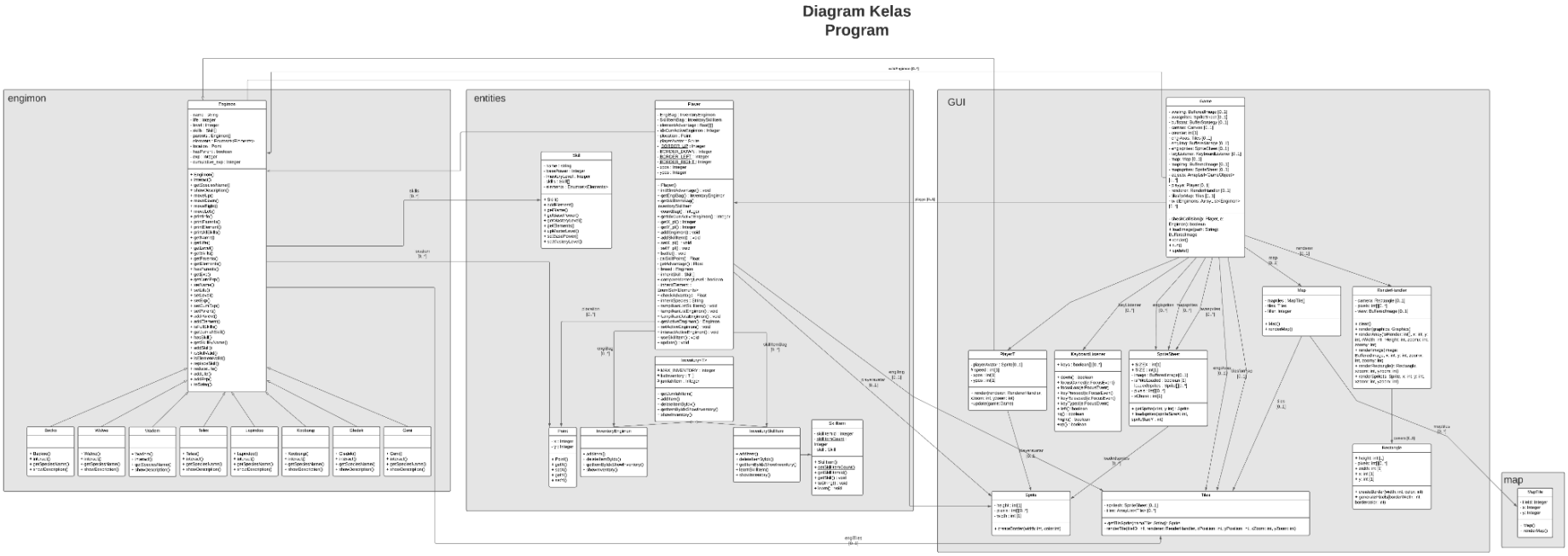
Kelas : 03
Nama Kelompok : mbohwes
1. 13519112 / Pratama Andiko
2. 13519138 / Bintang Fajarianto
3. 13519140 / Fabian Savero Diaz Pranoto
4. 13519146 / Fadel Ananda Dotty
5. 13519157 / Ryandito Diandaru
6. 13519165 / Kadek Surya Mahardika
Asisten Pembimbing : Jan Meyer Saragih

1. Diagram Kelas



File gambar diagram kelas dilampirkan bersama file laporan jika gambar diatas kurang jelas.

Dalam mendesain program pada tugas besar kali ini terdapat 7 kelas utama(Engimon, Inventory, Skill, Skill Item, Player, Point, ReadWriteGame). Kelas Engimon akan menginherit menjadi beberapa species Engimon(Becko, Geni, Gledek, Koobong, Lapindoo, Teles, Wadem, Watoo). Kelas Generic Inventory akan menginherit kelas InventoryEngimon dan InventorySkillItem.

