

Laporan Pengerjaan Tugas Teori Teknik Pemrograman Pertemuan 2

Oleh:

Nama : Muhammad Nabil Syauqi Rasyiq

NIM : 241524018

Kelas : 1A



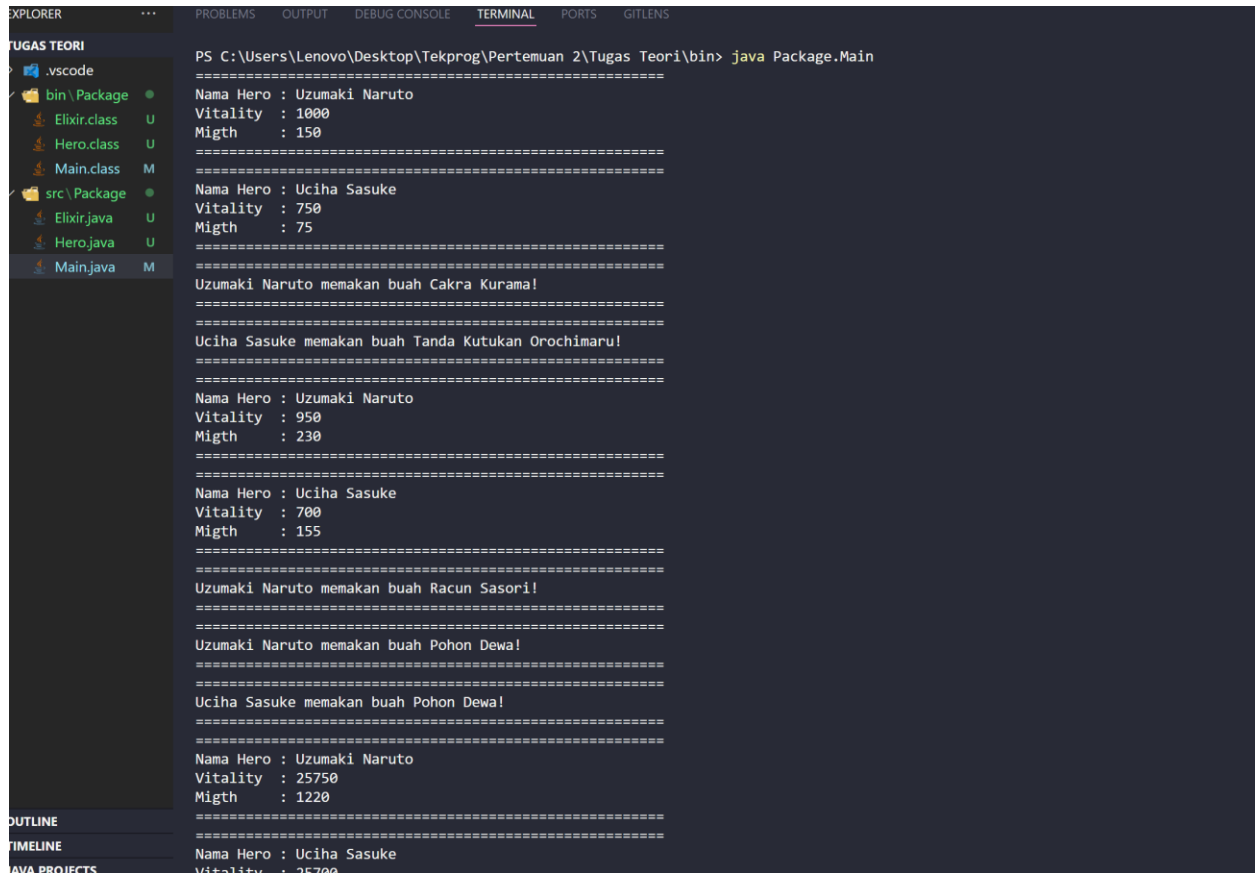
**Sarjana Terapan Program Studi Teknik Informatika
Jurusan Teknik Komputer dan Informatika
Politeknik Negeri Bandung
2025**

DAFTAR ISI

DAFTAR ISI	2
HASIL Pengerjaan	3
Hasil Akhir Program	3
Source code.....	3
Source code menggunakan gambar	6
Penjelasan	9
Kendala dan solusi	9
REFERENSI	11

