

WORKSHOP APLIKASI MOBILE FRAMEWORK

Acara 9 - 10



Dosen Pengampu:

Mochammad Rifki Ulil Albaab, ST., M.Tr.T.

Disusun Oleh:

Nama : Mochammad Ferdian Saputro
NIM : E41231827

MATA KULIAH

WORKSHOP APLIKASI MOBILE FRAMEWORK

POLITEKNIK NEGERI JEMBER

2023/2024

DAFTAR ISI

DAFTAR ISI	2
BAB I PENDAHULUAN	3
1.1 Pengertian GitHub	Error! Bookmark not defined.
BAB II ALAT DAN BAHAN	4
2.1 Alat dan Bahan	4
BAB III PEMBAHASAN	5
3.1 ACARA 9	5
3.1.1 Membuat Fuction	5
3.2 ACARA 10	5
3.2.1 Enkapsulasi	5
3.2.2 Inheritance	6
3.2.3 Polymorps	7
3.2.4 Constructors	9

BAB I

PENDAHULUAN

1.1 Function

Function adalah blok kode yang dirancang untuk menjalankan suatu tindakan tertentu dan dapat digunakan kembali dalam program. Function membantu meningkatkan efisiensi dan keteraturan kode dengan menghindari duplikasi.

Struktur Function

1. **Nama Function** → Digunakan untuk memanggil function.
2. **Parameter** → Nilai yang dikirimkan ke function untuk diproses.
3. **Isi Function** → Tindakan atau perintah yang akan dijalankan.
4. **Return (Opsional)** → Mengembalikan nilai sebagai hasil dari function.

Function dapat menerima lebih dari satu parameter dan dipisahkan dengan tanda koma. Jika function mengembalikan nilai, sintaks return digunakan sebelum function berakhir.

BAB II

ALAT DAN BAHAN

2.1 Alat dan Bahan

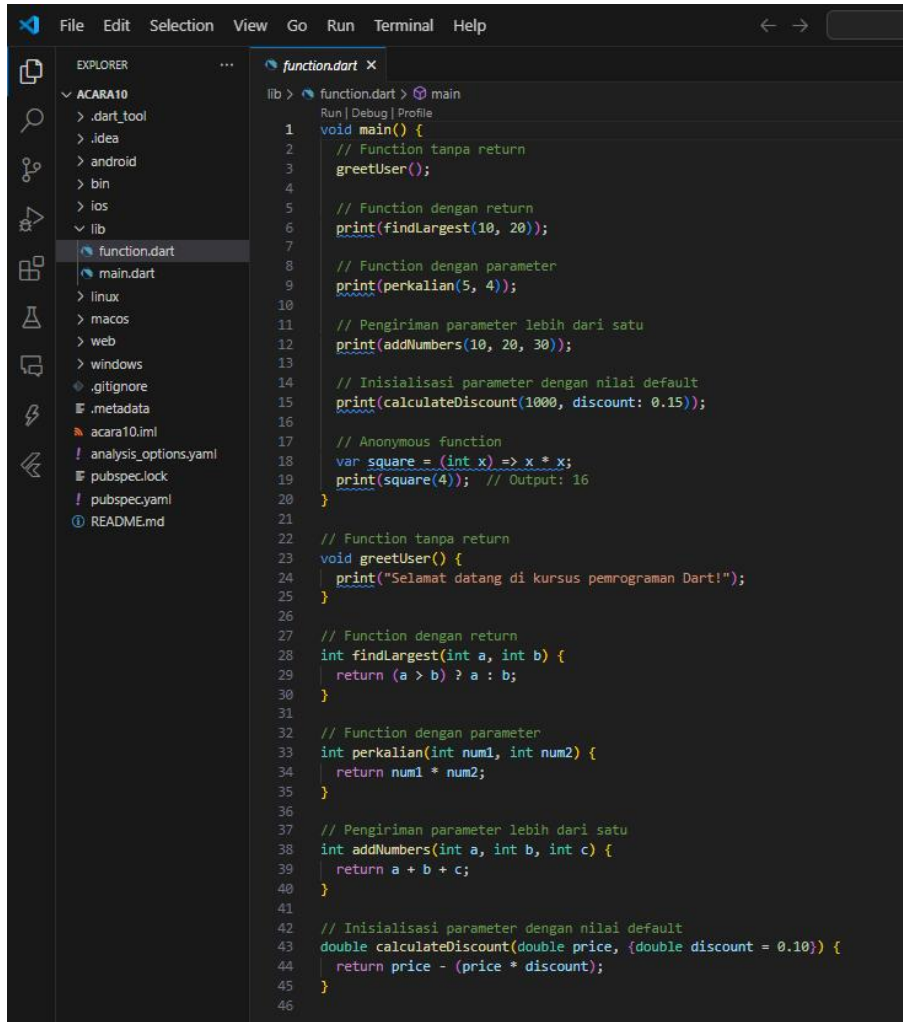
- Kertas HVS
- Spidol Besar
- Bolpoin
- Laptop