2. Penerapan Konsep OOP

2.1. Polymorphism

Konsep polymorphism pada beberapa fungsi di dalam program. Salah satu fungsi yang memanfaatkan konsep polymorphism adalah fungsi breed. Fungsi tersebut memiliki dua parameter berupa tipe pointer Engimon. Meskipun memiliki parameter engimon, objek yang merupakan kelas turunan dari kelas engimon juga dapat dimasukkan sebagai parameter pada fungsi breed. Keuntungan dari penggunaan polymorphism adalah mengurangi coupling, meningkatkan *readability* kode yang telah dibuat, dan menggunakan kembali fungsi yang sama untuk tipe data yang berbeda. Cuplikan kode breed yang memanfaatkan *polymorphism* sebagai berikut:

```
public Engimon breed(Engimon dad, Engimon mom){
        try{
            … some code

            } else {
                throw new Exception("Level Orang Tua Kurang Tinggi!");
            }
        } catch (Exception e){
            e.printStackTrace();
        }
```

2.2. Inheritance/Composition/Aggregation

Pada program diterapkan konsep Inheritance dengan dibuatnya *derived class* dari *parent class* Engimon. Dibuatkan 8 kelas turunan dari kelas Engimon yaitu kelas Beckoo, Geni, Gledek, Koobong, Lapindoo, Teles, Wadem, dan Watwo. Keuntungan dibuat kelas turunan adalah mempermudah membuat objek engimon berdasarkan spesiesnya, memanfaatkan penggunaan kembali kode kelas *parent*, dan meningkatkan *readability* kode, selain itu juga memudahkan perubahan pada spesies-spesies yang mungkin akan diberi atribut/method unik

jika memang di kemudian hari akan ditambahkan. Di bawah merupakan cuplikan kelas Becko yang merupakan kelas turunan dari kelas Engimon.

```
public class Beckoo extends Engimon {
    public Beckoo() {
        this.setName("Wild Beckoo");
        this.addElement(Elements.ICE);
        this.addElement(Elements.WATER);
        this.addSkill(new Skill("Beku Dingin", 100, 1, EnumSet.of(Elements.ICE, Elements.WATER)));
```

2.3. Abstract Class

Kelas Inventory menggunakan abstrak kelas karena memiliki method yang implementasinya berbeda untuk tiap subclassnya(antara Inventory<Engimon> dan Inventory<SkillItem>)

```
public abstract class  Inventory<T> {
    protected static final int MAX_INVENTORY = 10;
    protected List<T> listInventory = new ArrayList<>();
    protected static int jumlahItem = 0;

public abstract class Engimon implements MoveAction {
    private String name;
    private int life;
    private int level;
    private Skill[] skills;
    private Engimon[] parents;
    private EnumSet<Elements> elements;
    private Point location;
    private boolean hasParent;
    private int exp;
    private int cumulative_exp;
```

2.4. Interface

```
public interface GameObject {
    public void render(RenderHandler renderer, int xzoom, int yzoom);
    public void update(Game game);
}

public interface IState {
    public void update();
    public void render();
    public void enter();
    public void exit();
}
```

2.5. Generic Type & Wildcards

Konsep kelas template dan generik diterapkan pada implementasi kelas inventory. Kelas inventory akan dimanfaatkan untuk menyimpan objek engimon dan skill item. Konsep ini cocok diterapkan untuk kelas inventory karena meskipun memiliki inventory akan berisi dua objek yang berbeda, namun aksi-aksi yang dilakukan terhadap kedua inventory tetap sama sehingga lebih baik memakai kelas generik.

```
public abstract class  Inventory<T> {
    protected static final int MAX_INVENTORY = 10;
    protected List<T> listInventory = new ArrayList<>();
    protected static int jumlahItem = 0;

    public static int getJumlahItem()
    {
        return jumlahItem;
    }
    //abstract method
```

```
    public abstract void addItem(T item , int n);
    public abstract void deleteItemByIdx(int index, int n);
    public abstract T getItemByIdxShowInventory(int index);
    public abstract void showInventory();
    }
```

2.6. Exception Handling

Untuk menangani masalah yang mungkin terjadi, beberapa fungsi jika memenuhi syarat yang akan menyebabkan error akan melempar sebuah exception. Fungsi tersebut lalu akan dipanggil dengan menggunakan blok try-catch untuk dilakukannya *exception handling*. Salah satu fungsi yang melempar exception adalah fungsi breed yang akan melempar sebuah exception ketika kedua parent tidak melebihi level 30. Cupikan fungsi breed sebagai berikut:

```
public Engimon breed(Engimon dad, Engimon mom){
        try{
            … some code

            } else {
                throw new Exception("Level Orang Tua Kurang Tinggi!");
            }
        } catch (Exception e){
            e.printStackTrace();
        }
```

