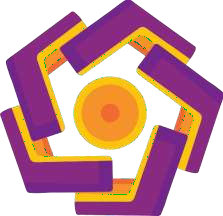
**Penghitung Luas Bangun Datar**

**Ujian Akhir Semester Mata Kuliah Pemrograman**

**Dosen Pengampu : Adjie Kusuma Wardhana M.Kom, M.Eng**



**Disusun oleh :**

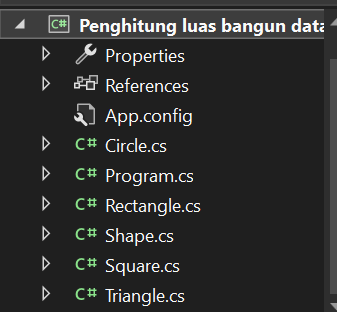
Kelas : 23IF01

Nama : Rayyis Hammam Mukhayyar

NIM : 23.11.5405

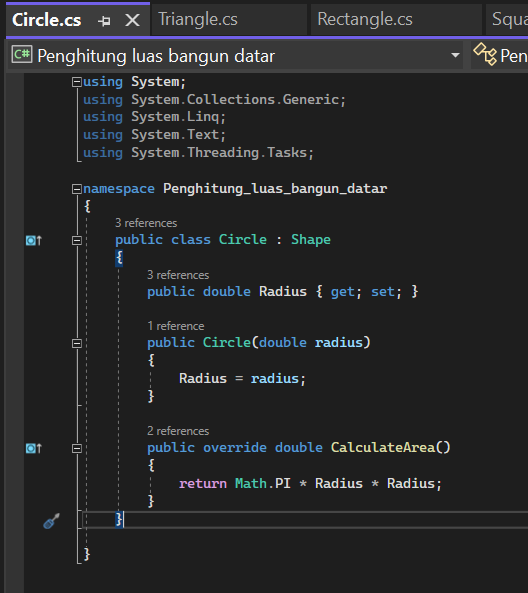
**PRODI INFORMATIKA FAKULTAS ILMU KOMPUTER UNIVERSITAS AMIKOM 2024/2025**

**Structur Project**

****

**Penjelasan Program dan Code**

**Circle.cs**

****

**using System ; :** Mengimpor Namespace ‘system’, yang berisi fundamental classes dan base classes yang mendefinisikan commonly-used data types, events, dan event handlers, interfaces, attributes, dan processing exceptions.

**using System.Collection.Generic; :** Mengimpor namespace ‘System.Collection.Generic’, yang menyediakan interfaces dan classes yang mendefiniskan generic collecton, seperti ‘List<T>’, ‘Dictionary<Tkey, TValue >, dll.

**using System.Linq; :** Mengimpor namespace ‘System.Linq’, yang menyediakan classes dan interfaces yang mendukung LINQ (Language Integrated Query).

**using System.Text; :** Mengimpor namespace ‘System.Text’, yang berisi classes yang mewakili ASCII, Unicode, dan karakter encoding lain, dan classes untuk mengkonversi antar encoding karkter.

**using system.Threading.Tasks; :** Mengimpor namespace ‘System.Threading.Tasks’, yang menyediakan jenis untuk menulis parallel code dan asynchronus code.

**namespace Penghitung\_luas\_bangun\_datar :** Mendefinisikan sebuah namespace bernama ‘Penghitung\_luas\_bangun\_datar’ untuk mengelompokkan class-class terkait, dan untuk menghindarikonflik nama dengan class lain yang memiliki nama sama di namespace berbeda.

**Public class Circle : Shape :** Mendeklarasikan sebuah public class bernama ‘Circle’ yang mewarisi (inherit) dari class ‘Shape’. Ini berarti ‘Circle’ adalah subclass dari ‘Shape’.

**public double Radius{ get; set; } :** Mendeklarasikan property public ‘Radius’ bertipe ‘double’ dengan getter dan setter otomatis. Property ini di gunkakan untuk menyimpan radius / jari jari dari linkaran.

