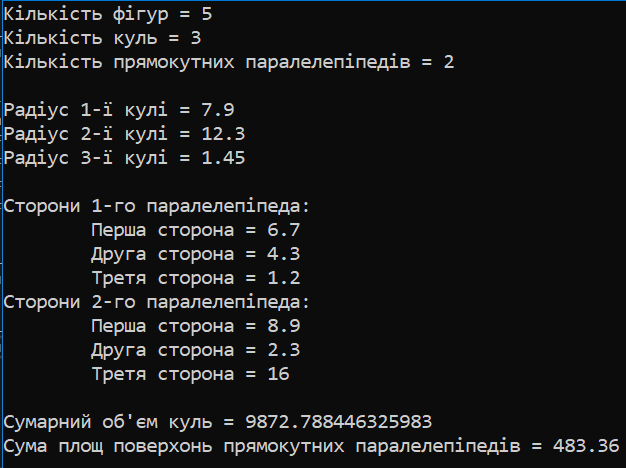
listofballs = input\_ball(n\_ball) # Вводимо радіуси кіл

listofparallelepipeds = input\_parallelepiped(n - n\_ball) # Вводимо сторони паралелепіпедів

find\_area\_volume(listofballs, listofparallelepipeds, n\_ball, n - n\_ball) # Знаходимо та виводимо суму об'ємів та площ поверхонь

if \_\_name\_\_ == "\_\_main\_\_":

main()



**Висновок –** я удосконалив свої навички у створенні та використанні класів та об'єктів та мовах C++ та Python. Опанував такі поняття, як «наслідування» та «поліморфізм». Я створив програму, яка за введеними сторонамми фігур обчислює площу поверхні та об'єм фігури та обчислює суму об'ємів куль та площ поверхонь прямокутних паралелепіпедів.