2.7. Java Collection

Collection digunakan untuk implementasi Inventory<Engimon> dan Inventory<SkillItem>. Pada Inventory<Engimon> akan dilakukan pengelompokan engimon berdasarkan spesies, lalu akan diurutkan berdasarkan level tertinggi terlebih dahulu. Pada Inventory<SkillItem> akan dilakukan pengurutan dari Base Power yang terbesar terlebih dahulu.

```
Collections.sort(listInventory, Comparator.comparing(Engimon::getSpeciesName)
                    .thenComparing((Engimon e1, Engimon e2) -> e2.getLevel()-e1.getLevel()));

Collections.sort(listInventory, (SkillItem e1, SkillItem e2) -> e2.getSkill().getBasePower()-e1.getSkill().getBasePower());
```

2.8. Java API

API yang digunakan dalam pengerjaan tugas besar ini adalah Java 2D API dan file API. Java 2D API berfungsi untuk menampilkan program dalam bentuk grafik 2 dimensi sedangkan Java File API digunakan untuk mengatur file-file eksternal yang akan digunakan dalam program.

```
import java.awt.*;
import java.awt.image.BufferStrategy;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
import java.lang.Runnable;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
```

3. Bonus Yang dikerjakan

3.1. Bonus yang diusulkan oleh spek

3.1.1. Multi-threading (real-time gameplay)

3.1.2. Dual Element Breeding

3.1.3. Purely Random Wild Engimon Generation

3.1.4. Unit Testing Implementation

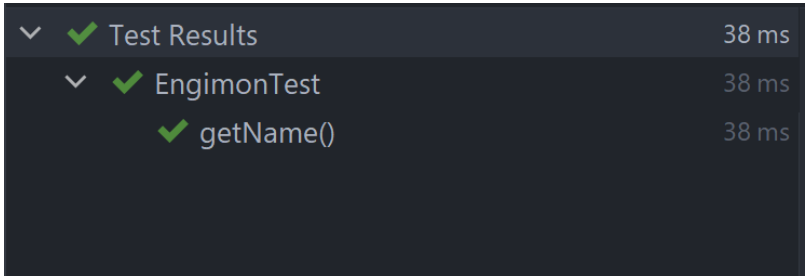
Library yang digunakan untuk melakukan *testing* pada kelas yang dibuat adalah library JUnit5. Tes yang dilakukan pada kelas yang dibuat meliputi tes terhadap method-method seperti *getter* dan *setter* pada kelas tersebut. Unit tes dibuat pada kelas engimon dan *subclassnya*, kelas inventory engimon, kelas player, dan kelas skill.

Unit testing class Engimon

Contoh Method Testing:

```
@Test
    void getName() {
        Beckoo test = new Beckoo("Beckoo", 1, 1);
        assertEquals("Beckoo", test.getName());

        Geni test1 = new Geni("Geni", 1, 1);
        assertEquals("Geni", test1.getName());

        Gledek test2 = new Gledek("Gledek", 1, 1);
        assertEquals("Gledek", test2.getName());

        Koobong test3 = new Koobong("Koobong", 1, 1);
        assertEquals("Koobong", test3.getName());

        Lapindoo test4 = new Lapindoo("Lapindoo", 1, 1);
        assertEquals("Lapindoo", test4.getName());

        Teles test5 = new Teles("Teles", 1, 1);
        assertEquals("Teles", test5.getName());

        Wadem test6 = new Wadem("Wadem", 1, 1);
        assertEquals("Wadem", test6.getName());

        Watoo test7 = new Watoo("Watoo", 1, 1);
        assertEquals("Watoo", test7.getName());
    }
```

Hasil Testing:

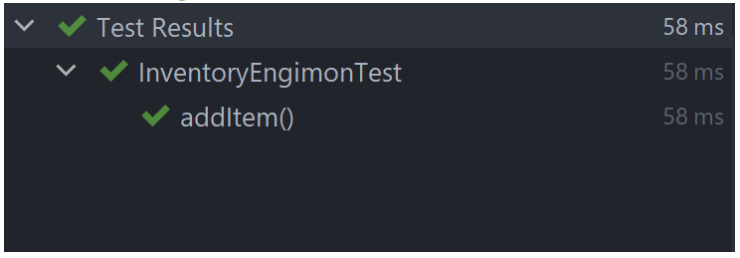


Unit testing class Inventory

Contoh Method Testing:

```
@Test
    void addItem() {
        Beckoo engimon1 = new Beckoo("Beckoo", 1, 1);
        Geni engimon2 = new Geni("Geni", 1, 1);
        Inventory inven = new InventoryEngimon();
        inven.addItem(engimon1, 1);
        assertEquals(1, inven.getJumlahItem());
        inven.addItem(engimon2, 1);
        assertEquals(2, inven.getJumlahItem());
    }
```