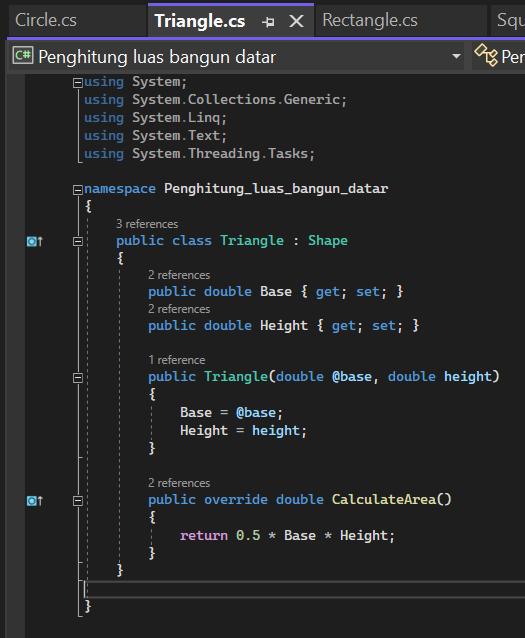
**public Circle(double radius) :** Mendeklarasikan sebuah constructor untuk class ‘circle’ yang menerima satu parameter ‘radius’ bertipe ‘double’.

**Radius = radius; :** Menginisialisasi property ‘Radius’ dengan nilai parameter ‘radius’ yang diterima oleh constructor.

**public override double CalculateArea( ) :** Mendeklarasikan method public ‘CalculateArea’ yang mengoverride method dengan nama sama di superclass ‘Shape’. Method ini bertipe ‘double’.

**return Math.PI \* Radius; :** Menghitung dan mengembalikan luas lingkaran menggunkan formula πx Radius^2. ‘Math.PI’ adalah konstanta dari namespace ‘System’ yang merepresantikan nilai pi (π).

**Triangle.cs**



**public class Triangle : Shape :** Mendeklarasikan sebuah public class bernama ‘Triangle’ yang mewarisi (inherit) dari class ‘Shape’. Ini berarti ‘Triangle’ adalah subclass dari ‘Shape’.

**public double Base { get; set; } :** Mendeklarasikan property public ‘Base’ bertipe ‘double’ dengan getter dan setter otomatis. Property ini digunakan untuk menyimpan panjang alas dari segitiga.

**public double Height { get; set; } :** Mendeklarasikan property public ‘Height’ bertipe ‘double’ dengan getter dan setter otomatis. Property ini di gunakan untuk menyimpan tinggi dari segitiga.

**public Triangle(double @base, double height) :** Mendeklarasikan sebuah constructor untuk class ‘Triangle’ yang menerima dua parameter, ‘@base’ dan ‘height’, keduanya bertipe ‘double’.

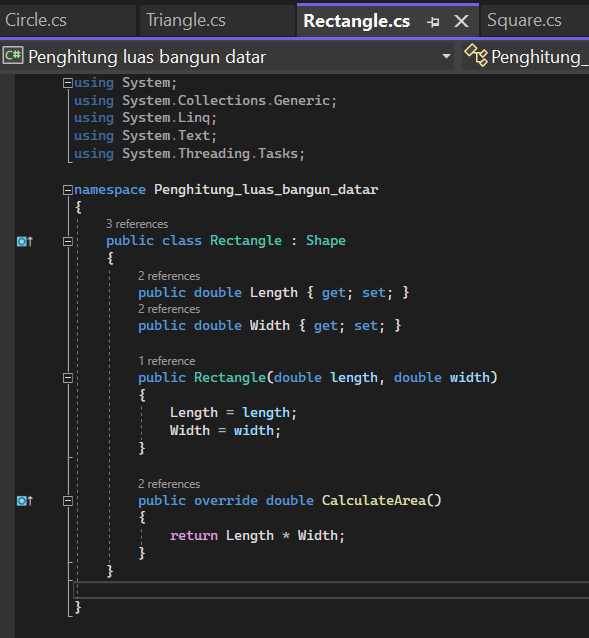
**Base = @base; :** Menginisialisasi property ‘Base’ dengan nilai parameter ‘@base’ yang diterima oleh contructor.

