LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
JAVA INHERITANCE (PEWARISAN) II
MODUL VIII

Disusun Oleh :

Sofia Melati Bareut Runa - 105222005

PROGRAM STUDI ILMU KOMPUTER
FAKULTAS SAINS DAN KOMPUTER
UNIVERSITAS PERTAMINA
2025

I. Pendahuluan

Dalam pembuatan permainan interaktif berbasis teks, merancang game petualangan dengan fitur login dan inventori sering kali menemui berbagai tantangan yang cukup signifikan bagi para pengembang yang baru memulai. Pengelolaan karakter pemain, pertarungan, dan penyimpanan barang secara manual dapat mengakibatkan kerumitan kode yang signifikan, serta meningkatkan kemungkinan munculnya kesalahan atau inkonsistensi dalam logika permainan. Untuk mengatasi masalah tersebut, program ini dibuat berdasarkan studi kasus yang diberikan sebagai Ujian Rumah dalam mata kuliah Pemrograman Berorientasi Objek (PBO). Program ini dibuat untuk secara otomatis mengatur informasi tentang pemain dan perjalanan mereka, yang meliputi fitur login, pertarungan, pemulihan, serta sistem inventaris yang terstruktur.

Program ini dilengkapi dengan fitur yang memungkinkan pemantauan informasi akun pengguna, meliputi nama pengguna, kata sandi, poin kesehatan (HP), serta barang yang telah dikumpulkan seperti gaun dan jenis kain. Setelah proses pendaftaran dan masuk yang berhasil, sistem akan memberikan kesempatan kepada pemain untuk menjelajahi dunia permainan dan berinteraksi dengan berbagai fitur sesuai dengan aturan yang telah ditetapkan, yaitu:

1. Peserta diizinkan untuk bertanding hanya satu kali sebelum harus melakukan jeda untuk memulihkan diri.
2. Setiap pertarungan akan mengurangi HP pemain dan memberikan hadiah berupa kostum jika berhasil dimenangkan.
3. Istirahat akan memulihkan sebagian HP dan memberikan kain secara acak.

Fitur terakhir dari program ini adalah penyajian informasi inventaris dalam format yang teratur dan sistematis, yang mencakup data mengenai jumlah gaun yang telah dikumpulkan, bahan yang tersedia, serta keadaan kesehatan para pemain.

II. Variabel

No	Nama Variabel	Tipe data	Fungsi
1.	*username*	String	Untuk menyimpan nama pengguna
2.	*password*	String	Untuk menyimpan kata sandi pengguna
3.	*health*	Integer	Untuk menyimpan HP (nyawa) *player*
4.	*dresses*	String[]	Untuk menyimpan dress yang diperoleh dalam bentuk array
5.	*materials*	String[]	Untuk menyimpan jenis kain/material dalam bentuk array
6.	*dressCount*	Integer	Untuk menyimpan jumlah dress yang telah dikumpulkan
7.	*materialCount*	Integer	Untuk menyimpan jumlah bahan/material yang ditemukan
8.	*justFought*	Boolean	Untuk menunjukkan apakahh pemain baru saja bertarung (true) atau tidak (false)
9.	*input*	Scanner	Untuk menerima input pengguna
	savedPlayer	Player	Untuk menyimpan objek *player* yang telah teregistrasi dan digunakan saat *login*
	loggedIn	Boolean	Untuk menunjukkan status login pengguna
	Pilihan	String	Unruk menyimpan input pilihan *user* pada menu digame.
	uN, pS	String	Input dari pengguna saat mencoba *login* (username dan password)
	aksi	String	Input dari pilihan aksi petualangan (fight, heal, inventory, exit)
	user, pass	String	Input dari pengguna saat registrasi
	dress	String	Untuk menyimpan hasil random nama dress setelah bertarung

