LAPORAN PRAKTIKUM 6

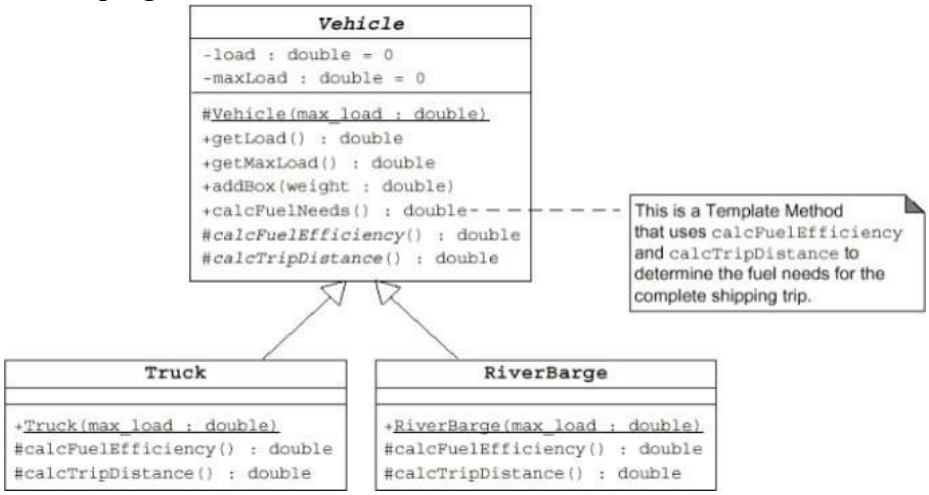
Oleh :

