Кам’янець-Подільський національний університет імені Івана Огієнка

КАФЕДРА КОМП’ЮТЕРНИХ НАУК

Навчальна дисципліна «Об’єктно-орієнтоване програмування»

ЛАБОРАТОРНА РОБОТА #0104

Тема:

Поліморфізм

Варіант №3

**Виконала**:  
студентка 1-го курсу  
групи KNms1-B24  
Некрасова Ю.Д.

**Прийняв**:  
Слободянюк О.В.

Кам’янець-Подільський – 2025

1. **Короткі теоретичні відомості**

Поліморфізм — це принцип об'єктно-орієнтованого програмування, який дозволяє об'єктам різних типів бути обробленими через спільний інтерфейс. Це означає, що одна і та ж операція може бути виконана по-різному в залежності від типу об'єкта, на якому вона виконується.

У C# поліморфізм реалізується через два основні механізми:

1. Поліморфізм під час виконання (динамічний поліморфізм) — використовує механізм перевизначення методів (override).
2. Поліморфізм під час компіляції (статичний поліморфізм) — використовує механізм перевантаження методів (overload).

Поліморфізм під час виконання (динамічний поліморфізм) досягається через наслідування та перевизначення методів. Клас-потік може перевизначити методи свого базового класу. Коли викликається метод, виконується версія методу, що відповідає типу об'єкта, на якому він викликається, а не типу змінної, через яку цей метод викликається.

Поліморфізм під час компіляції (статичний поліморфізм) досягається через перевантаження методів (overloading), коли метод має однакове ім'я, але різні параметри (типи або кількість параметрів).

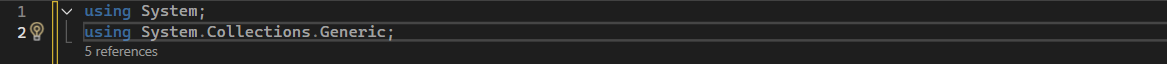
1. **Умови завдання**

Створити додаток, який задовольняє вимогам, наведеним в завданні. Успадкування застосовувати тільки в тих завданнях, в яких це логічно обґрунтоване. Аргументувати приналежність класу кожного створюваного методу і коректно перевизначити для кожного класу методи equals(), hashCode(), toString().

Створити об'єкт класу Літак, використовуючи класи Крило, Шасі, Двигун. Методи: літати, задавати маршрут, вивести на консоль маршрут.

1. **Послідовність виконання роботи**

Бібліотеки:



Клас Engine (Двигун):

* Має властивість Power, що описує потужність двигуна. Метод ToString() для виведення інформації про двигун.

Зображення, що містить текст, знімок екрана, Мультимедійне програмне забезпечення, програмне забезпечення

Вміст, створений ШІ, може бути неправильним.

Клас Wing (Крило):

* Має властивість Span, що описує розмах крила.
* Метод ToString() для виведення інформації про крило.

Зображення, що містить текст, знімок екрана, Мультимедійне програмне забезпечення, програмне забезпечення

Вміст, створений ШІ, може бути неправильним.

Клас Chassis (Шасі):

* Має властивість Material, що описує матеріал шасі.
* Метод ToString() для виведення інформації про шасі.

Зображення, що містить текст, знімок екрана, Мультимедійне програмне забезпечення, програмне забезпечення

Вміст, створений ШІ, може бути неправильним.

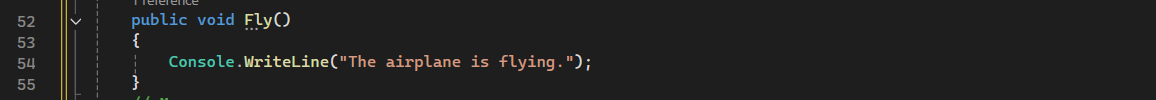
Клас Airplane (Літак):

* Має властивості для збереження об'єктів типу Engine, Wing і Chassis.

Зображення, що містить текст, знімок екрана, Мультимедійне програмне забезпечення, програмне забезпечення

Вміст, створений ШІ, може бути неправильним.

* Має методи:
  + Fly(), який виводить повідомлення про політ літака.



* + SetRoute(), який задає маршрут.



* + DisplayRoute(), який виводить на консоль маршрут.

Зображення, що містить текст, Мультимедійне програмне забезпечення, знімок екрана

Вміст, створений ШІ, може бути неправильним.

* + Метод ToString() для виведення інформації про літак.



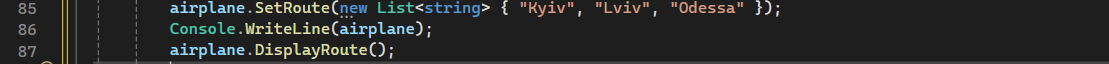
Клас Program:

* Створюється об'єкт Airplane, який ініціалізується через об'єкти класів Engine, Wing і Chassis.