	material	String	Untuk menyimpan hasil random jenis bahan yang didapatkan

III. Constructor and Method

No	Nama Method	Jenis Method	Fungsi
1.	***User***	**Constructor**	**Untuk menginisialisasi objek User dengan atributnya**
2.	***Player***	**Constructor**	**Untuk menginisialisasi objek Player dengan memanggil konstruktor User dengan super**
3.	*Login()*	Functional	Untuk mengecek apakah username dan password cocok untuk menampilkan hasil login.
4.	*healPlayer()*	Procedural	Untuk menambahhh HP pemain hingga maksimal, dan reset status *justFought*
5.	*fightMonster()*	Procedural	Untuk membuat *player* melawan monster yang akan mengurangi HP *player* dengan bayaran mendapatkan dress jika menang atau game over jika kalah
6.	*findMaterial(String)*	Procedural	Untuk memberikan bahan acak yang didapatkan dari eksploaris untuk disimpan.
7.	*showInventory(String)*	Procedural	Untuk menampilkan isi inventori
19.	***main(String[])***	**Main Method**	**Menjalankan simulasi perpustakaan dengan menambahkaan buku, meminjam, dan mengembalikan buku.**

IV. Dokumentasi dan Pembahasan Code

Berikut ini ditampilkan dokumentasi dan penjelasan mengenai program yang telah dikembangkan untuk system OOP untuk Sistem Game Open World (Gathdres Adventure).

```java
import java.util.Scanner;
import java.util.Random;

class User {
    protected String username;
    protected String password;

    public User(String username, String password) {
        this.username = username;
        this.password = password;
    }

    public void login (String uN, String pS) {
        if (this.username.equals(uN) && this.password.equals(pS)) {
            System.out.println(x:"Login berhasil.");
        } else {
            System.out.println(x:"Login gagal.");
        }
    }
}

class Player extends User {
    public int health = 100;
    String[] dresses = new String[10];
    String[] materials = new String[10];
    public int dressCount = 0;
    public int materialCount = 0;
    public boolean justFought = false;

    public Player(String username, String password) {
        super(username, password);
    }
```

```java
    public void healPlayer() {
        health = Math.min(a:100, health + 20);
        System.out.println("Kamu beristirahat dan memulihkan diri. HP : " + health);
        justFought = false;
    }

    public void fightMonster() {
        System.out.println(x:"Monster muncul! Kamu bertarung...");
        health -= 30;
        if (health <= 0) {
            System.out.println(x:"Kamu kalah... Game over.");
            System.exit(status:0);
        } else {
            System.out.println("Kamu berhasil mengalahkan monster! HP tersisa: " + health);
            String dress = "Dress Lv." + (new Random().nextInt(bound:3) + 1);
            dresses[dressCount++] = dress;
            System.out.println("Kamu menang dan mendapatkan: " + dress);
            justFought = true;
        }
    }

    public void findMaterial() {
        String material = "Kain " + (new Random().nextBoolean() ? "Sutra" : "Satin");
        materials[materialCount++] = material;
        System.out.println("Dress yang ditemukan memiliki bahan : " + material);
    }
```

```
61      public void showInventory() {
62          System.out.println("HP    : " + health);
63          System.out.print(s:"Dress: ");
64
65          for (int i = 0; i < dressCount; i++) {
66              System.out.print(dresses[i] + " ");
67          } System.out.println();
68          System.out.print("Jumlah Bahan: " + materialCount + "\n");
69
70          for (int i = 0; i < materialCount; i++) {
71              System.out.print(materials[i] + " ");
72          } System.out.println(x:"\n");
73      }
74  }
```