Filusive Nathan Fernanda

21091397073

D4 MANAJEMEN INFORMATIKA
FAKULTAS VOKASI
UNIVERSITAS NEGERI
SURABAYA TAHUN 2022

1. Buat program berdasarkan UML berikut



- **Source Code**

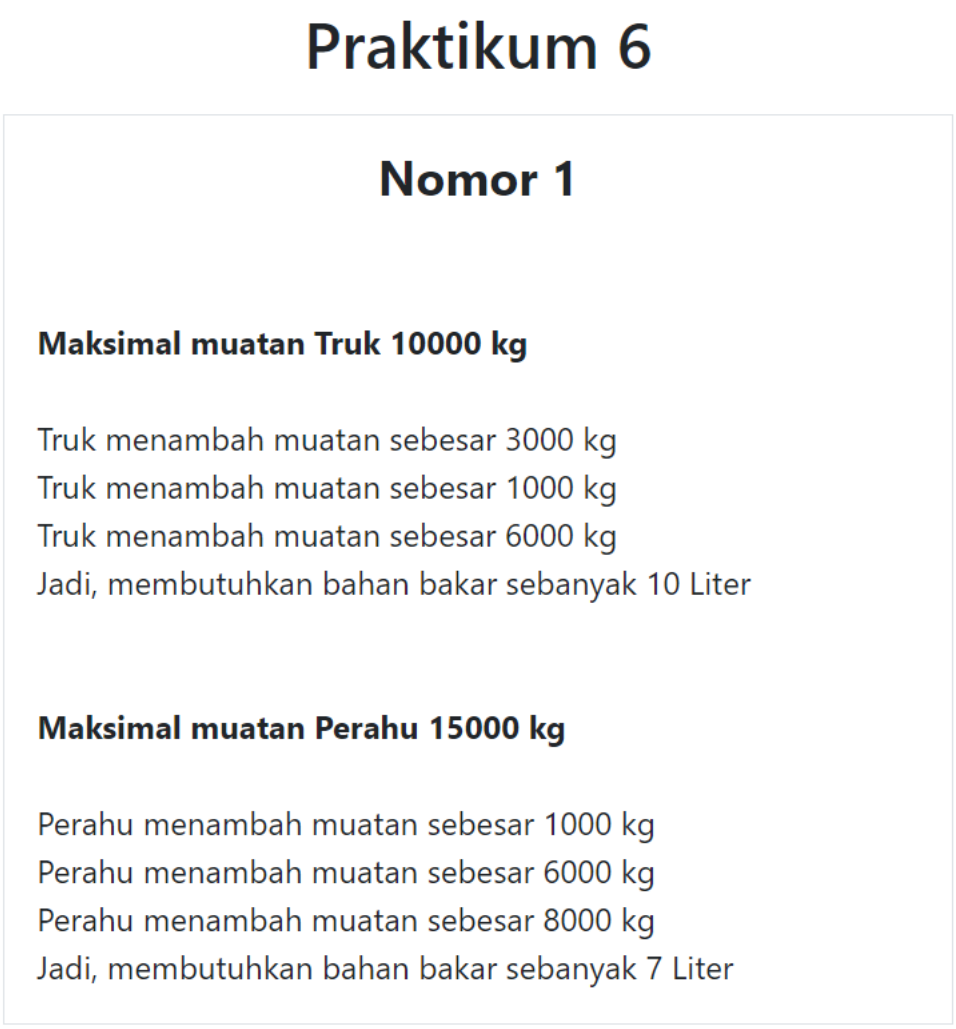
```php
<?php
require_once "1a.php"; ?>

<!DOCTYPE html>
<html lang="id">

<head>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" crossorigin="anonymous">

    <title>Praktikum 6</title>
</head>

<body>
    <div class="container">
        <br>
        <h2 class="text-center">Praktikum 6</h2>
        <div class="row">
            <div class="col-5 mx-auto border p-3 mt-2">
                <h4 class="text-center"><strong>Nomor 1</strong></h4>
                <br><br>
                <b><?=$truck->getMaxLoad() . " kg" ?> <br></b>
                <br>
                <?=$truck->addBox(3000) . " kg" ?> <br>
                <?=$truck->addBox(1000) . " kg" ?> <br>
                <?=$truck->addBox(6000) . " kg" ?> <br>

                    <?php echo "Jadi, membutuhkan bahan bakar sebanyak " . $truck->calcFuelNeeds() . " Liter" . "<br>"; ?>
                <br>
                <br>
                <?=$truck->addBox(3000) . " kg" ?> <br>
                <?=$truck->addBox(1000) . " kg" ?> <br>
                <?=$truck->addBox(6000) . " kg" ?> <br>

                    <?php echo "Jadi, membutuhkan bahan bakar sebanyak " . $truck->calcFuelNeeds() . " Liter" . "<br>"; ?>
                <br>
                <br>
                <b><?=$riverBarge->getMaxLoad() . " kg" ?> <br></b>
                <br>
                <?=$riverBarge->addBox(1000) . " kg" ?> <br>
                <?=$riverBarge->addBox(6000) . " kg" ?> <br>
                <?=$riverBarge->addBox(8000) . " kg" ?> <br>

                    <?php echo "Jadi, membutuhkan bahan bakar sebanyak " . $riverBarge->calcFuelNeeds() . " Liter"; ?>
            </div>
        </div>
    </div>
</body>
</html>
```

```php
<?php

require_once "1b.php";

1 reference | 0 implementations
class Truck extends Vehicle
{
    1 reference | 0 overrides | prototype
    public function __construct($maxLoad, $name)
    {
        $this->maxLoad = $maxLoad;
        $this->name = $name;
    }

    2 references | 0 overrides | prototype
    public function calcFuelNeeds()
    {
        $fuel = $this->calcFuelEfficiency();
        $trip = $this->calcTripDistance();

        return ceil($fuel /= $trip);
    }
}

1 reference | 0 implementations
class RiverBarge extends Vehicle
{
    1 reference | 0 overrides | prototype
    public function __construct($maxLoad, $name)
    {
        $this->maxLoad = $maxLoad;
        $this->name = $name;
    }

    2 references | 0 overrides | prototype
    public function calcFuelNeeds()
    {
        $fuel = $this->calcFuelEfficiency();
        $trip = $this->calcTripDistance();

        return ceil($fuel /= $trip);
    }
}
$truck = new Truck(10000, "Truk");
$riverBarge = new RiverBarge(15000, "Perahu");
```

```php
<?php

2 references | 2 implementations
abstract class Vehicle
{
    4 references
    private $load = 0;
    protected $maxLoad = 0,
        $name;

    0 references | 2 overrides
    protected function __construct($maxLoad, $name)
    {
        $this->$maxLoad = $maxLoad;
        $this->$name = $name;
    }

    0 references | 0 overrides
    public function getLoad()
    {
        return $this->load;
    }

    2 references | 0 overrides
    public function getMaxLoad()
    {
        echo "Maksimal muatan " . $this->name . " ";
        return $this->maxLoad;
    }

```

```
        6 references | 0 overrides
26      public function addBox($weight)
27      {
28          if ($this->load >= $this->maxLoad) {
29              echo "$this->name menambah muatan sebesar $weight <br>";
30              echo "Muatan telah penuh tidak bisa menambah lagi";
31          } else {
32              $this->load += $weight;
33              echo "$this->name menambah muatan sebesar $weight";
34          }
35      }
36
        2 references | 2 overrides
37      abstract public function calcFuelNeeds();
38
        2 references | 0 overrides
39      protected function calcFuelEfficiency()
40      {
41          $range = 50000000;
42          $range /= $this->load;
43          return $range;
44      }
45
        2 references | 0 overrides
46      protected function calcTripDistance()
47      {
48          return 500;
49      }
50  }
```