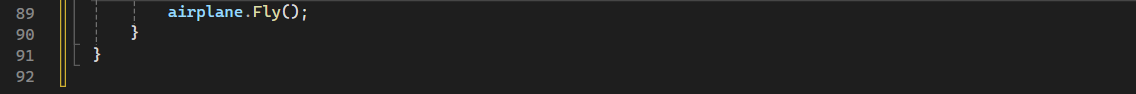
Зображення, що містить текст, знімок екрана

Вміст, створений ШІ, може бути неправильним.

* Встановлюється маршрут, і після цього виводиться інформація про літак і маршрут.



* Викликається метод Fly(), що виводить повідомлення про політ літака.



1. **Код програми**

using System;

using System.Collections.Generic;

public class Engine

{

public int Power { get; set; }

public Engine(int power)

{

Power = power;

}

public override string ToString()

{

return $"Engine Power: {Power} HP";

}

}

public class Wing

{

public double Span { get; set; }

public Wing(double span)

{

Span = span;

}

public override string ToString()

{

return $"Wing Span: {Span} meters";

}

}

public class Chassis

{

public string Material { get; set; }

public Chassis(string material)

{

Material = material;

}

public override string ToString()

{

return $"Chassis Material: {Material}";

}

}

public class Airplane

{

public Engine Engine { get; set; }

public Wing Wing { get; set; }

public Chassis Chassis { get; set; }

public List<string> Route { get; set; } = new List<string>();

public Airplane(Engine engine, Wing wing, Chassis chassis)

{

Engine = engine;

Wing = wing;

Chassis = chassis;

}

public void Fly() // Метод для польоту

{

Console.WriteLine("The airplane is flying.");

}

public void SetRoute(List<string> route) // Метод для встановлення маршруту

{

Route = route;

Console.WriteLine("Route set.");

}

public void DisplayRoute() // Метод для виведення маршруту

{

Console.WriteLine("Route:");

foreach (var point in Route)

{

Console.WriteLine(point);

}

}

public override string ToString()

{

return $"{Engine}, {Wing}, {Chassis}";

}

}

public class Program

{

public static void Main(string[] args)

{

Engine engine = new Engine(1500); // Створення об'єктів класів крило, шасі, двигун

Wing wing = new Wing(25.5);

Chassis chassis = new Chassis("Aluminum");

Airplane airplane = new Airplane(engine, wing, chassis); // Створення об'єкта літак

airplane.SetRoute(new List<string> { "Kyiv", "Lviv", "Odessa" }); // Встановлення маршруту