Gambar 4.1 Full Code Class User & Player

```
75
76  public class DressUP {
        Run | Debug
77      public static void main(String[] args) {
78          Scanner input = new Scanner(System.in);
79          Player savedPlayer = null;
80          boolean loggedIn = false;
81
82          System.out.println(x:"=== GATHDRE GAME ===");
83
84          //LOGIN DAN REGISTER
85          while (savedPlayer == null || !loggedIn) {
86              System.out.println(x:"\n=== MAIN MENU ===");
87              System.out.println(x:"1. Register");
88              System.out.println(x:"2. Login");
89              System.out.print(s:"Pilih: ");
90              String pilihan = input.nextLine();
91
92              if (pilihan.equals(anObject:"1")) {
93                  System.out.print(s:"Masukkan username: ");
94                  String user = input.nextLine();
95                  System.out.print(s:"Masukkan password: ");
96                  String pass = input.nextLine();
97                  savedPlayer = new Player(user, pass);
98                  System.out.println(x:"Registrasi berhasil. Silakan login.");
99                  System.out.println();
```

```
101             } else if (pilihan.equals(anObject:"2")) {
102                 if (savedPlayer == null) {
103                     System.out.println(x:"Belum ada akun terdaftar.");
104                     System.out.println(x:"Silakan registrasi terlebih dahulu.");
105                     System.out.println();
106                 }
107
108                 System.out.print(s:"Username: ");
109                 String uN = input.nextLine();
110                 System.out.print(s:"Password: ");
111                 String pS = input.nextLine();
112
113                 if (savedPlayer.username.equals(uN) && savedPlayer.password.equals(pS)) {
114                     savedPlayer.login(uN, pS);
115                     loggedIn = true;
116                     System.out.println("Selamat datang, " + uN);
117                 } else {
118                     System.out.println(x:"Login gagal. Silakan coba lagi.");
119                 }
120             }
121         }
```

```
123         //ADVENTURE
124         while (true) {
125             System.out.println(x:"\n=== Petualangan Dimulai ===");
126             System.out.println(x:"1. Maju (Lawan monster)");
127             System.out.println(x:"2. Mundur (Istirahat & cari bahan)");
128             System.out.println(x:"3. Lihat Inventori");
129             System.out.println(x:"4. Keluar");
130             System.out.print(s:"Pilih: ");
131             String aksi = input.nextLine();
132 
133             if (aksi.equals(anObject:"1")) {
134                 if (!savedPlayer.justFought) {
135                     savedPlayer.fightMonster();
136                 } else {
137                     System.out.println (x:"Kamu baru saja bertarung, istirahat dulu sebelum bertarung lagi.");
138                 }
139             } else if (aksi.equals(anObject:"2")) {
140                 savedPlayer.healPlayer();
141                 savedPlayer.findMaterial();
142             } else if (aksi.equals(anObject:"3")) {
143                 savedPlayer.showInventory();
144             } else if (aksi.equals(anObject:"4")) {
145                 System.out.println(x:"Terima kasih telah bermain!");
146                 break;
147             } else {
148                 System.out.println(x:"Pilihan tidak dikenali.");
149             }
150         } input.close();
151     }
152 }
```

Gambar 4.2 Full Code Class Main

Untuk pemahaman lebih lanjut, *code* ini akan dijelaskan per baris sebagai berikut.

CLASS USER (PARENT CLASS/SUPER CLASS)

```
1  import java.util.Scanner;
2  import java.util.Random;
```

Baris 1. . Kode ini digunakan untuk mengimpor *library* eksternal yaitu *Scanner* untuk digunakan sebagai pembaca input dari pengguna.

Baris 2. Kode ini digunakan untuk mengimpor *library* eksternal yaitu *Random* untuk digunakan sebagai memilih suatu objek/angka secara acak.

```
4  class User {
5      protected String username;
6      protected String password;
```

- **Baris 4.** Kode ini mendeklarasikan kelas dengan nama **User** yang sama dengan nama file java dengan deklarasi dalam *default*.
- **Baris 5, & 6.** Kode ini menyimpan attribute dimana mendeskripsikan data atau informasi dari objek pada ***class user***.
- **Baris 5.** Variabel yang menyimpan **username** dengan tipe data String.
- **Baris 6.** Variabel yang menyimpan **password** dengan tipe data String.
- **INFO :** Kedua kode menggunakan akses *protected* yang dimana menyatakan bahwa kelas ini hanya bisa diakses oleh subclass yang *extends* dengan *superclass* ini.

```
8      public User(String username, String password) {
9          this.username = username;
10         this.password = password;
11     }
```