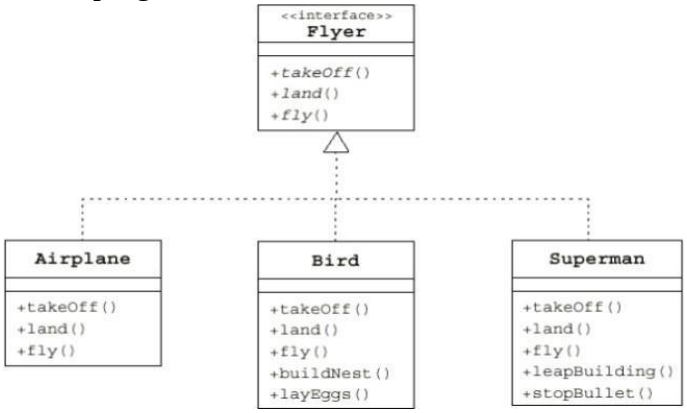
- **Output**



- **Analisis**

  Implementasi dari abstract class pada class Vehicle, method calcFuelNeeds digunakan untuk menghitung bahan bakar yang digunakan. Abstract method diletakkan pada class Vehicle sebagai parent class dan diakses oleh child classnya yaitu class Truk, dan class RiverBarge yang akan mengembalikan nilai yang dihasilkan dari pembagian 2 method yaitu calcFuelEfficiency dancalcTripDistance.

2. Buat program berdasarkan UML berikut

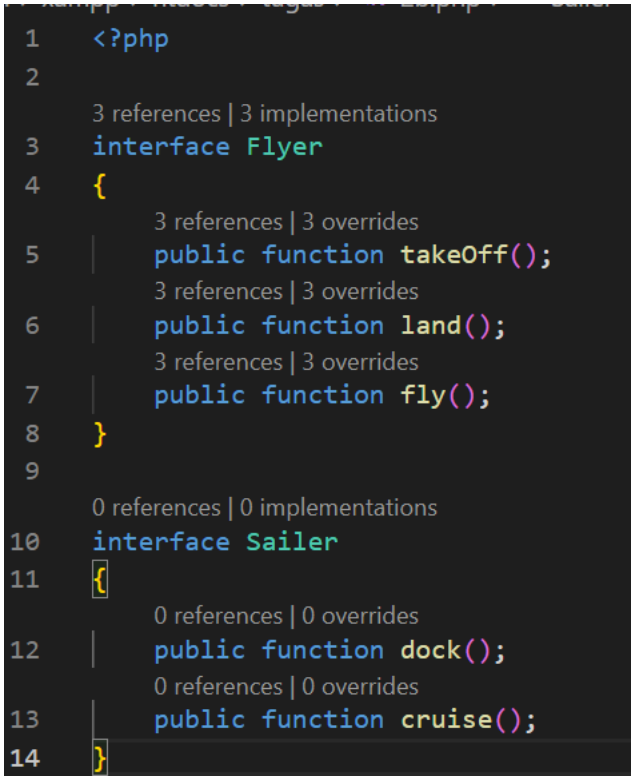


- **Source Code**

```php
<?php
require_once "2a.php"; ?>

<!DOCTYPE html>
<html lang="en">

<head>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" crossorigin="anonymous">

    <title>Praktikum 6</title>
</head>

<body>
    <div class="container">
        <br>
        <h2 class="text-center">Praktikum 6</h2>
        <div class="row">
            <div class="col-5 mx-auto border p-3 mt-2">
                <h4 class="text-center"><strong>Nomor 2</strong></h4>
                <br><br>
                <b><?php echo "Superman"; ?></b> <br>
                <?= $superman->land() ?> <br>
                <?= $superman->takeOff() ?> <br>
                <?= $superman->fly() ?> <br>
                <?= $superman->leapBuilding() ?> <br>
                <?= $superman->stopBullet() ?> <br>
                <br>
                <b><?php echo "Bird"; ?></b> <br>
                <?= $bird->buildNest() ?> <br>
                <?= $bird->takeOff() ?> <br>
                <?= $bird->fly() ?> <br>
                <?= $bird->land() ?> <br>
                <?= $bird->layEggs() ?> <br>
                <br>
                <b><?php echo "Airplane"; ?></b> <br>
                <?= $airplane->takeOff() ?> <br>
                <?= $airplane->fly() ?> <br>
                <?= $airplane->land() ?> <br>
            </div>
        </div>
    </div>
</body>
</html>
```