BAB III PEMBAHASAN

3.1 ACARA 9

3.1.1 Membuat Fuction

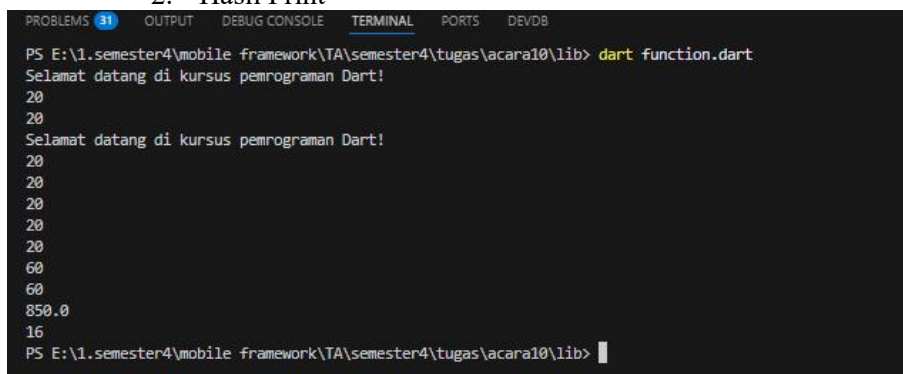
1. Code



The screenshot shows an IDE with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'ACARA10' with various files and folders. The code editor displays the content of 'function.dart'. The code defines a main function and several utility functions: greetUser, findLargest, perkalian, addNumbers, and calculateDiscount. It also includes an anonymous function for calculating a square.

```
lib > function.dart > main
Run | Debug | Profile
1 void main() {
2   // Function tanpa return
3   greetUser();
4
5   // Function dengan return
6   print(findLargest(10, 20));
7
8   // Function dengan parameter
9   print(perkalian(5, 4));
10
11  // Pengiriman parameter lebih dari satu
12  print(addNumbers(10, 20, 30));
13
14  // Inisialisasi parameter dengan nilai default
15  print(calculateDiscount(1000, discount: 0.15));
16
17  // Anonymous function
18  var square = (int x) => x * x;
19  print(square(4)); // Output: 16
20 }
21
22 // Function tanpa return
23 void greetUser() {
24   print("Selamat datang di kursus pemrograman Dart!");
25 }
26
27 // Function dengan return
28 int findLargest(int a, int b) {
29   return (a > b) ? a : b;
30 }
31
32 // Function dengan parameter
33 int perkalian(int num1, int num2) {
34   return num1 * num2;
35 }
36
37 // Pengiriman parameter lebih dari satu
38 int addNumbers(int a, int b, int c) {
39   return a + b + c;
40 }
41
42 // Inisialisasi parameter dengan nilai default
43 double calculateDiscount(double price, {double discount = 0.10}) {
44   return price - (price * discount);
45 }
46
```

