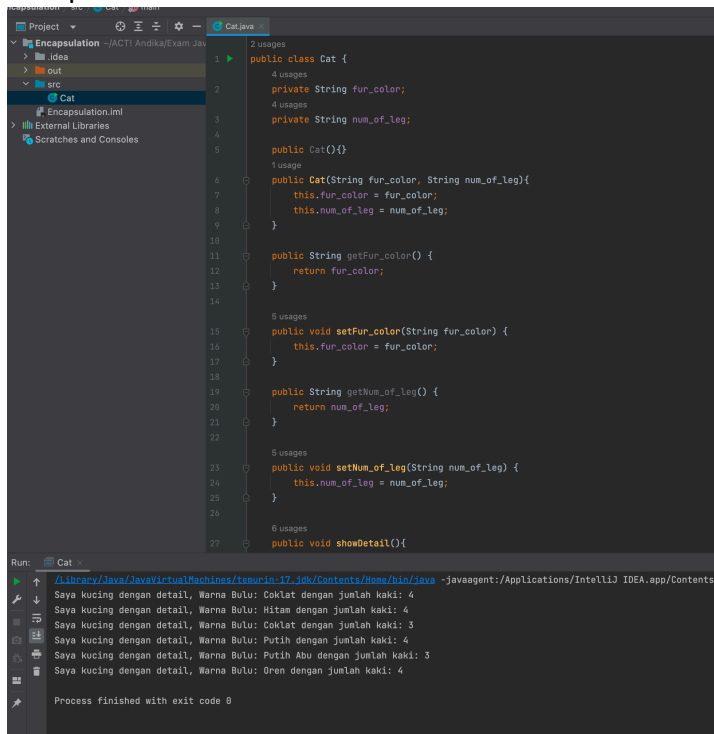


1. Encapsulation

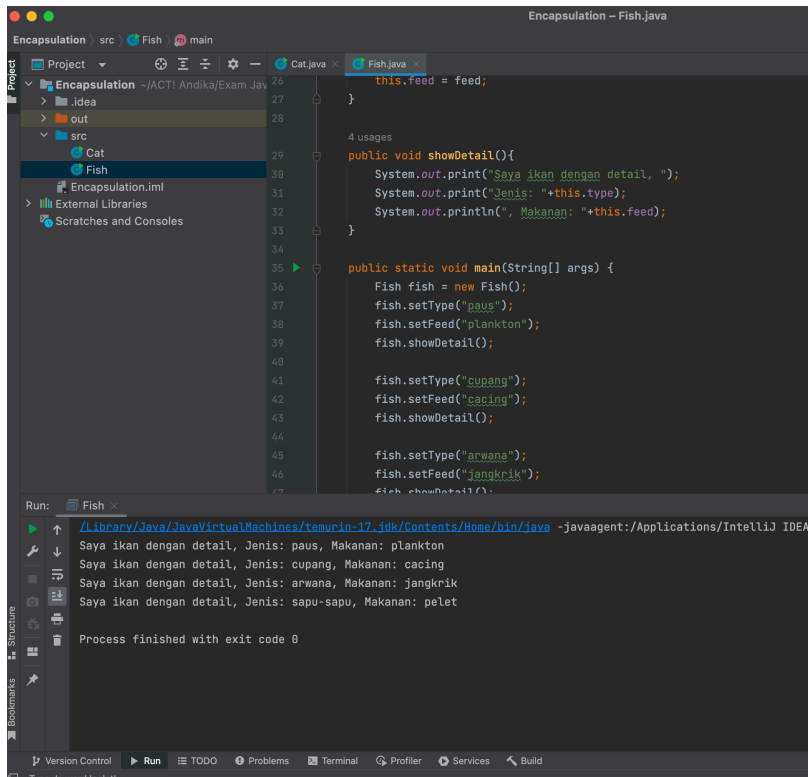


```
Project Encapsulation - [ACT] Andika/Exam Jav
  .idea
  out
  src
    Encapsulation.iml
  External Libraries
  Scratches and Consoles

Cat.java
1 public class Cat {
2     private String fur_color;
3     private String num_of_leg;
4
5     public Cat(){}
6     public Cat(String fur_color, String num_of_leg){
7         this.fur_color = fur_color;
8         this.num_of_leg = num_of_leg;
9     }
10
11     public String getFur_color() {
12         return fur_color;
13     }
14
15     public void setFur_color(String fur_color) {
16         this.fur_color = fur_color;
17     }
18
19     public String getNum_of_leg() {
20         return num_of_leg;
21     }
22
23     public void setNum_of_leg(String num_of_leg) {
24         this.num_of_leg = num_of_leg;
25     }
26
27     public void showDetail(){
28
29 }
30 }
```