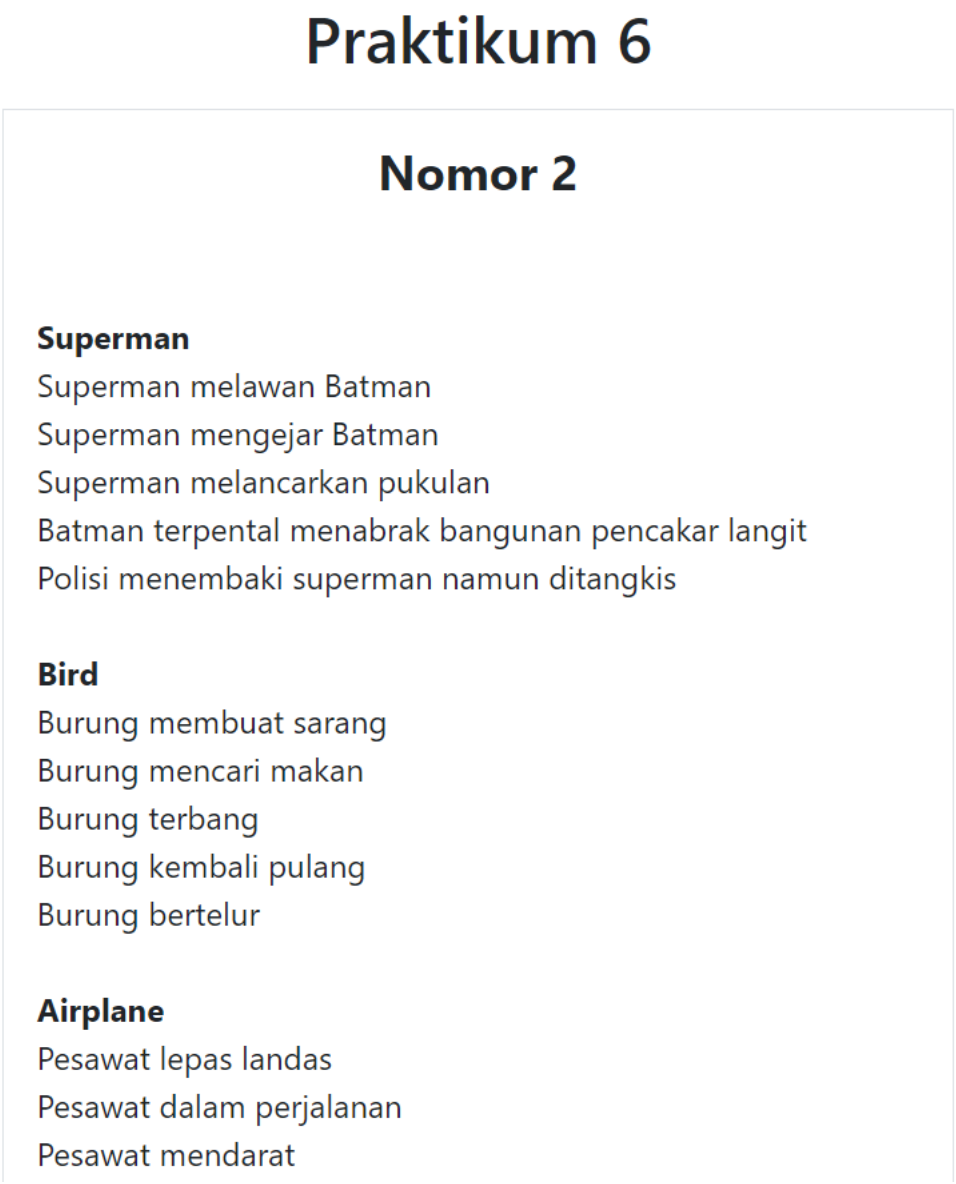
```php
<?php

require_once "2b.php";

1 reference | 0 implementations
class Airplane implements Flyer
{
    3 references | 0 overrides
    public function takeOff()
    {
        return "Pesawat lepas landas";
    }

    3 references | 0 overrides
    public function land()
    {
        return "Pesawat mendarat";
    }

    3 references | 0 overrides
    public function fly()
    {
        return "Pesawat dalam perjalanan";
    }
}

1 reference | 0 implementations
class Bird implements Flyer
{
    3 references | 0 overrides
    public function takeOff()
```

```php
    }

    3 references | 0 overrides
    public function land()
    {
        return "Superman melawan Batman";
    }

    3 references | 0 overrides
    public function fly()
    {
        return "Superman melancarkan pukulan";
    }

    1 reference | 0 overrides
    public function leapBuilding()
    {
        return "Batman terpental menabrak bangunan pencakar langit";
    }

    1 reference | 0 overrides
    public function stopBullet()
    {
        return "Polisi menembaki superman namun ditangkis";
    }
}

$airplane = new Airplane();
$bird = new Bird();
$superman = new Superman();
```



```php
<?php

3 references | 3 implementations
interface Flyer
{
    3 references | 3 overrides
    public function takeOff();
    3 references | 3 overrides
    public function land();
    3 references | 3 overrides
    public function fly();
}

0 references | 0 implementations
interface Sailer
{
    0 references | 0 overrides
    public function dock();
    0 references | 0 overrides
    public function cruise();
}
```