2. Hasil Print



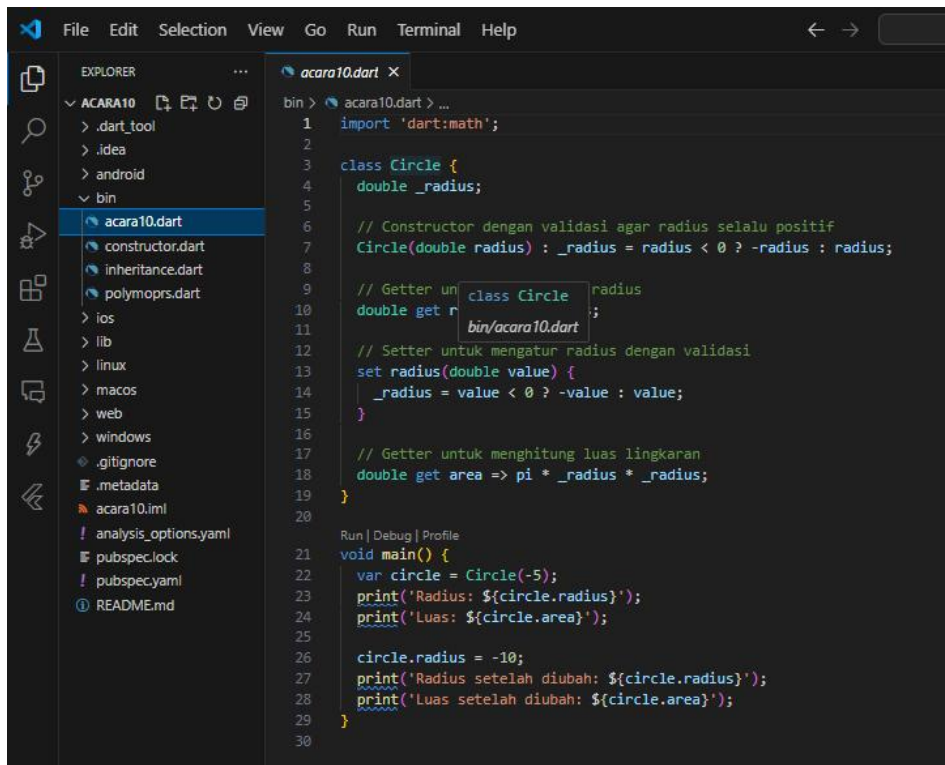
The screenshot shows a terminal window with the command 'dart function.dart' executed. The output displays the results of the program, including the greeting message and the numerical outputs of the various functions.

```
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\lib> dart function.dart
Selamat datang di kursus pemrograman Dart!
20
20
Selamat datang di kursus pemrograman Dart!
20
20
20
20
20
60
60
850.0
16
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\lib>
```

3.2 ACARA 10

3.2.1 Enkapsulasi

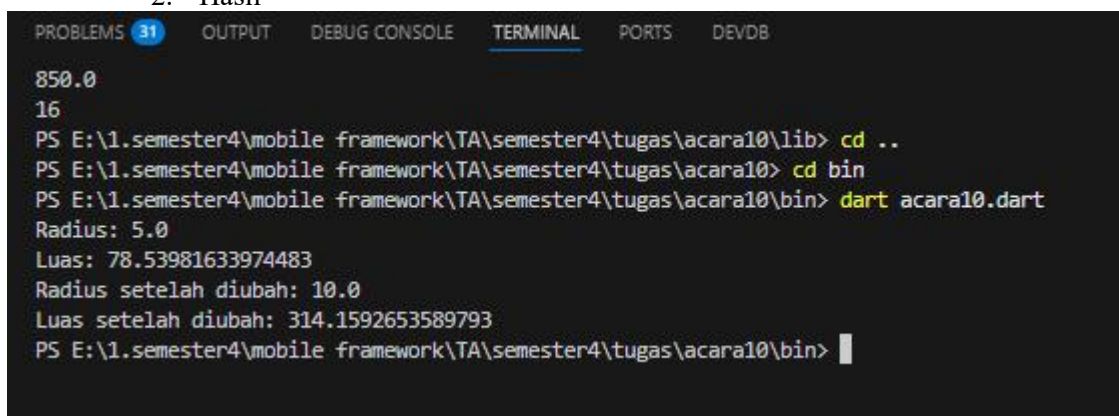
1. Code



```
1 import 'dart:math';
2
3 class Circle {
4   double _radius;
5
6   // Constructor dengan validasi agar radius selalu positif
7   Circle(double radius) : _radius = radius < 0 ? -radius : radius;
8
9   // Getter untuk radius
10  double get radius => _radius;
11
12  // Setter untuk mengatur radius dengan validasi
13  set radius(double value) {
14    _radius = value < 0 ? -value : value;
15  }
16
17  // Getter untuk menghitung luas lingkaran
18  double get area => pi * _radius * _radius;
19 }
20
21 void main() {
22   var circle = Circle(-5);
23   print('Radius: ${circle.radius}');
24   print('Luas: ${circle.area}');
25
26   circle.radius = -10;
27   print('Radius setelah diubah: ${circle.radius}');
28   print('Luas setelah diubah: ${circle.area}');
29 }
30
```