**Height = height; :** Menginisialisasi property ‘Height’ dengan nilai parameter ‘height’ yang diterima oleh constructor.

**public override double CalculateArea( ) :** Mendeklarasikan method public ‘CalculateArea’ yang mengoverride method dengan nama sama di superclass ‘Shape’. Method ini bertipe ‘double’.

**return 0.5 \* Base \* Height; :** Menghitung dan mengembalikan luas segitiga menggunakan formula ½ x Base x Height.

**Rectangle.cs**

****

**public class Rectangle : Shape :** Mendeklarasikan sebuah public class bernama ‘Rectangle’ yang mewarisi (inherit) dari class ‘Shape’. Ini berarti ‘Rectangle’ adalah subclass dari ‘Shape’.

**public double Length { get; set; } :** Mendeklarasikan property public ‘Length’ bertipe ‘double’ dengan getter dan setter otomatis. Property ini digunakan untuk menyimpan panjang dari persegi panjang.

**public double Width { get; set; } :** Mendeklarasikan property public ‘Width’ bertipe ‘double’ deangan getter dan setter otomatis. Property ini digunakan untuk menyimpan lebar dari persegi panjang.

**public Rectangle(double length, double width) :** Mendeklarasikan sebuah constructor untuk class ‘Rectangle’ yang menerima dua parameter, ‘length’ dan ‘width’, keduanya bertipe ‘double’.

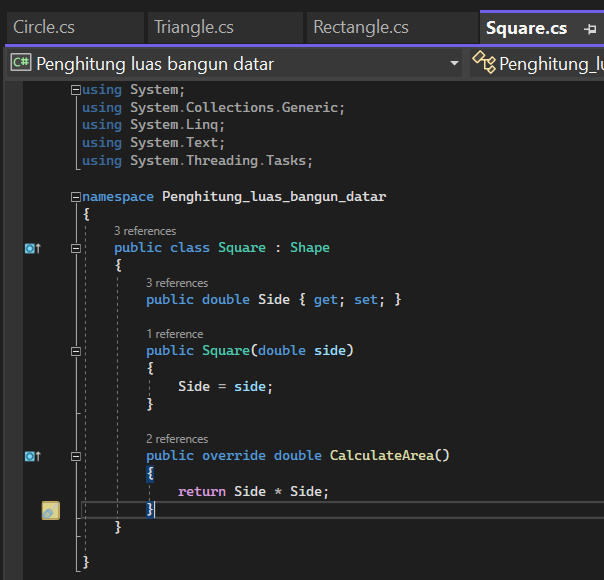
**Length = length; :** Menginisialisasi property ‘Length’ dengan nilai parameter ‘length’ yang diterima oleh contructor.

**Width = width; :** Menginisialisasi property ‘With’ dengan nilai parameter ‘width’ yang diterima oleh contructor.

**public override double CalculateArea( ) :** Mendeklarasikan method public ‘CalculateArea’ yang mengoverride method dengan nama sama di superclass ‘Shape’. Method ini bertipe ‘double’.

**return Length \* Width; :** Menghitung dan mengembalikan luas persegi panjang menggunaka formula Legth x Width.

**Square.cs**

****

**public class Square : Shape :** Mendeklarasikan sebuah public class bernama ‘Square’ yang mewarisi (inherit) dari class ‘Shape’. Ini berarti ‘Square’ adalah subclass dari ‘Shape’.

**public double side { get; set; } :** Mendeklarasikan property public ‘Side’ bertipe ‘double’ dengan getter dan setter otomatis. Property ini digunakan untuk menyimpan panjang sisi dari persegi.