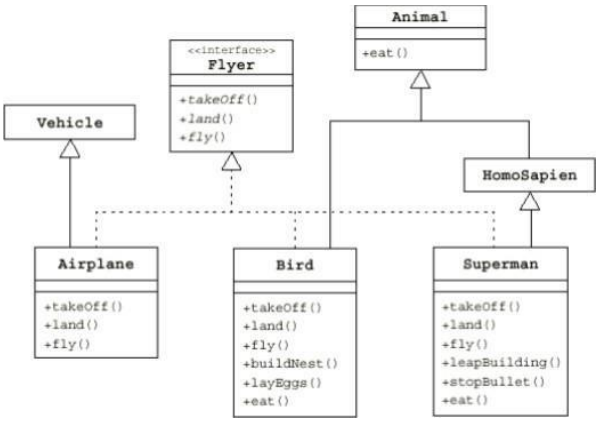
- **Output**



- **Analisis**

  Implementasi Polymhorpism dengan penggunaan Interface Flyer sehingga semua class yang Implements dari interface Fyler harus memiliki method takeoff, land, dan fly.

3. Buat program berdasarkan UML berikut



- **Source Code**

```php
<?php
require_once "3a.php"; ?>

<!DOCTYPE html>
<html lang="id">

<head>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" crossorigin="anonymous">

    <title>Praktikum 6</title>
</head>
<body>
    <div class="container">
        <h2 class="text-center">Praktikum 6</h2>
        <div class="row">
            <div class="col-5 mx-auto border p-3 mt-2">
                <h4 class="text-center"><strong>Nomor 3</strong></h4>
                <br><br>
                <?= $harimau->eat() ?> <br>
                <?= $manusia->eat() ?> <br>
                <br>
                <b><?= $airplane2->getMaxLoad() . " kg" ?> <br></b>
                <?= $airplane2->addBox(5000) . " kg" ?> <br>
                <?= $airplane2->addBox(7000) . " kg" ?> <br>
                <?= $airplane2->addBox(3000) . " kg" ?> <br>
                <?= $airplane2->addBox(4000) . " kg" ?> <br>
                <?= $airplane2->takeOff() ?> <br>
                <?= $airplane2->fly() ?> <br>
                <?= $airplane2->land() ?> <br>
```

```
                    <?= $airplane2->fly() ?> <br>
                    <?= $airplane2->land() ?> <br>

                    <?php echo "Jadi, membutuhkan bahan bakar sebanyak " .
                        $airplane2->calcFuelNeeds() .
                        " Liter" .
                        "<br>"; ?>
                    <br>
                    <?= $superman2->eat() ?> <br>
                    <?= $superman2->land() ?> <br>
                    <?= $superman2->takeOff() ?> <br>
                    <?= $superman2->fly() ?> <br>
                    <?= $superman2->leapBuilding() ?> <br>
                    <?= $superman2->stopBullet() ?> <br>
                </div>
            </div>
        </div>
    </body>
    </html>
```

```php
<?php

require_once "3b.php";
require_once "3c.php";

2 references | 2 implementations
class Animal
{
    6 references
    protected $name;

    1 reference | 0 overrides
    public function __construct($name)
    {
        $this->name = $name;
    }

    3 references | 0 overrides
    public function eat()
    {
        return $this->name . " sedang makan";
    }
}

2 references | 1 implementation
class Homosapiens extends Animal
{
}

1 reference | 0 implementations
class Airplane2 extends Vehicle implements Flyer
```

```php
{
    1 reference | 0 overrides | prototype
    public function __construct($maxLoad, $name)
    {
        $this->maxLoad = $maxLoad;
        $this->name = $name;
    }

    5 references | 0 overrides
    public function takeOff()
    {
        return "$this->name lepas landas";
    }

    5 references | 0 overrides
    public function land()
    {
        return "$this->name mendarat";
    }

    5 references | 0 overrides
    public function fly()
    {
        return "$this->name dalam perjalanan";
    }

    3 references | 0 overrides | prototype
    public function calcFuelNeeds()
    {
        $fuel = $this->calcFuelEfficiency();
```

```php
        $trip = $this->calcTripDistance();

        return ceil($fuel /= $trip);
    }
}

1 reference | 0 implementations
class Superman2 extends Homosapiens implements Flyer
{
    5 references | 0 overrides
    public function takeOff()
    {
        return "$this->name mengejar Batman";
    }

    5 references | 0 overrides
    public function land()
    {
        return "$this->name melawan Batman";
    }

    5 references | 0 overrides
    public function fly()
    {
        return "$this->name melancarkan pukulan";
    }

    1 reference | 0 overrides
    public function leapBuilding()
    {
        return "Batman terpental menabrak bangunan pencakar langit";
    }

    1 reference | 0 overrides
    public function stopBullet()
    {
        return "Polisi menembaki $this->name namun ditangkis";
    }
}

$harimau = new Animal("Harimau");
$manusia = new Homosapiens("Núñez");
$airplane2 = new Airplane2(20000, "Batik Air");
$superman2 = new Superman2("Superman");
```

```php
<?php

3 references | 3 implementations
abstract class Vehicle
{
    8 references
    private $load = 0;
    protected $maxLoad = 0,
        $name;

    0 references | 3 overrides
    protected function __construct($maxLoad, $name)
    {
        $this->$maxLoad = $maxLoad;
        $this->$name = $name;
    }

    0 references | 0 overrides
    public function getLoad()
    {
        return $this->load;
    }

    3 references | 0 overrides
    public function getMaxLoad()
    {
        echo "Maksimal muatan " . $this->name . " ";
        return $this->maxLoad;
    }