Run: Cat

```
/Library/Java/JavaVirtualMachines/temurin-17-jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/
Saya kucing dengan detail, Warna Bulu: Coklat dengan jumlah kaki: 4
Saya kucing dengan detail, Warna Bulu: Hitam dengan jumlah kaki: 4
Saya kucing dengan detail, Warna Bulu: Coklat dengan jumlah kaki: 3
Saya kucing dengan detail, Warna Bulu: Putih dengan jumlah kaki: 4
Saya kucing dengan detail, Warna Bulu: Putih Abu dengan jumlah kaki: 3
Saya kucing dengan detail, Warna Bulu: Oren dengan jumlah kaki: 4
Process finished with exit code 0
```



```
Encapsulation - Fish.java
Project Encapsulation - [ACT] Andika/Exam Jav
  .idea
  out
  src
    Encapsulation.iml
  External Libraries
  Scratches and Consoles

Fish.java
26     this.feed = feed;
27 }
28
29     public void showDetail(){
30         System.out.print("Saya ikan dengan detail, ");
31         System.out.print("Jenis: "+this.type);
32         System.out.println(", Makanan: "+this.feed);
33     }
34
35     public static void main(String[] args) {
36         Fish fish = new Fish();
37         fish.setType("paus");
38         fish.setFeed("plankton");
39         fish.showDetail();
40
41         fish.setType("cupang");
42         fish.setFeed("cacing");
43         fish.showDetail();
44
45         fish.setType("arwana");
46         fish.setFeed("jangkrak");
47         fish.showDetail();
48     }
49 }
```

Run: Fish

```
/Library/Java/JavaVirtualMachines/temurin-17-jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA
Saya ikan dengan detail, Jenis: paus, Makanan: plankton
Saya ikan dengan detail, Jenis: cupang, Makanan: cacing
Saya ikan dengan detail, Jenis: arwana, Makanan: jangkrik
Saya ikan dengan detail, Jenis: sapu-sapu, Makanan: pelet
Process finished with exit code 0
```

The screenshot shows the IntelliJ IDEA interface with the 'Encapsulation - Flower.java' file open. The project structure on the left includes 'Encapsulation.iml' and 'src' containing 'Cat', 'Fish', and 'Flower'. The code in 'Flower.java' defines a 'Flower' class with attributes 'jenis', 'color', and 'num_of_petal', and a 'main' method that creates and displays three flower objects: 'Bangkai' (Hitam, 12 petals), 'Anggrek' (ungu, 4 petals), and 'Mawar' (merah, 6 petals). The 'Run' console at the bottom shows the output of the program, displaying details for each flower and a final line for 'Melati' (kuning, 3 petals) which is not in the code. The status bar at the bottom indicates 'Build completed successfully in 1 sec, 662 ms (workspace: src)'.

```
Encapsulation - Flower.java

41 System.out.print("jenis: " + this.name + ", ");
42 System.out.print("color: " + this.color + ", ");
43 System.out.println("num of petal: " + this.num_of_petal);
44 }
45
46 public static void main(String[] args) {
47     Flower flower = new Flower();
48     flower.setName("Bangkai");
49     flower.setColor("Hitam");
50     flower.setNum_of_petal(12);
51     flower.showDetail();
52
53     flower.setName("Anggrek");
54     flower.setColor("ungu");
55     flower.setNum_of_petal(4);
56     flower.showDetail();
57
58     flower.setName("Mawar");
59     flower.setColor("merah");
60     flower.setNum_of_petal(6);
61     flower.showDetail();
62 }
```