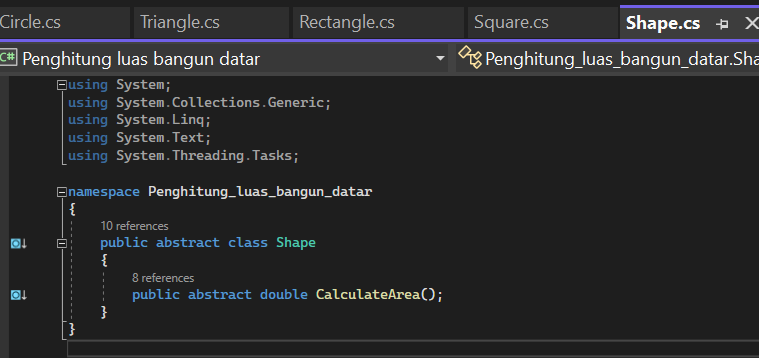
**public Square(double side) :** Mendeklarasikan sebuah contructor untuk class ‘Square’ yang menerima satu parameter ‘side’ bertipe ‘double’.

**Side = side; :** Menginisialisasi property ‘Side’ dengan nilai parameter ‘side’ yang diterima oleh constructor.

**public override double CalculateArea( ) :** Mendeklarasikan method public ‘CalculateArea’ yang mengoverride method dengan nama sama di superclass ‘Shape’. Method ini bertipe ‘double’.

**retrun Side \* Side; :** Menghitung dan mengembalikkan luas persegi menggunakan formula Side x Side atau Side^2.

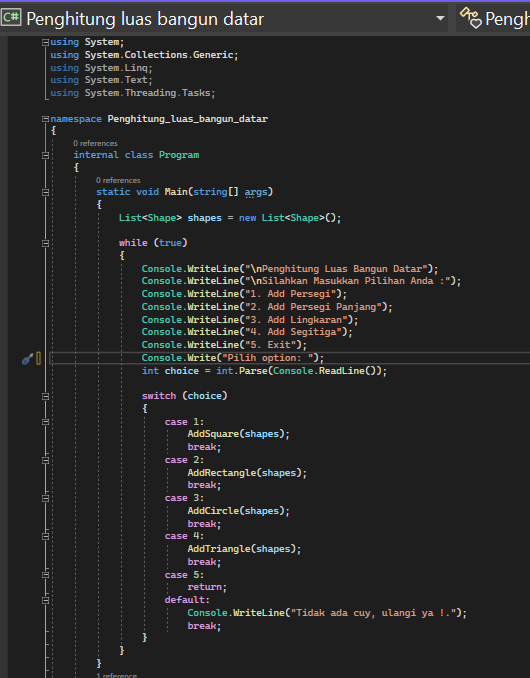
**Shape.cs**

****

**public abstract class Shape :** Mendeklarasikan sebuah abstract class bernama ‘Shape’.

**public abstract douoble CalculateArea( ); :** Mendeklarasikan sebuah method abstract ‘CalculateArea’ yang bertipe ‘double’. Method ini tidak memiliki implementasi di dalam ‘Shape’ dan harus diimplementasikan oleh semua class yang mewarisi ‘Shape’.

**Program.cs**

****

**namespace Penghitung\_luas\_bangun\_datar :** Mendefinisikan sebuah namespace bernama `Penghitung\_luas\_bangun\_datar` untuk mengelompokkan class-class terkait, dan untuk menghindari konflik nama dengan class lain yang memiliki nama sama di namespace berbeda.

**internal class Program :** Mendeklarasikan sebuah internal class bernama `Program`. Class ini hanya dapat diakses dalam assembly yang sama.

**static void Main(string[] args) :** Mendefinisikan method `Main`, yang merupakan entry point dari aplikasi. Method ini dieksekusi pertama kali saat program dijalankan.

**List<Shape> shapes = new List<Shape>(); :** Membuat sebuah list untuk menyimpan objek-objek dari type `Shape`.

**while (true):** Memulai sebuah loop yang berjalan terus-menerus hingga secara eksplisit dihentikan.

**Console.WriteLine(...) dan Console.Write("Pilih Option: ") :** Menampilkan teks ke console untuk memberikan petunjuk kepada pengguna.