HASIL Pengerjaan

Hasil Akhir Program



```
PS C:\Users\Lenovo\Desktop\Tekprog\Pertemuan 2\Tugas Teori\bin> java Package.Main
=====
Nama Hero : Uzumaki Naruto
Vitality : 1000
Migth : 150
=====
Nama Hero : Uchiha Sasuke
Vitality : 750
Migth : 75
=====
Uzumaki Naruto memakan buah Cakra Kurama!
=====
Uchiha Sasuke memakan buah Tanda Kutukan Orochimaru!
=====
Nama Hero : Uzumaki Naruto
Vitality : 950
Migth : 230
=====
Nama Hero : Uchiha Sasuke
Vitality : 700
Migth : 155
=====
Uzumaki Naruto memakan buah Racun Sasori!
=====
Uzumaki Naruto memakan buah Pohon Dewa!
=====
Uchiha Sasuke memakan buah Pohon Dewa!
=====
Nama Hero : Uzumaki Naruto
Vitality : 25750
Migth : 1220
=====
Nama Hero : Uchiha Sasuke
Vitality : 25700
```

Source code

Main.java	<pre>package <u>P</u>ackage; public class Main { public static void main(String[] args) { // Membuat Object Hero Hero naruto = new Hero("Uzumaki Naruto", 1000, 150); Hero sasuke = new Hero("Uchiha Sasuke", 750, 75); // Mendisplay informasi Hero naruto.display(); sasuke.display(); } }</pre>
-----------	--

	<pre> //Membuat object Elixir Elixir racunSasori = new Elixir("Racun Sasori", 0, 0, 10, 200); Elixir racunOrochimaru = new Elixir("Tanda Kutukan Orochimaru", 80, 250, 0, 300); Elixir cakraKurama = new Elixir("Cakra Kurama", 80, 250, 0, 300); Elixir buahCakra = new Elixir("Pohon Dewa", 1000, 25000, 0, 0); //Menggunakan Elixir naruto.useElixir(cakraKurama); sasuke.useElixir(racunOrochimaru); naruto.display(); sasuke.display(); naruto.useElixir(racunSasori); naruto.useElixir(buahCakra); sasuke.useElixir(buahCakra); naruto.display(); sasuke.display(); } } </pre>
Hero.java	<pre> package <u>P</u>ackage; public class Hero { private String heroName; private <i>int</i> vitality; private <i>int</i> might; public Hero(String <i>heroName</i>, <i>int</i> <i>vitality</i>, <i>int</i> <i>might</i>) { <i>this</i>.heroName = <i>heroName</i>; <i>this</i>.vitality = <i>vitality</i>; <i>this</i>.might = <i>might</i>; } public void addVitality(<i>int</i> <i>modifierV</i>){ <i>this</i>.vitality += <i>modifierV</i>; } } </pre>

	<pre> public void addMigth(int modifierM){ this.might += modifierM; } public void display(){ System.out.println("===== ====="); System.out.println("Nama Hero : " + this.heroName); System.out.println("Vitality : " + this.vitality); System.out.println("Migth : " + this.might); System.out.println("===== ====="); } public String getName(){ return this.heroName; } public void useElixir(Elixir elixir) { System.out.println("===== ====="); System.out.println(getName() + " memakan buah " + elixir.getElixirName()+ "!"); System.out.println("===== ====="); addMigth(elixir.getStrengthBoost() - elixir.getPowerDrain()); addVitality(elixir.getRejuvenation() - elixir.getCorrosion()); } } </pre>
Elixir.java	<pre> package <u>P</u>ackage; public class Elixir { private String elixirName; private int strengthBoost; private int rejuvenation; private int powerDrain; private int corrosion; </pre>

```

public Elixir(String elixirName, int strengthBoost, int rejuvenation, int
powerDrain, int corrosion) {
    this.elixirName = elixirName;
    this.strengthBoost = strengthBoost;
    this.rejuvenation = rejuvenation;
    this.powerDrain = powerDrain;
    this.corrosion = corrosion;
}

public String getElixirName() {
    return elixirName;
}

public int getStrengthBoost(){
    return this.strengthBoost;
}

public int getRejuvenation() {
    return this.rejuvenation;
}

public int getPowerDrain(){
    return this.powerDrain;
}

public int getCorrosion(){
    return this.corrosion;
}
}