Penggunaan `_radius` yang dimana tanda (`_`) bersifat private didalam dart sehingga tidak dapat diakses dari luar class

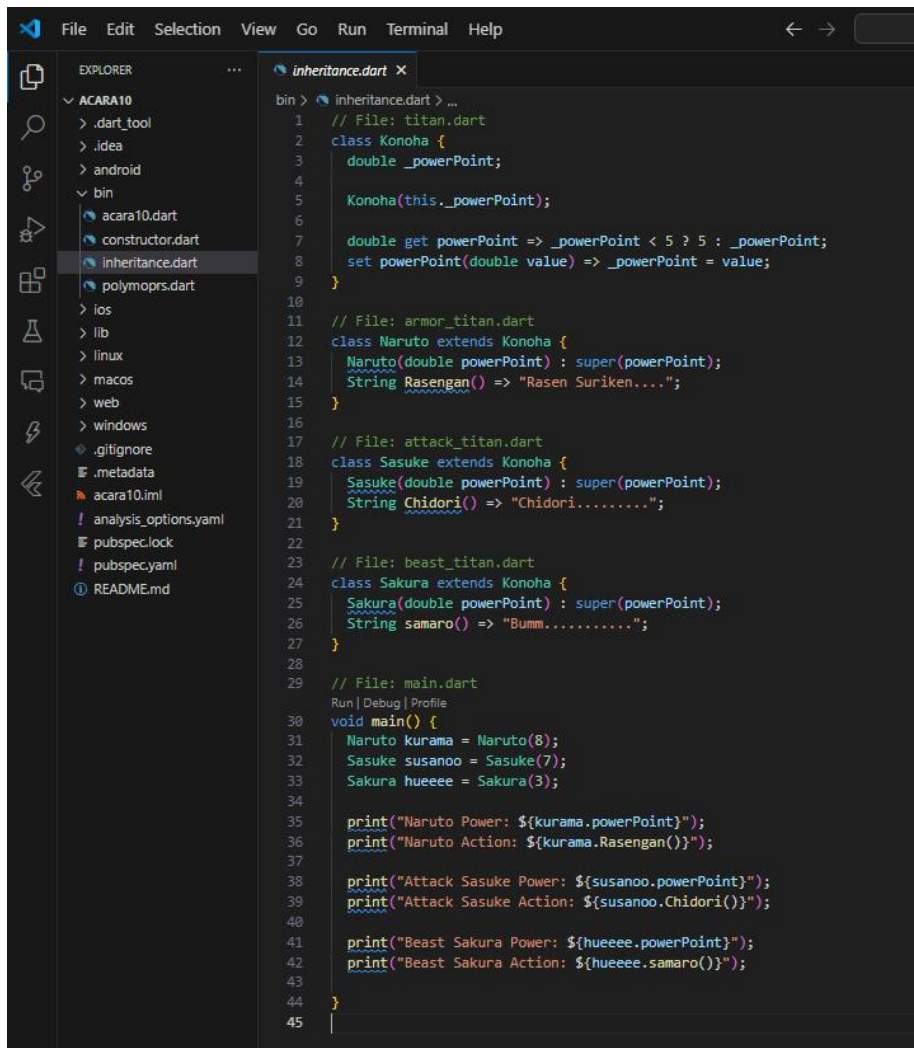
2. Hasil



```
850.0
16
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\lib> cd ..
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10> cd bin
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin> dart acara10.dart
Radius: 5.0
Luas: 78.53981633974483
Radius setelah diubah: 10.0
Luas setelah diubah: 314.1592653589793
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin>
```