Console.WriteLine(airplane); // Виведення інформації про літак

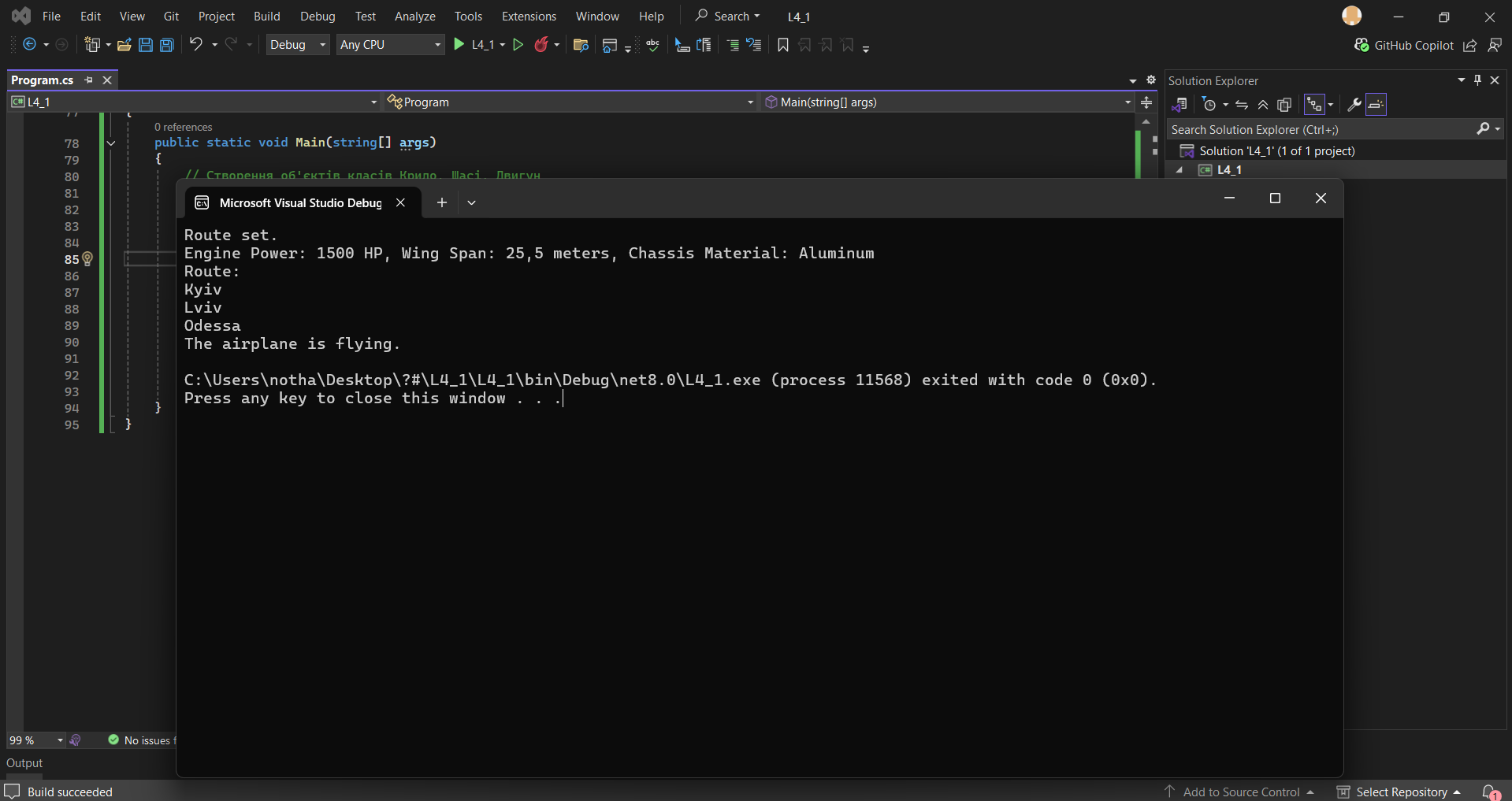
airplane.DisplayRoute(); // Виведення маршруту

airplane.Fly(); // Політ

}

}

1. **Приклад виконання програми**



1. **Умови завдання**

У кожному варіанті завдання при описі класів самостійно визначити необхідні поля, властивості та методи вводу/виводу. Деякі методи класу-предка повинні бути віртуальними і абстрактними. У програмі-клієнті для збереження сукупності об’єктів використати масив.

Створити клас TBody, який представляє просторову геометричну фігуру з методами обчислення площі поверхні та об’єму. На основі цього класу створити класи нащадки TParallelepiped та TBall. Випадковим чином створити певну кількість паралелепіпедів та куль, щоб їх сумарна кількість дорівнювала *n* . Знайти сумарну площу поверхонь усіх геометричних тіл.

1. **Послідовність виконання роботи**

Біблілтека:



Клас TBody:

* Абстрактний клас для геометричних фігур, який має два абстрактні методи:
  + SurfaceArea() — для обчислення площі поверхні.
  + Volume() — для обчислення об'єму.
* Також є метод DisplayInfo(), який виводить площу поверхні та об'єм для кожного тіла.

Зображення, що містить текст, Шрифт, знімок екрана

Вміст, створений ШІ, може бути неправильним.

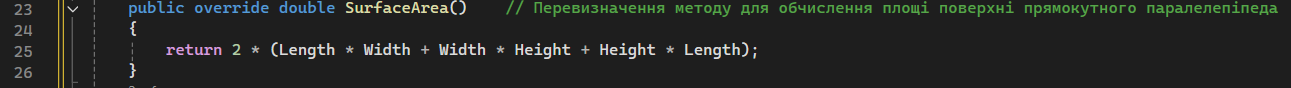
Клас TParallelepiped:

* Нащадок класу TBody, що представляє прямокутний паралелепіпед.
* Має три властивості: довжину, ширину і висоту.

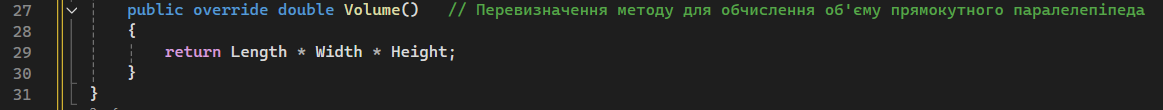
Зображення, що містить текст, знімок екрана, Мультимедійне програмне забезпечення, програмне забезпечення

Вміст, створений ШІ, може бути неправильним.