**int choice = int.Parse(Console.ReadLine()); :** Membaca input dari pengguna, mengubahnya menjadi integer, dan menyimpannya dalam variabel `choice`.

**switch (choice) :** Memilih tindakan berdasarkan nilai `choice`.

**case 1, case 2, case 3, case 4, case 5 :** Memanggil method yang sesuai atau menghentikan loop.

**return :** Menghentikan eksekusi method `Main`, yang berarti keluar dari program.

**default :** Menangani input yang tidak valid dengan menampilkan pesan kesalahan.

⚫ **private static void AddSquare(List<Shape> shapes) :** Method untuk menambahkan objek `Square` ke dalam list `shapes`.

**Console.WriteLine(...) dan Console.Write(...) :** Menampilkan teks ke console untuk meminta input dari pengguna.

**double side = double.Parse(Console.ReadLine()); :** Membaca input dari pengguna dan mengubahnya menjadi `double`.

**Square square = new Square(side); :** Membuat objek `Square` dengan nilai `side` yang diinput oleh pengguna.

**shapes.Add(square); :** Menambahkan objek `Square` ke dalam list `shapes`.

**Console.WriteLine(...) :** Menampilkan luas persegi ke console.

**⚫ private static void AddRectangle(List<Shape> shapes) :** Method untuk menambahkan objek `Rectangle` ke dalam list `shapes`.

**Console.WriteLine(...) dan Console.Write(...) :** Menampilkan teks ke console untuk meminta input dari pengguna.

**double length = double.Parse(Console.ReadLine()); :** Membaca input panjang dari pengguna dan mengubahnya menjadi `double`.

**double width = double.Parse(Console.ReadLine()); :** Membaca input lebar dari pengguna dan mengubahnya menjadi `double`.

**Rectangle rectangle = new Rectangle(length, width); :** Membuat objek `Rectangle` dengan nilai `length` dan `width` yang diinput oleh pengguna.

**shapes.Add(rectangle); :** Menambahkan objek `Rectangle` ke dalam list `shapes`.

**Console.WriteLine($"Luas persegi panjang : {rectangle.CalculateArea()}"); :** Menampilkan luas persegi panjang ke console.

**⚫ private static void AddCircle(List<Shape> shapes) :** Method untuk menambahkan objek `Circle` ke dalam list `shapes`.

**Console.WriteLine(...) dan Console.Write(...) :** Menampilkan teks ke console untuk meminta input dari pengguna.

**double radius = double.Parse(Console.ReadLine()); :** Membaca input jari-jari dari pengguna dan mengubahnya menjadi `double`.

**Circle circle = new Circle(radius); :** Membuat objek `Circle` dengan nilai `radius` yang diinput oleh pengguna.

**shapes.Add(circle); :** Menambahkan objek `Circle` ke dalam list `shapes`.

**Console.WriteLine($"Luas lingkaran : {circle.CalculateArea()}") :** Menampilkan luas lingkaran ke console.

**⚫ private static void AddTriangle(List<Shape> shapes) :** Method untuk menambahkan objek `Triangle` ke dalam list `shapes`.

**Console.WriteLine(...) dan Console.Write(...) :** Menampilkan teks ke console untuk meminta input dari pengguna.

**double @base = double.Parse(Console.ReadLine()); :** Membaca input panjang alas dari pengguna dan mengubahnya menjadi `double`.

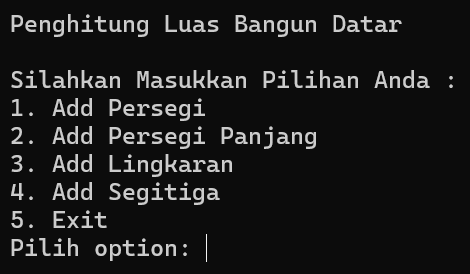
**double height = double.Parse(Console.ReadLine()); :** Membaca input tinggi dari pengguna dan mengubahnya menjadi `double`.