3.2.2 Inheritance

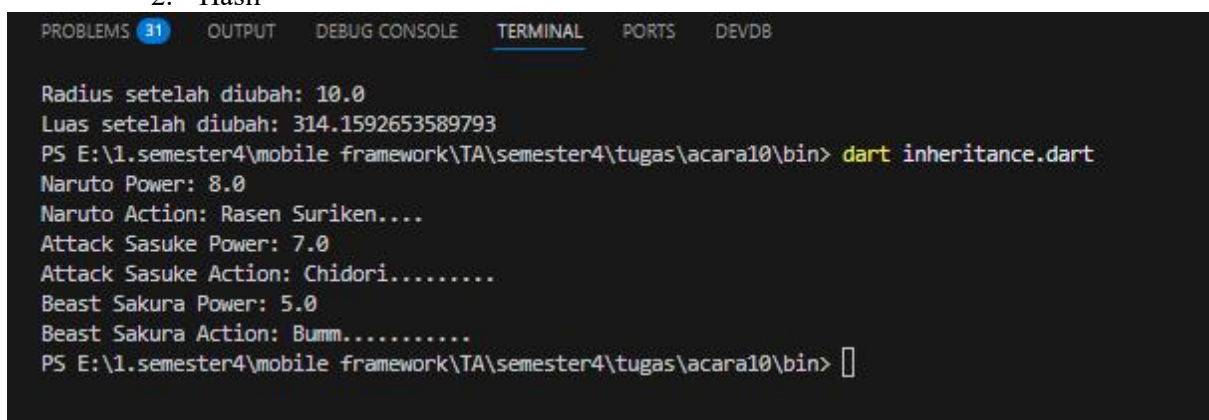
1. Code



```
1 // File: titan.dart
2 class Konoha {
3   double _powerPoint;
4
5   Konoha(this._powerPoint);
6
7   double get powerPoint => _powerPoint < 5 ? 5 : _powerPoint;
8   set powerPoint(double value) => _powerPoint = value;
9 }
10
11 // File: armor_titan.dart
12 class Naruto extends Konoha {
13   Naruto(double powerPoint) : super(powerPoint);
14   String Rasengan() => "Rasen Suriken...";
15 }
16
17 // File: attack_titan.dart
18 class Sasuke extends Konoha {
19   Sasuke(double powerPoint) : super(powerPoint);
20   String Chidori() => "Chidori.....";
21 }
22
23 // File: beast_titan.dart
24 class Sakura extends Konoha {
25   Sakura(double powerPoint) : super(powerPoint);
26   String samaro() => "Bumm.....";
27 }
28
29 // File: main.dart
30 void main() {
31   Naruto kurama = Naruto(8);
32   Sasuke susanoo = Sasuke(7);
33   Sakura hueeee = Sakura(3);
34
35   print("Naruto Power: ${kurama.powerPoint}");
36   print("Naruto Action: ${kurama.Rasengan()}");
37
38   print("Attack Sasuke Power: ${susanoo.powerPoint}");
39   print("Attack Sasuke Action: ${susanoo.Chidori()}");
40
41   print("Beast Sakura Power: ${hueeee.powerPoint}");
42   print("Beast Sakura Action: ${hueeee.samaro()}");
43 }
44
45
```

Penggunaan Extends sendiri disini sudah merupakan implementasi dari inheritance dimana satu class bisa digunakan di class lainnya