- **Baris 8, 9, & 10.** Kode ini menyimpan konstruktor dimana metode yang akan digunakan untuk membuat objek baru dengan menginisialisasi variable pada baris kode yang diatas yaitu **Baris 5, & 6.**
- **Baris 8.** Kode ini menerima 2 parameter yaitu **username**, dan **password** dari variable pada baris kode sebelumnya.
- **Baris 9.** Kode ini menginisialisasi atribut **username** dengan nilai parameter dengan nilai parameter **username** menggunakan **this.username.**
- **Baris 10.** Kode ini menginisialisasi atribut **password** dengan nilai parameter dengan nilai parameter **password** menggunakan **this.password.**

```
13      public void login (String uN, String pS) {
14          if (this.username.equals(uN) && this.password.equals(pS)) {
15              System.out.println(x:"Login berhasil.");
16          } else {
17              System.out.println(x:"Login gagal.");
18          }
19      }
20  }
21
```

Baris 13 – 17. Kode program ini merupakan metode milik *class user*. Dimana pada **Baris 14** memeriksa apakah input login (username, password) cocok dengan data yang tersimpan didalam objek. Jika cocok maka hasilnya ada pada **Baris 15** dan jika tidak sesuai maka hasil nya ada pada **Baris 17**.

CLASS PLAYER (CHILD CLASS/SUBCLASS)

```
22  class Player extends User {
23      public int health = 100;
24      String[] dresses = new String[10];
25      String[] materials = new String[10];
26      public int dressCount = 0;
27      public int materialCount = 0;
28      public boolean justFought = false;
29
```

- **Baris 22.** Kode ini mendeklarasikan kelas dengan nama **Player** yang sama dengan nama file java dengan deklarasi dalam *default*. *Class* ini mewarisi *class user* sehingga **bisa mengakses username, password, dan metode login().**
- **Baris 23 – 28.** Kode ini menyimpan attribute dimana mendeskripsikan data atau informasi dari objek pada ***class Player***.
- **Baris 23.** Variabel yang menyimpan **health** dengan tipe data Integer dimana memiliki nilai awal yaitu 100.
- **Baris 24.** Variabel ini merupakan array yang menyimpan maksimal 10 elemen **dress (hasil bertarung)** dengan tipe data String.
- **Baris 25.** Variabel ini merupakan array yang menyimpan maksimal 10 elemen **bahan (kain)** dengan tipe data String.
- **Baris 26 & Baris 27.** Variable ini menyimpan jumlah item yang telah dimiliki karena array tidak bisa mengecek jumlah item secara otomatis.
- **Baris 28.** Variabel ini sebagai pembatas dari program ini agar **pemain yang barus saja bertarung tidak bertarung sebelum istirahat, sehingga mencegah adanya spam fight.**

```
30      public Player(String username, String password) {
31          super(username, password);
32      }
```

- **Baris 8, 9, & 10.** Kode ini menyimpan konstruktor dimana metode yang akan digunakan untuk membuat objek baru dengan menginisialisasi variable pada *parent class.*
- **Baris 8.** Kode ini melakukan pewarisan constructor dengan menggunakan kode *super (…)* yang akan memanggil *constructor* milik *parent class (user).*

```
34      public void healPlayer() {
35          health = Math.min(a:100, health + 20);
36          System.out.println("Kamu beristirahat dan memulihkan diri. HP : " + health);
37          justFought = false;
38      }
```

- **Baris 34 – 37.** Kode program ini merupakan metode milik *class player*.
- **Baris 35.** Metode ini melakukan penghitungan menggunakan objek Math.min dikarenakan maksimum health adalah 100. Metode ini dipanggil **saat *player* baru selesai bertarung dengan monster dengan begitu player akan mendapat penyembuhan pada health sebesar 20.**
- **Baris 36 & 37.** Metode ini **menampilkan HP** setelah di sembuhkan dan melakukan **reset pada justFought** agar player bisa bertarung lagi.

```
40      public void fightMonster() {
41          System.out.println(x:"Monster muncul! Kamu bertarung...");
42          health -= 30;
43          if (health <= 0) {
44              System.out.println(x:"Kamu kalah... Game over.");
45              System.exit(status:0);
46          } else {
47              System.out.println("Kamu berhasil mengalahkan monster! HP tersisa: " + health);
48              String dress = "Dress Lv." + (new Random().nextInt(bound:3) + 1);
49              dresses[dressCount++] = dress;
50              System.out.println("Kamu menang dan mendapatkan: " + dress);
51              justFought = true;
52          }
53      }
```

- **Baris 40 – 51.** Kode program ini merupakan metode milik *class player*.
- **Baris 41 & 42.** Metode ini menampilkan **munculnya monster** dan ***player* bertarung dengan monster** itu dengan akibat setelahnya **health sebelumnya dikurangi dengan 30 (berkurang 30).**
- **Baris 43 – 45.** Metode ini melakukan perhitungan kemungkinan jika *player* melawan monster sampai **health yang dimiliki 0** maka *player* **dinyatakan kalah** dan **game langsung berakhir**.