```

```php
    public function addBox($weight)
    {
        if ($this->load >= $this->maxLoad) {
            echo "$this->name menambah muatan sebesar $weight <br>";
            echo "Muatan telah penuh tidak bisa menambah lagi";
        } else {
            $this->load += $weight;
            echo "$this->name menambah muatan sebesar $weight";
        }
    }

    3 references | 3 overrides
    abstract public function calcFuelNeeds();

    3 references | 0 overrides
    protected function calcFuelEfficiency()
    {
        $range = 50000000;
        $range /= $this->load;
        return $range;
    }

    3 references | 0 overrides
    protected function calcTripDistance()
    {
        return 500;
    }
}
```

```php
<?php

5 references | 5 implementations
interface Flyer
{
    5 references | 5 overrides
    public function takeOff();
    5 references | 5 overrides
    public function land();
    5 references | 5 overrides
    public function fly();
}

0 references | 0 implementations
interface Sailer
{
    0 references | 0 overrides
    public function dock();
    0 references | 0 overrides
    public function cruise();
}
```

- **Output**

Nomor 3

Harimau sedang makan
Núñez sedang makan

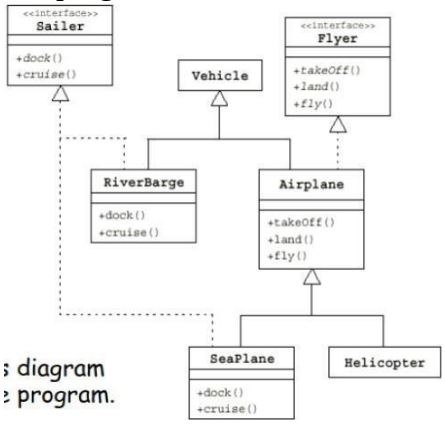
Maksimal muatan Batik Air 20000 kg
Batik Air menambah muatan sebesar 5000 kg
Batik Air menambah muatan sebesar 7000 kg
Batik Air menambah muatan sebesar 3000 kg
Batik Air menambah muatan sebesar 4000 kg
Batik Air lepas landas
Batik Air dalam perjalanan
Batik Air mendarat
Jadi, membutuhkan bahan bakar sebanyak 6 Liter

Superman sedang makan
Superman melawan Batman
Superman mengejar Batman
Superman melancarkan pukulan
Batman terpental menabrak bangunan pencakar langit
Polisi menembaki Superman namun ditangkis

- **Analisis**

  Terdapat interface Flyer dan abstract class Vehicle. Class airplane merupakan implementasi dari interface Flyer dan turunan dari Vehicle sehingga class Airplane harus memiliki method calcFuelNeeds, takeoff, land, dan fly. Class Bird adalah implementasi dari Flyer dan turunan dari Animal sehingga memiliki method takeoff, land, fly, dan eat. Class Superman turunan dari homosapiens yang juga turunan dari Animal, serta implementasi dari interface Flyer. Maka class Superman memiliki method eat, takeoff, land, fly.