2. Hasil



```
PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS DEVDB

Radius setelah diubah: 10.0
Luas setelah diubah: 314.1592653589793
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin> dart inheritance.dart
Naruto Power: 8.0
Naruto Action: Rasen Suriken...
Attack Sasuke Power: 7.0
Attack Sasuke Action: Chidori.....
Beast Sakura Power: 5.0
Beast Sakura Action: Bumm.....
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin>
```

3.2.3 Polymorps

1. Code


```
bin > polymoprs.dart > main
1 // File: bangun_datar.dart
2 abstract class BangunDatar {
3     double luas();
4     double keliling();
5 }
6
7 // File: jajargenjang.dart
8 class Jajargenjang extends BangunDatar {
9     double alas, tinggi, sisiMiring;
10
11     Jajargenjang(this.alas, this.tinggi, this.sisiMiring);
12
13     @override
14     double luas() => alas * tinggi;
15
16     @override
17     double keliling() => 2 * (alas + sisiMiring);
18 }
19
20 // File: layang_layang.dart
21 class LayangLayang extends BangunDatar {
22     double diagonal1, diagonal2, sisiA, sisiB;
23
24     LayangLayang(this.diagonal1, this.diagonal2, this.sisiA, this.sisiB);
25
26     @override
27     double luas() => 0.5 * diagonal1 * diagonal2;
28
29     @override
30     double keliling() => 2 * (sisiA + sisiB);
31 }
32
33 // File: main.dart
34 void main() {
35     Jajargenjang jajargenjang = Jajargenjang(8, 5, 7);
36     LayangLayang layangLayang = LayangLayang(6, 8, 5, 5);
37     print("Luas Jajargenjang: ${jajargenjang.luas()}");
38     print("Keliling Jajargenjang: ${jajargenjang.keliling()}");
39
40     print("Luas Layang-Layang: ${layangLayang.luas()}");
41     print("Keliling Layang-Layang: ${layangLayang.keliling()}");
42 }
43
44
```