Run: Flower

/Library/Java/JavaVirtualMachines/temurin-17-jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA
Saya bunga dengan detail, jenis: Bangkai, color: Hitam, num of petal: 12
Saya bunga dengan detail, jenis: Anggrek, color: ungu, num of petal: 4
Saya bunga dengan detail, jenis: Mawar, color: merah, num of petal: 6
Saya bunga dengan detail, jenis: Melati, color: kuning, num of petal: 3
Process finished with exit code 0

The screenshot shows the IntelliJ IDEA interface with the 'Encapsulation - Car.java' file open. The project structure on the left includes 'Encapsulation.iml' and 'src' containing 'Cat', 'Fish', 'Flower', and 'Car'. The code in 'Car.java' defines a 'Car' class with attributes 'type', 'color', and 'num_of_tire', and a 'main' method that creates and displays four car objects: 'sedan' (Merah, 4 tires), 'Truk' (Hijau, 12 tires), 'Tronton' (Kuning, 8 tires), and 'Angkot' (Coklat, 6 tires). The 'Run' console at the bottom shows the output of the program, displaying details for each car. The status bar at the bottom indicates 'Build completed successfully in 1 sec, 978 ms (workspace: src)'.

```
Encapsulation - Car.java

41 System.out.print("Type: " + this.type + ", ");
42 System.out.print("color: " + this.color + ", ");
43 System.out.println("num of tire: " + this.num_of_tire);
44 }
45
46 public static void main(String[] args) {
47     Car car = new Car();
48     car.setType("sedan");
49     car.setColor("Merah");
50     car.setNum_of_tire(4);
51     car.showDetail();
52
53     car.setType("Truk");
54     car.setColor("Hijau");
55     car.setNum_of_tire(12);
56     car.showDetail();
57
58     car.setType("Tronton");
59     car.setColor("Kuning");
60     car.setNum_of_tire(8);
61     car.showDetail();
62 }
```

Run: Car

/Library/Java/JavaVirtualMachines/temurin-17-jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA
Saya mobil dengan detail, Type: sedan, color: Merah, num of tire: 4
Saya mobil dengan detail, Type: Truk, color: Hijau, num of tire: 12
Saya mobil dengan detail, Type: Tronton, color: Kuning, num of tire: 8
Saya mobil dengan detail, Type: Angkot, color: Coklat, num of tire: 6
Process finished with exit code 0

2. Data Abstraction

The screenshot displays the IntelliJ IDEA IDE interface. The top toolbar shows icons for Project, Calculator.java, Calculator, and main. The left sidebar shows the Project view with the following structure:

- Abstraction ~ /ACT! Andika/Exam Java Day
 - .idea
 - out
 - src
 - Calculator.java
 - Calculator
 - Matematika
 - Operasi
 - Abstraction.iml
 - External Libraries
 - Scratches and Consoles

The main editor shows the code for `Calculator.java`:

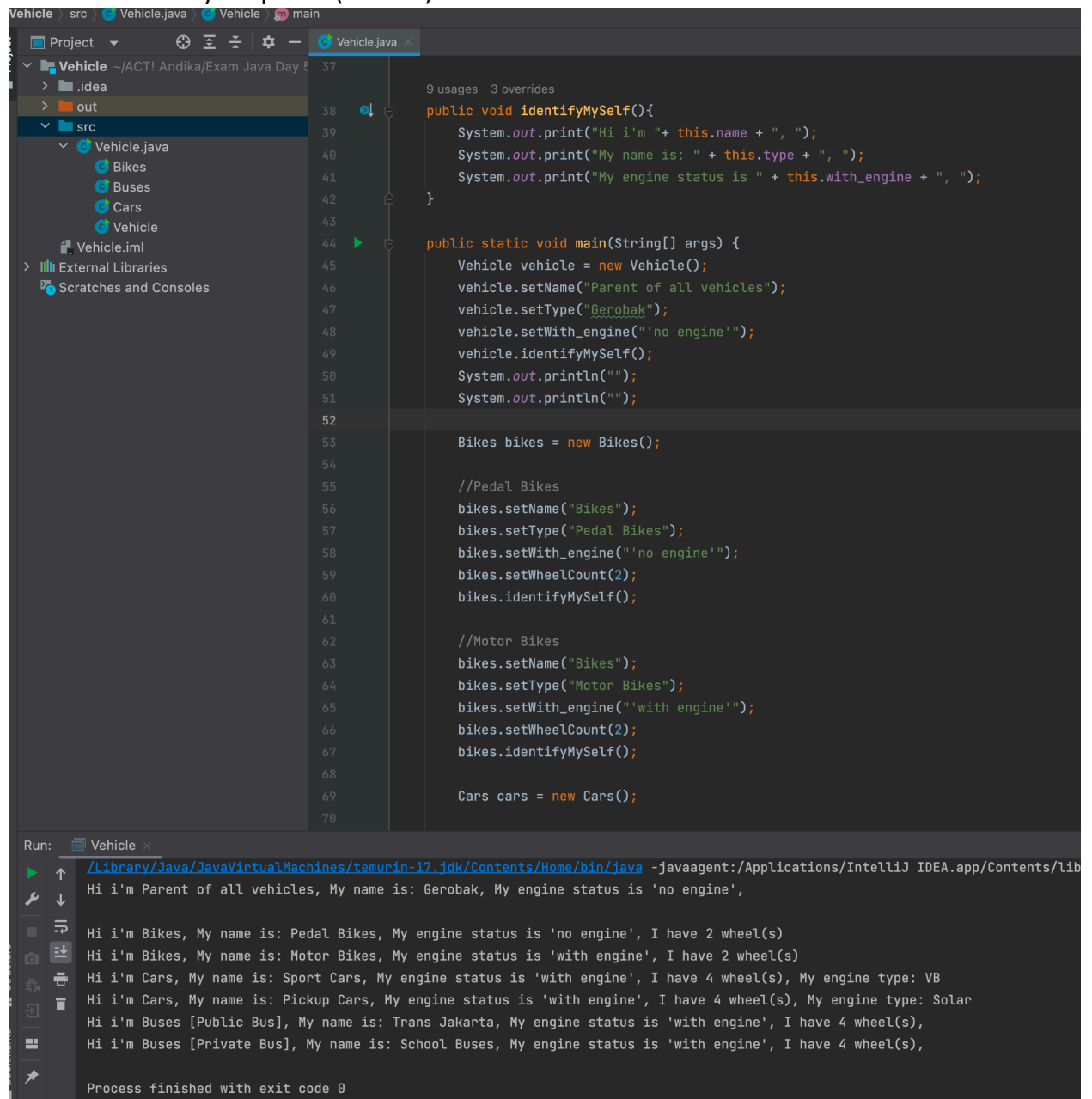
```
1 import java.util.Scanner;
2 interface Operasi {
3     int hitungJumlah();
4     int hitungKurang();
5 }
6 class Matematika implements Operasi {
7     private int bilangan1;
8     private int bilangan2;
9
10    public Matematika() {
11    }
12
13    public int getBilangan1() {
14        return bilangan1;
15    }
16 }
```

The bottom console shows the output of the program:

```
Run: Calculator x
/Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.a
+++++++ CALCULATOR ++++++
1: Open Calculator
99: Exit
Masukkan pilihan anda:
Bilangan 1: 25
Bilangan 2: 67
+++++++ Masukkan Operasi ++++++
1. Penjumlahan
2. Pengurangan
+++++++ Masukkan Operasi ++++++
Masukkan Pilihan Operasi: 2
Jumlah: -42

Process finished with exit code 0
```