4. Buat program berdasarkan UML berikut



- **Source Code**

```php
<?php
    require_once '4a.php';
?>

<!DOCTYPE html>
<html lang="en">

<head>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" crossorigin="anonymous">

    <title>Praktikum 6</title>
</head>
<body>
    <h2 class="text-center">Praktikum 6</h2>
        <div class="container">
            <div class="row">
                <h4 class="text-center"><strong>Nomor 4</strong></h4>
                <br><br>
                    <div class="col-4 mx-auto border p-2 mt-2">
                        <b><?= $riverBarge2->getMaxLoad() . ' kg'; ?> <br></b>
                        <?= $riverBarge2->addBox(12000) . ' kg'; ?> <br>
                        <?= $riverBarge2->addBox(14000) . ' kg'; ?> <br>
                        <?= $riverBarge2->addBox(1000) . ' kg'; ?> <br>
                        <?= $riverBarge2->addBox(3000) . ' kg'; ?> <br>
                        <?= $riverBarge2->dock(); ?> <br>
                        <?= $riverBarge2->cruise(); ?> <br>
                        <?php
                                echo "Jadi, membutuhkan bahan bakar sebanyak " . $riverBarge2->calcFuelNeeds() . ' Liter'. '<br>';
                            ?>
```

```
                    </div>
                    <div class="col-4 mx-auto border p-2 mt-2">
                        <b><?= $seaPlane->getMaxLoad() . ' kg'; ?> <br></b>
                        <?= $seaPlane->addBox(12000) . ' kg'; ?> <br>
                        <?= $seaPlane->addBox(8000) . ' kg'; ?> <br>
                        <?= $seaPlane->dock(); ?> <br>
                        <?= $seaPlane->cruise(); ?> <br>
                        <?= $seaPlane->takeOff(); ?> <br>
                        <?= $seaPlane->fly(); ?> <br>
                        <?= $seaPlane->land(); ?> <br>
                        <?php
                            echo "Jadi, membutuhkan bahan bakar sebanyak " . $seaPlane->calcFuelNeeds() . ' Liter'. '<br>';
                        ?>
                    </div>
                    <div class="col mx-auto border p-2 mt-2">
                        <b><?= $helicopter->getMaxLoad() . ' kg'; ?> <br></b>
                        <?= $helicopter->addBox(8000) . ' kg'; ?> <br>
                        <?= $helicopter->addBox(2000) . ' kg'; ?> <br>
                        <?= $helicopter->takeOff(); ?> <br>
                        <?= $helicopter->fly(); ?> <br>
                        <?= $helicopter->land(); ?> <br>
                        <?php
                            echo "Jadi, membutuhkan bahan bakar sebanyak " . $helicopter->calcFuelNeeds() . ' Liter'. '<br>';
                        ?>
                    </div>
                </div>
            </div>
        </body>
        </html>
```

```php
<?php

require_once '4b.php';
require_once '4c.php';

1 reference | 0 implementations
class RiverBarge2 extends Vehicle implements Sailer {
    1 reference | 0 overrides | prototype
    public function __construct($maxLoad, $name) {
        $this->maxLoad = $maxLoad;
        $this->name = $name;
    }

    6 references | 0 overrides | prototype
    public function calcFuelNeeds() {
        $fuel = $this->calcFuelEfficiency();
        $trip = $this->calcTripDistance();

        return ceil($fuel /= $trip);
    }

    2 references | 0 overrides
    public function dock() {
        return $this->name . ' berada di dermaga';
    }

    2 references | 0 overrides
    public function cruise() {
        return $this->name . ' sedang berlayar';
    }
```

```php
class Airplane2 implements Flyer {
    5 references | 0 overrides
    public function takeOff() {
        return 'Pesawat lepas landas';
    }
    5 references | 0 overrides
    public function land() {
        return 'Pesawat mendarat';
    }
    5 references | 0 overrides
    public function fly() {
        return 'Pesawat dalam perjalanan';
    }
}

1 reference | 0 implementations
class SeaPlane extends Vehicle implements Sailer
    1 reference | 0 overrides | prototype
    public function __construct($maxLoad, $name)
        $this->maxLoad = $maxLoad;
        $this->name = $name;
    }

    6 references | 0 overrides | prototype
    public function calcFuelNeeds() {
        $fuel = $this->calcFuelEfficiency();
        $trip = $this->calcTripDistance();

        return ceil($fuel /= $trip);
    }
```

```php
    public function dock() {
        return $this->name . ' berada di dermaga';
    }

    2 references | 0 overrides
    public function cruise() {
        return $this->name . ' sedang berlayar';
    }

    1 reference | 0 overrides
    public function takeOff() {
        return $this->name . ' lepas landas';
    }

    1 reference | 0 overrides
    public function land() {
        return $this->name . ' mendarat';
    }

    1 reference | 0 overrides
    public function fly() {
        return $this->name . ' dalam perjalanan';
    }
}

1 reference | 0 implementations
class Helicopter extends Vehicle {
    1 reference | 0 overrides | prototype
    public function __construct($maxLoad, $name) {
        $this->maxLoad = $maxLoad;
        $this->name = $name;
```

```php

    6 references | 0 overrides | prototype
    public function calcFuelNeeds() {
        $fuel = $this->calcFuelEfficiency();
        $trip = $this->calcTripDistance();

        return ceil($fuel /= $trip);
    }
    1 reference | 0 overrides
    public function takeOff() {
        return $this->name . ' lepas landas';
    }

    1 reference | 0 overrides
    public function land() {
        return $this->name . ' mendarat';
    }

    1 reference | 0 overrides
    public function fly() {
        return $this->name . ' dalam perjalanan';
    }
}

$riverBarge2 = new RiverBarge2(30000, 'Storeum');
$seaPlane = new SeaPlane(20000, 'Falcon');
$helicopter = new Helicopter(10000, 'Hurricane');
```

```php
<?php

6 references | 7 implementations
abstract class Vehicle {
    12 references
    private $load = 0;
    protected $maxLoad = 0, $name;

    0 references | 6 overrides
    protected function __construct($maxLoad, $name) {
        $this->$maxLoad = $maxLoad;
        $this->$name = $name;
    }

    0 references | 0 overrides
    public function getLoad() {
        return $this->load;
    }

    6 references | 0 overrides
    public function getMaxLoad() {
        echo 'Maksimal muatan ' . $this->name . ' ' ;
        return $this->maxLoad;
    }

    18 references | 0 overrides
    public function addBox($weight) {
        if ($this->load >= $this->maxLoad) {
            echo "$this->name menambah muatan sebesar $weight <br>";
            echo 'Muatan telah penuh tidak bisa menambah lagi';
        }else {
```

```php
    6 references | 6 overrides
    abstract public function calcFuelNeeds();

    6 references | 0 overrides
    protected function calcFuelEfficiency() {
        $range = 50000000;
        $range /= $this->load;
        return $range;
    }

    6 references | 0 overrides
    protected function calcTripDistance() {
        return 500;
    }
}
```

```php
<?php

6 references | 6 implementations
interface Flyer {
    5 references | 6 overrides
    public function takeOff();
    5 references | 6 overrides
    public function land();
    5 references | 6 overrides
    public function fly();
}

2 references | 2 implementations
interface Sailer {
    2 references | 2 overrides
    public function dock();
    2 references | 2 overrides
    public function cruise();
}
```