```

Source code menggunakan gambar

```
1 package Package;
2
3 public class Hero {
4     private String heroName;
5     private int vitality;
6     private int might;
7
8     public Hero(String heroName, int vitality, int might) {
9         this.heroName = heroName;
10        this.vitality = vitality;
11        this.might = might;
12    }
13
14    public void addVitality(int modifierV){
15        this.vitality += modifierV;
16    }
17
18    public void addMigth(int modifierM){
19        this.might += modifierM;
20    }
21
22    public void display(){
23        System.out.println("=====");
24        System.out.println("Nama Hero : " + this.heroName);
25        System.out.println("Vitality : " + this.vitality);
26        System.out.println("Migth : " + this.might);
27        System.out.println("=====");
28    }
29
30    public String getName(){
31        return this.heroName;
32    }
33
34    public void useElixir(Elixir elixir) {
35        System.out.println("=====");
36        System.out.println(getName() + " memakan buah " + elixir.getElixirName()+ "!");
37        System.out.println("=====");
38        addMigth(elixir.getStrengthBoost() - elixir.getPowerDrain());
39        addVitality(elixir.getRejuvenation() - elixir.getCorrosion());
40    }
41 }
```



```
1 package _Package;
2
3 public class Main {
4
5     public static void main(String[] args) {
6
7         // Membuat Object Hero
8         Hero naruto = new Hero("Uzumaki Naruto", 1000, 150);
9         Hero sasuke = new Hero("Uchiha Sasuke", 750, 75);
10
11         // Mendisplay informasi Hero
12         naruto.display();
13         sasuke.display();
14
15         //Membuat object Elixir
16         Elixir racunSasori = new Elixir("Racun Sasori", 0, 0, 10, 200);
17         Elixir racunOrochimaru = new Elixir("Tanda Kutukan Orochimaru", 80, 250, 0, 300 );
18         Elixir cakraKurama = new Elixir("Cakra Kurama", 80, 250, 0, 300 );
19         Elixir buahCakra = new Elixir("Pohon Dewa", 1000, 25000, 0, 0 );
20
21         //Menggunakan Elixir
22         naruto.useElixir(cakraKurama);
23         sasuke.useElixir(racunOrochimaru);
24
25         naruto.display();
26         sasuke.display();
27
28         naruto.useElixir(racunSasori);
29         naruto.useElixir(buahCakra);
30         sasuke.useElixir(buahCakra);
31
32         naruto.display();
33         sasuke.display();
34
35     }
36 }
```



```

1  package _Package;
2
3  public class Elixir {
4      private String elixirName;
5      private int strengthBoost;
6      private int rejuvenation;
7      private int powerDrain;
8      private int corrosion;
9
10     public Elixir(String elixirName, int strengthBoost, int rejuvenation, int powerDrain, int corrosion) {
11         this.elixirName = elixirName;
12         this.strengthBoost = strengthBoost;
13         this.rejuvenation = rejuvenation;
14         this.powerDrain = powerDrain;
15         this.corrosion = corrosion;
16     }
17
18     public String getElixirName() {
19         return elixirName;
20     }
21
22     public int getStrengthBoost(){
23         return this.strengthBoost;
24     }
25
26
27     public int getRejuvenation() {
28         return this.rejuvenation;
29     }
30
31     public int getPowerDrain(){
32         return this.powerDrain;
33     }
34
35     public int getCorrosion(){
36         return this.corrosion;
37     }
38 }
39

```

Penjelasan

Konsep : Program yang dibuat merupakan contoh pengimplementasian Interaksi kelas pada java, program ini menggunakan tema Hero, dimana Object dari class hero dapat berinteraksi sementara dengan object dari class Elixir.

Kendala dan solusi

Kendala yang dialami oleh saya adalah kurangnya pemahaman saya terkait interaksi dependences dan bagaimana contoh pengaplikasiannya, untuk menyelesaikan kendala ini saya mencari tahu lebih jauh terkait interaksi dependences ini, juga melihat penerapan encapsulation dari youtube : https://youtu.be/gI9dI0VG9YU?list=PLZS-MHyEIRo6V4_vk1s1NcM2HoW5KFG7i . walau pada contoh penerapan encapsulation tersebut

menggunakan interaksi Has-a. tetapi saya berhasil mengembangkan dengan ide yang sedikit berbeda menjadi interaksi use-a.

Link Github : <https://github.com/Rasyiq603011/Tekprog-teori-W2>

REFERENSI

https://youtu.be/gI9dI0VG9YU?list=PLZS-MHyEIRo6V4_vk1s1NcM2HoW5KFG7i

<https://www.techtarget.com/searchapparchitecture/definition/dependency-injection>