3. Inheritance & Polymorphism (Vehicle)



The screenshot displays the IntelliJ IDEA IDE with the `Vehicle.java` file open. The left sidebar shows the project structure with `Vehicle` as the root, containing `src` and `out` directories. The `src` directory lists `Bikes`, `Buses`, `Cars`, and `Vehicle`. The `Vehicle.java` file is shown with the following code:

```
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70
```

```
public void identifyMySelf(){  
    System.out.print("Hi i'm " + this.name + ", ");  
    System.out.print("My name is: " + this.type + ", ");  
    System.out.print("My engine status is " + this.with_engine + ", ");  
}  
  
public static void main(String[] args) {  
    Vehicle vehicle = new Vehicle();  
    vehicle.setName("Parent of all vehicles");  
    vehicle.setType("Gerobak");  
    vehicle.setWith_engine("no engine");  
    vehicle.identifyMySelf();  
    System.out.println("");  
    System.out.println("");  
  
    Bikes bikes = new Bikes();  
  
    //Pedal Bikes  
    bikes.setName("Bikes");  
    bikes.setType("Pedal Bikes");  
    bikes.setWith_engine("no engine");  
    bikes.setWheelCount(2);  
    bikes.identifyMySelf();  
  
    //Motor Bikes  
    bikes.setName("Bikes");  
    bikes.setType("Motor Bikes");  
    bikes.setWith_engine("with engine");  
    bikes.setWheelCount(2);  
    bikes.identifyMySelf();  
  
    Cars cars = new Cars();
```

The bottom of the image shows the Run console output for the `Vehicle` class:

```
Run: Vehicle  
/Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/Lib  
Hi i'm Parent of all vehicles, My name is: Gerobak, My engine status is 'no engine',  
  
Hi i'm Bikes, My name is: Pedal Bikes, My engine status is 'no engine', I have 2 wheel(s)  
Hi i'm Bikes, My name is: Motor Bikes, My engine status is 'with engine', I have 2 wheel(s)  
Hi i'm Cars, My name is: Sport Cars, My engine status is 'with engine', I have 4 wheel(s), My engine type: VB  
Hi i'm Cars, My name is: Pickup Cars, My engine status is 'with engine', I have 4 wheel(s), My engine type: Solar  
Hi i'm Buses [Public Bus], My name is: Trans Jakarta, My engine status is 'with engine', I have 4 wheel(s),  
Hi i'm Buses [Private Bus], My name is: School Buses, My engine status is 'with engine', I have 4 wheel(s),  
  
Process finished with exit code 0
```

4. Inheritance & Polymorphism (Animal)

The screenshot displays the IntelliJ IDEA IDE with a project named 'Animal'. The 'src' directory contains files for 'Animal.java', 'Animal', 'Carnivor', 'Herbivor', and 'Omnivor', along with 'Animal.iml'. The 'Animal.java' file is open, showing a class hierarchy and a main method. The code defines an 'Animal' class with a 'name' attribute and an 'identifyMyself()' method. It also defines three subclasses: 'Herbivor', 'Carnivor', and 'Omnivor', each with its own 'identifyMyself()' method. The 'main' method creates instances of these classes and calls their 'identifyMyself()' methods. The output window shows the results of the execution, displaying the name, food, and teeth for each animal.

```
55     System.out.print("My Name is " + this.name);
56 }
57
58 public static void main(String[] args) {
59     Animal animal = new Animal( type: "Parent of All Animal", name: "Binatang");
60     animal.identifyMyself();
61     System.out.println("");
62
63     Herbivor herbivor = new Herbivor();
64     herbivor.setType("Herbivor");
65     herbivor.setName("Kambing");
66     herbivor.setFood("tumbuhan");
67     herbivor.setTeeth("tumpul");
68     herbivor.identifyMyself();
69
70     Carnivor carnivor = new Carnivor();
71     carnivor.setType("Carnivor");
72     carnivor.setName("Singa");
73     carnivor.setFood("daging");
74     carnivor.setTeeth("tajam");
75     carnivor.identifyMyself();
76
77     Omnivor omnivor = new Omnivor();
78     omnivor.setType("Omnivor");
79     omnivor.setName("Ayam");
80     omnivor.setFood("semua");
81     omnivor.setTeeth("tajam dan tumpul");
82     omnivor.identifyMyself();
83 }
```

Run: Animal x

```
/Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.
Hi I'm Parent of ALL Animal, My Name is Binatang
Hi I'm Herbivor, My Name is Kambing, My Food is 'tumbuhan, I have tumpul teeth
Hi I'm Carnivor, My Name is Singa, My Food is 'daging, I have tajam teeth
Hi I'm Omnivor, My Name is Ayam, My Food is 'semua, I have tajam dan tumpul teeth
```

5. Mohon maaf belum bisa dikerjakan karena keterbatasan waktu