Hasil Testing:

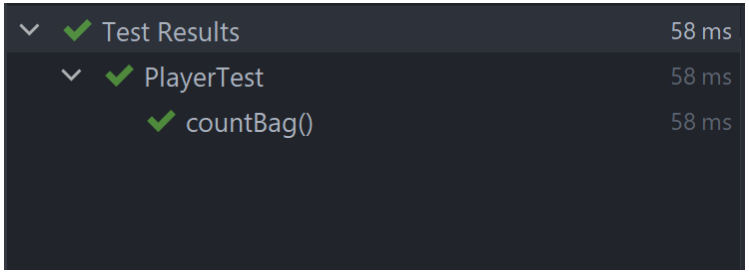


Unit testing class Player

Contoh Method Testing:

```
@Test
    void countBag() {
        Player p = new Player();
        Wadem w = new Wadem("Tes", 1, 1);
        p.addEngimon(w);
        Skill s = new Skill("duarr", 10, 5, EnumSet.of(Elements.ELECTRIC));
        SkillItem si = new SkillItem(s);
        p.addSkillItem(si, 1);
        assertEquals(2, p.countBag());
    }
```

Hasil Testing:

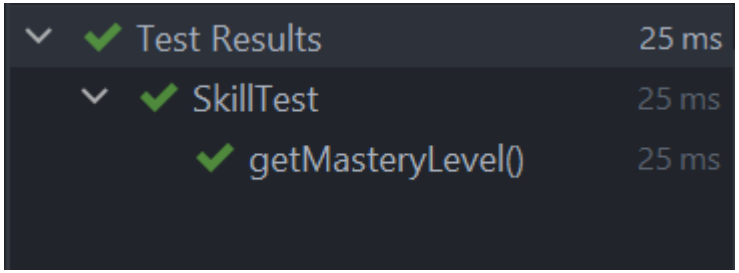


Unit testing class Skill

Contoh Method Testing:

```
@Test
    void getMasteryLevel() {
        Skill tes = new Skill("tes", 10, 1, EnumSet.of(Elements.FIRE));
        assertEquals(1, tes.getMasteryLevel());
    }
```

Hasil Testing:



3.2. Bonus Kreasi Mandiri

3.2.1. Realtime Movement

Game yang kami buat mengimpor KeyListener dan KeyEvent. Oleh karena itu game yang dibuat memungkinkan pergerakan dilakukan secara *realtime*.

3.2.2. Smooth Movement

Pergerakan pada game yang kami buat berjalan pada *frame rate* 60 Hz. Oleh karena itu pergerakan ini merupakan pergerakan yang *smooth*.

4. External Library

Untuk implementasi GUI digunakan library AWT

```
import java.awt.*;
```

Untuk implementasi unit testing digunakan library JUnit5

```
import org.junit.jupiter.api.Test;
```

Untuk implementasi load dan save file digunakan library Jackson

```
import com.fasterxml.jackson.annotation.JsonAutoDetect;
import com.fasterxml.jackson.annotation.PropertyAccessor;
import com.fasterxml.jackson.databind.DeserializationFeature;
import com.fasterxml.jackson.databind.ObjectMapper;
```

5. Pembagian Tugas

Modul (dalam poin spek)	Designer	Implementer
1. Engimon	13519140	13519140, 13519157,13519112
2. Skill	13519140	13519140,13519112
3. Skill Item	13519112	13519112
4.Inventory	13519112	13519112
5. Battle	13519165	13519140, 13519165
6. Breeding	13519138	13519138, 13519140, 13519165
7. Peta + GUI	13519140, 13519157,13519146	13519140,13519146, 13519157, 13519165

8. Player	13519138	13519138, 13519140, 13519157, 13519165
[BONUS] Multi Threading	13519146	13519157
[BONUS] Unit Testing Implementation	13519146	13519146
[BONUS] Purely Random Wild Engimon Generation	13519146	13519146
[BONUS] Dual element breeding	13519138	13519138
Laporan	13519112, 13519146	13519112, 13519146
Integrasi	13519140, 13519146, 13519157	13519140, 13519146, 13519157