- **Baris 47.** Metode ini melakukan perhitungan kemungkinan jika *player* melawan monster tapi selamat maka health *player* sekarang akan ditampilkan sebelum dipanggilnya method **healPlayer().**
- **Baris 48, 49, & 50.** Pada **Baris 48,** metode ini melakukan pembuatan dress acak dari level 1 hingga 3 sebagai hadiah melawan dan mengalahkan monster dengan perhitungan **nextInt(3)** yang menghasilkan 0-2 yang akan ditambah 1 menjadi 1-3. Lalu pada **Baris 49, & Baris 50** dress langsung akan disimpan ke inventori milik *player* dengan tingkatannya..
- **Baris 51.** Metode ini **menampilkan justFought** yang bertanda bahwa *player* baru saja selesai bertarung.

```
54
55      public void findMaterial() {
56          String material = "Kain " + (new Random().nextBoolean() ? "Sutra" : "Satin");
57          materials[materialCount++] = material;
58          System.out.println("Dress yang ditemukan memiliki bahan : " + material);
59      }
```

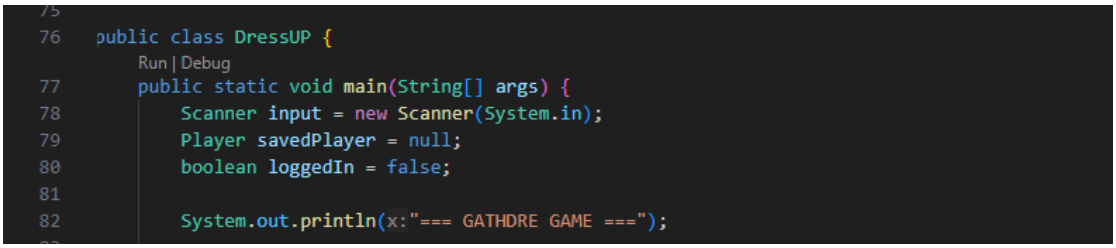
- **Baris 55 – 58.** Kode program ini merupakan metode milik *class player*.
- **Baris 56, 57, & 58.** Pada **Baris 56,** metode ini membuat material secara acak yaitu ”Kain Sutra” dan “Kain Satin”. Lalu pada **Baris 49, & Baris 50** dress langsung akan disimpan ke inventori milik *player* dengan tingkatannya lalu ditampilkan kepada *player* material apa yang didapatkan.

```
61      public void showInventory() {
62          System.out.println("HP   : " + health);
63          System.out.print(s:"Dress: ");
64
65          for (int i = 0; i < dressCount; i++) {
66              System.out.print(dresses[i] + " ");
67          } System.out.println();
68          System.out.print("Jumlah Bahan: " + materialCount + "\n");
69
70          for (int i = 0; i < materialCount; i++) {
71              System.out.print(materials[i] + " ");
72          } System.out.println(x:"\n");
73      }
74  }
```

- **Baris 61 – 72.** Kode program ini merupakan metode milik *class player*.
- **Baris 41 & 42.** Metode ini menampilkan **health** dan **dress** yang tersimpan didalam inventori.
- **Baris 65 – 68.** Metode ini melakukan perhitungan perulangan dalam menampilkan dress secara keseluruhan satu per satu dengan kondisi awal atau situasi awal dengan jumlah nol (0).

- **Baris 69 – 72.** Metode ini melakukan perhitungan perulangan dalam menampilkan material kain secara keseluruhan satu per satu dengan kondisi awal atau situasi awal dengan jumlah nol (0).