- **Output**

Praktikum 6

Nomor 4

Maksimal muatan Storeum 30000 kg	**Maksimal muatan Falcon 20000 kg**	**Maksimal muatan Hurricane 10000 kg**
Storeum menambah muatan sebesar 12000 kg	Falcon menambah muatan sebesar 12000 kg	Hurricane menambah muatan sebesar 8000 kg
Storeum menambah muatan sebesar 14000 kg	Falcon menambah muatan sebesar 8000 kg	Hurricane menambah muatan sebesar 2000 kg
Storeum menambah muatan sebesar 1000 kg	Falcon berada di dermaga	Hurricane lepas landas
Storeum menambah muatan sebesar 3000 kg	Falcon sedang berlayar	Hurricane dalam perjalanan
Storeum berada di dermaga	Falcon lepas landas	Hurricane mendarat
Storeum sedang berlayar	Falcon dalam perjalanan	Jadi, membutuhkan bahan bakar sebanyak 10 Liter
Jadi, membutuhkan bahan bakar sebanyak 4 Liter	Falcon mendarat	
	Jadi, membutuhkan bahan bakar sebanyak 5 Liter	

- **Analisis**

  Implementasi polymhorphism dengan interface dan abstract class ditunjukkan pada class SeaPlane yang implements interface Sailer, turunan dari class Airplane yang implements Flyer dan child dari Vehicle sehingga class SeaPlane memiliki method dock, cruise, takeoff, land, fly, dan calcFuelNeeds.