* Методи SurfaceArea() і Volume() реалізують відповідні формули для площі поверхні і об'єму паралелепіпеда:
  + Площа поверхні: 2×(L×W+W×H+H×L)



* + Об'єм: L×W×H



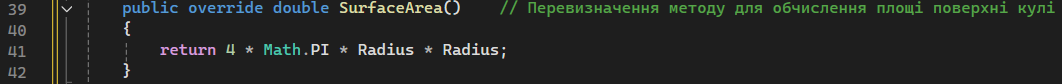
Клас TBall:

* Нащадок класу TBody, що представляє кулю.
* Має властивість радіус.

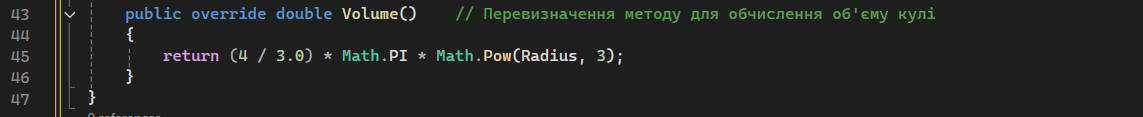
Зображення, що містить Мультимедійне програмне забезпечення, текст, Графічний редактор, знімок екрана

Вміст, створений ШІ, може бути неправильним.

* Методи SurfaceArea() і Volume() реалізують відповідні формули для площі поверхні і об'єму кулі:
  + Площа поверхні: 4×π×r2



* + Об'єм: ×π×r3



Клас Program:

* У програмі створюється масив об'єктів TBody, який містить випадково створені об'єкти класів TParallelepiped і TBall.
* Кількість тіл задається змінною n (в даному випадку 10).
* Випадковим чином створюються або паралелепіпеди, або кулі з випадковими розмірами (довжина, ширина, висота для паралелепіпедів і радіус для куль).

Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Вміст, створений ШІ, може бути неправильним.

* Потім обчислюється і виводиться сумарна площа поверхні всіх тіл, а також виводяться деталі кожного тіла.

Зображення, що містить текст, Мультимедійне програмне забезпечення, програмне забезпечення, Графічний редактор

Вміст, створений ШІ, може бути неправильним.

1. **Код програми**

using System;

public abstract class TBody

{

public abstract double SurfaceArea(); // Абстрактні методи для обчислення площі поверхні та об'єму

public abstract double Volume();

public void DisplayInfo() // Метод для виведення даних про геометричну фігуру

{

Console.WriteLine($"Surface Area: {SurfaceArea():F2}, Volume: {Volume():F2}");

}

}

public class TParallelepiped : TBody

{

public double Length { get; set; }

public double Width { get; set; }

public double Height { get; set; }

public TParallelepiped(double length, double width, double height)

{

Length = length;

Width = width;

Height = height;

}

public override double SurfaceArea() // Перевизначення методу для обчислення площі поверхні прямокутного паралелепіпеда

{

return 2 \* (Length \* Width + Width \* Height + Height \* Length);

}

public override double Volume() // Перевизначення методу для обчислення об'єму прямокутного паралелепіпеда

{

return Length \* Width \* Height;

}

}

public class TBall : TBody

{

public double Radius { get; set; }

public TBall(double radius)

{

Radius = radius;

}

public override double SurfaceArea() // Перевизначення методу для обчислення площі поверхні кулі

{

return 4 \* Math.PI \* Radius \* Radius;

}

public override double Volume() // Перевизначення методу для обчислення об'єму кулі

{

return (4 / 3.0) \* Math.PI \* Math.Pow(Radius, 3);

}

}

public class Program

{

public static void Main(string[] args)

{

Random random = new Random();

int n = 10; // Загальна кількість тіл

TBody[] bodies = new TBody[n]; // Масив для збереження геометричних тіл

for (int i = 0; i < n; i++) // Створення випадкових паралелепіпедів та куль

{

if (random.Next(2) == 0)

{ // Створення випадкового паралелепіпеда

double length = random.Next(1, 10);

double width = random.Next(1, 10);

double height = random.Next(1, 10);

bodies[i] = new TParallelepiped(length, width, height);

}

else

{

// Створення випадкової кулі

double radius = random.Next(1, 10);

bodies[i] = new TBall(radius);

}

}

double totalSurfaceArea = 0; // Обчислення сумарної площі поверхні всіх геометричних фігур

foreach (var body in bodies)

{

totalSurfaceArea += body.SurfaceArea();

}

Console.WriteLine($"Total Surface Area of all bodies: {totalSurfaceArea:F2}"); // Виведення результату

for (int i = 0; i < n; i++) // Виведення даних для кожного тіла

{

Console.WriteLine($"Body {i + 1}:");

bodies[i].DisplayInfo();

}

}

}

1. **Приклад виконання програми**

Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Вміст, створений ШІ, може бути неправильним.