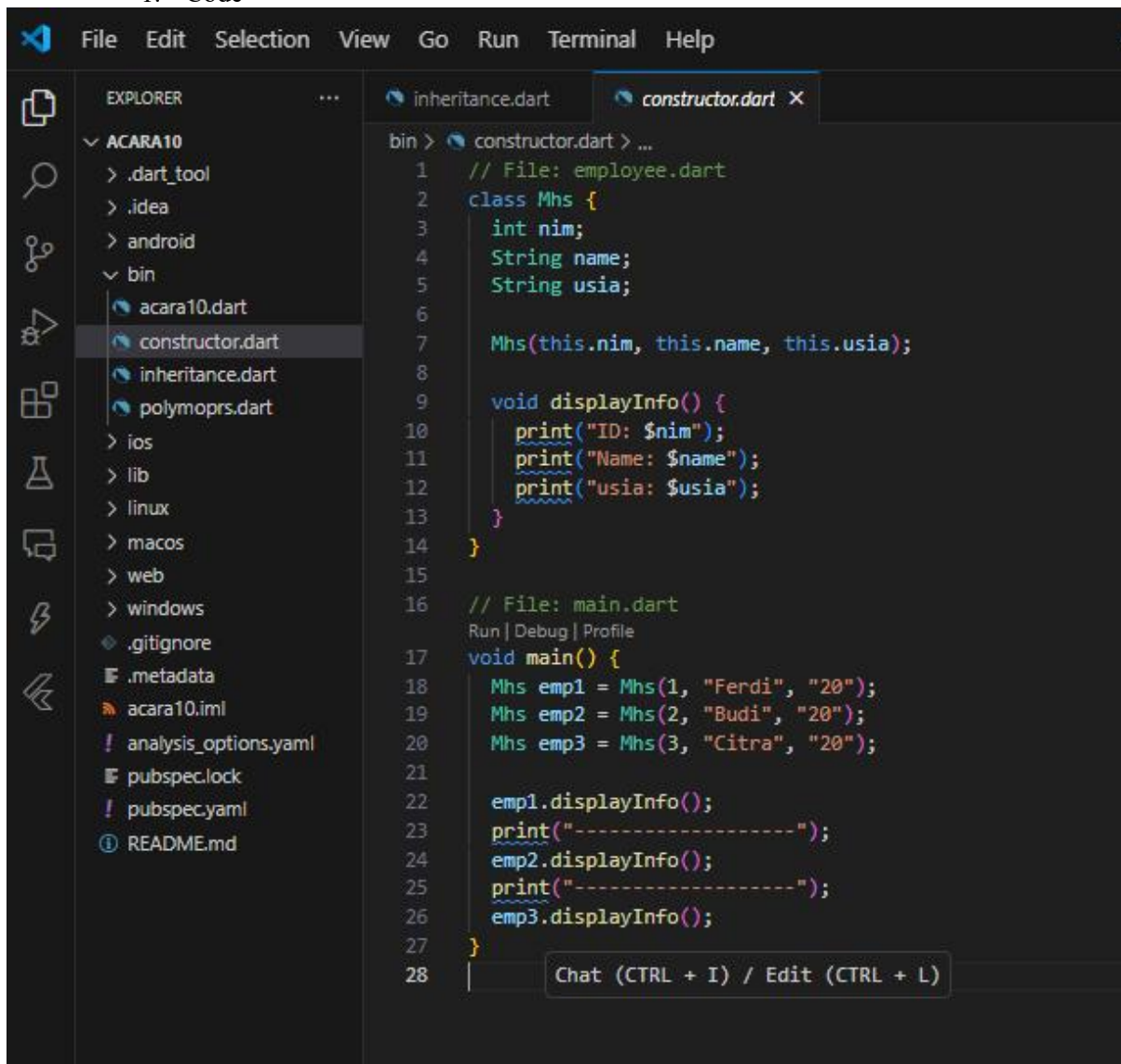
Jadi Bagian yang menggunakan polymorps adalah di bagian variable luas dan keliling dimana dua variable ini bisa digunakan untuk menghitung jajargenjang dan layang layang

2. Hasil

```
Beast Sakura Power: 5.0
Beast Sakura Action: Bumm.....
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin> dart polymoprs.dart
Error when reading 'polymoprs.dart': No such file or directory.
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin> dart polymoprs.dart
Luas Jajargenjang: 40.0
Keliling Jajargenjang: 30.0
Luas Layang-Layang: 24.0
Keliling Layang-Layang: 20.0
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin>
```


3.2.4 Constructors

1. Code



```
bin > constructor.dart > ...
1 // File: employee.dart
2 class Mhs {
3     int nim;
4     String name;
5     String usia;
6
7     Mhs(this.nim, this.name, this.usia);
8
9     void displayInfo() {
10         print("ID: $nim");
11         print("Name: $name");
12         print("usia: $usia");
13     }
14 }
15
16 // File: main.dart
17 void main() {
18     Mhs emp1 = Mhs(1, "Ferdin", "20");
19     Mhs emp2 = Mhs(2, "Budi", "20");
20     Mhs emp3 = Mhs(3, "Citra", "20");
21
22     emp1.displayInfo();
23     print("-----");
24     emp2.displayInfo();
25     print("-----");
26     emp3.displayInfo();
27 }
28
```

Bagian yang mengandung constructors ada dibagian `Mhs(this.nim,this.name,this.usia)` dimana pada bagian ini langsung dilakukan inisialiasasi atribut class dengan nilai yang diberikan

2. Hasil

```
Luas Layang-Layang: 24.0
Keliling Layang-Layang: 20.0
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin> dart constructors.dart
Error when reading 'constructors.dart': No such file or directory.
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin> dart constructor.dart
ID: 1
Name: Ferdi
usia: 20
-----
ID: 2
Name: Budi
usia: 20
-----
ID: 3
Name: Citra
usia: 20
PS E:\1.semester4\mobile framework\TA\semester4\tugas\acara10\bin> 
```