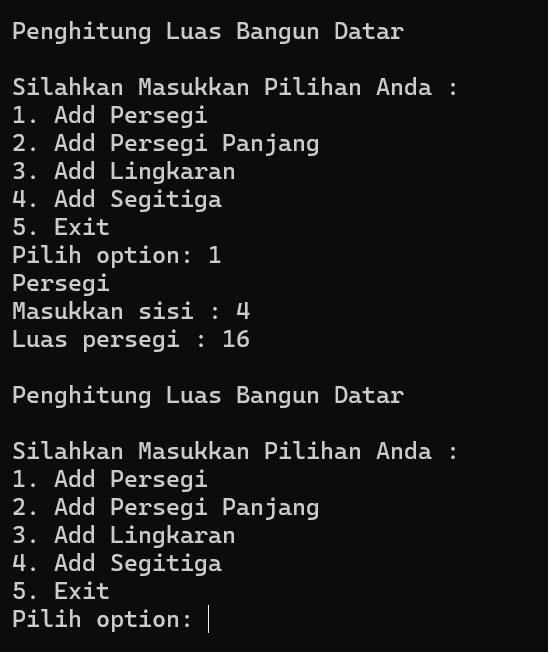
**Triangle triangle = new Triangle(@base, height);` :** Membuat objek `Triangle` dengan nilai `@base` dan `height` yang diinput oleh pengguna.

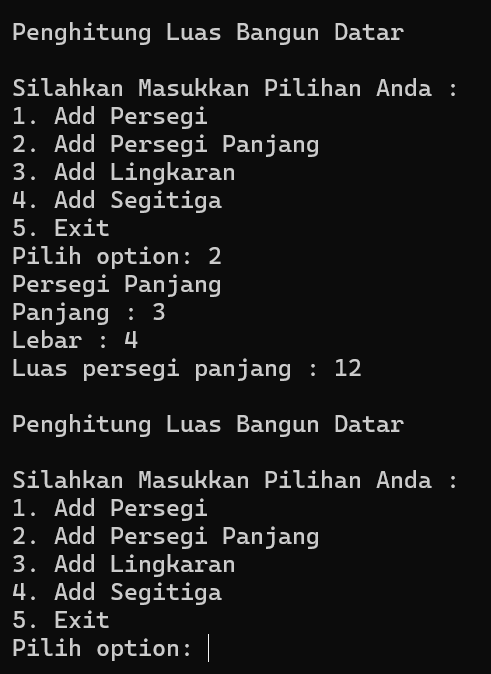
**shapes.Add(triangle); :** Menambahkan objek `Triangle` ke dalam list `shapes`.

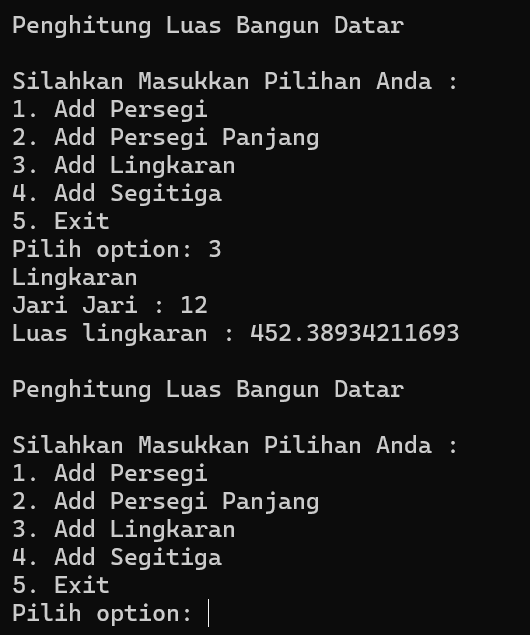
**Console.WriteLine($"Luas segitiga : {triangle.CalculateArea( )}” :** Menampilkan luas segitiga ke console.

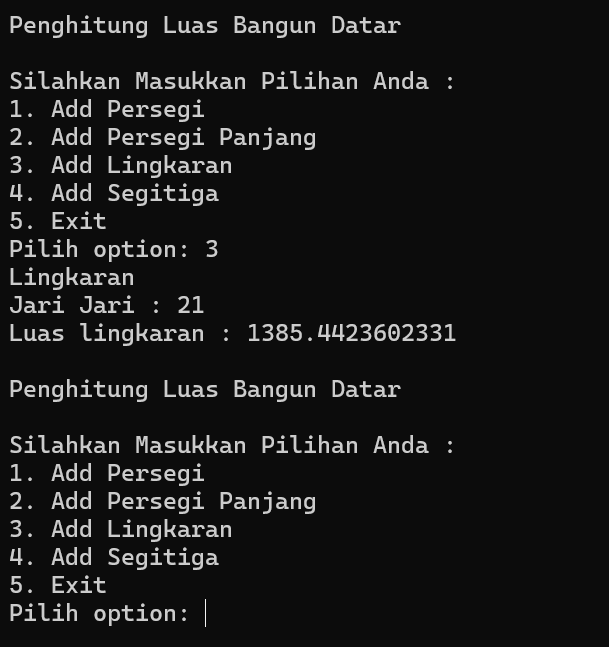
**OUTPUT**

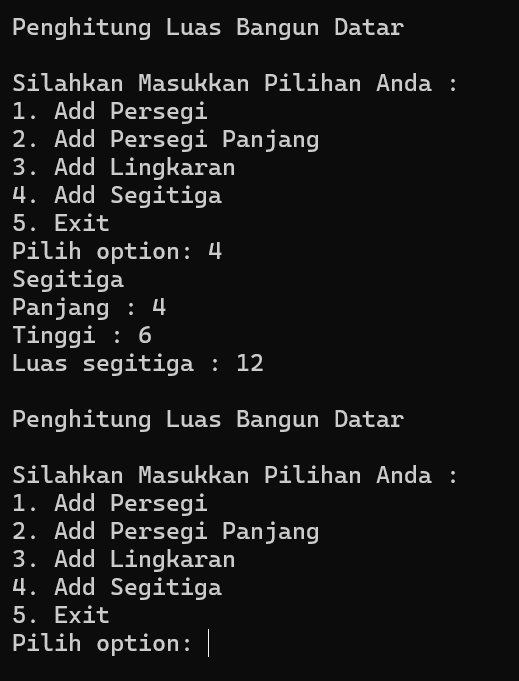
****

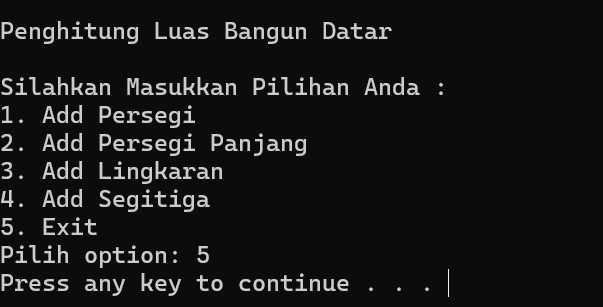
****

****

****

****

****

****

Penjelasan Program :

1. Encapsulation: Setiap kelas (`Shape`, `Square`, `Rectangle`, `Circle`, `Triangle`) memiliki properti dan metode yang relevan untuk menghitung luas. Properti-properti ini dienkapsulasi dalam kelas masing-masing.

2. Abstraction: Kelas abstrak `Shape` mendefinisikan metode `CalculateArea` yang harus diimplementasikan oleh kelas turunan. Ini menyediakan kerangka kerja umum untuk semua bentuk geometri.

3. Inheritance: Kelas `Square`, `Rectangle`, `Circle`, dan `Triangle` mewarisi dari kelas abstrak `Shape` dan mengimplementasikan metode `CalculateArea`.

4. Polymorphism: Dalam metode `AddSquare`, `AddRectangle`, `AddCircle`, dan `AddTriangle`, kita menggunakan polimorfisme untuk memanggil metode `CalculateArea` pada objek `Shape` tanpa harus mengetahui tipe spesifiknya.