CLASS MAIN



```
76  public class DressUP {
        Run | Debug
77      public static void main(String[] args) {
78          Scanner input = new Scanner(System.in);
79          Player savedPlayer = null;
80          boolean loggedIn = false;
81
82          System.out.println(x:"=== GATHDRE GAME ===");
```

- **Baris 76.** Kode ini mendeklarasikan kelas dengan nama **Main** yang sama dengan nama file java. Terlihat kode dideklarasikan di *Public* sehingga bisa diakses kapanpun dan dimanapun, dimana bertindak sebagai *blueprint* untuk objek produk yang memiliki atribut dan metode terkait.
- **Baris 77.** Kode ini merupakan metode class dimana berperan sebagai metode utama dalam menjalankan atau mengeksekusi program.
- **Baris 78.** Kode ini membuat objek ‘input’ dari kelas Scanner dengan menghubungkan program dan inputan pengguna.
- **Baris 79.** Kode ini membuat dan menyimpan objek setelah dilakukannya pendaftaran.
- **Baris 80.** Kode ini menggunakan Boolean untuk menentukan status apakah *player* berhasil melakukan aktivitas *login* atau belum.
- **Baris 82.** Kode menampilkan nama Game.

```
84          //LOGIN DAN REGISTER
85          while (savedPlayer == null || !loggedIn) {
86              System.out.println(x:"\n=== MAIN MENU ===");
87              System.out.println(x:"1. Register");
88              System.out.println(x:"2. Login");
89              System.out.print(s:"Pilih: ");
90              String pilihan = input.nextLine();
```

- **Baris 85.** Kode ini mendeklarasikan dan melakukan *loop* (perulangan) sampai ada akun yang melakukan registrasi dan berhasil login ke game.
- **Baris 86, 87, 88, & 89.** Kode ini **menampilkan menu utama game** sebelum masuk ke dalam game dengan beberapa opsi yang ada. Dengan syarat, *player* harus sudah teregistrasi untuk bisa memainkan game.

- **Baris 90.** Kode ini mendeklarasikan pilihan akan langsung masuk dan dieksekusi setelah diinput sesuai urutan FIFO (First in First out) jadi memilih pilihan harus satu-satu secara berurut..

```
91
92          if (pilihan.equals(anObject:"1")) {
93              System.out.print(s:"Masukkan username: ");
94              String user = input.nextLine();
95              System.out.print(s:"Masukkan password: ");
96              String pass = input.nextLine();
97              savedPlayer = new Player(user, pass);
98              System.out.println(x:"Registrasi berhasil. Silakan login.");
99              System.out.println();
100
```

- **Baris 92 – 96. Opsi 1 (Registrasi).** *Player* **melakukan registrasi** akun pada game dengan memasukkan username dan password.
- **Baris 97.** Pada kode ini, objek *player* yang sudah dibuat langsung disimpan didalam *savedPlayer.*
- **Baris 98.** Kode ini menampilkan pernyataan bahwa registrasi berhasil.

```
101         } else if (pilihan.equals(anObject:"2")) {
102             if (savedPlayer == null) {
103                 System.out.println(x:"Belum ada akun terdaftar.");
104                 System.out.println(x:"Silakan registrasi terlebih dahulu.");
105                 System.out.println();
106             }
```

Baris 101 – 105. Opsi 2 (Login). Disini kode mendeklarasikan kemungkinan dimana **jika savedPlayer yang ada kosong/null** atau objek *player* belum terbentuk maka akan di berikan tampilan **saran agar *user* melakukan registrasi** terlebih dahulu sebelum login.

```
107
108             System.out.print(s:"Username: ");
109             String uN = input.nextLine();
110             System.out.print(s:"Password: ");
111             String pS = input.nextLine();
112
```

Baris 108 – 111. Opsi 2 (Login). *Player* melakukan login akun pada game dengan memasukkan username dan password yang telah dibuat ataupun random.

```
113             if (savedPlayer.username.equals(uN) && savedPlayer.password.equals(pS)) {
114                 savedPlayer.login(uN, pS);
115                 loggedIn = true;
116                 System.out.println("Selamat datang, " + uN);
117             } else {
118                 System.out.println(x:"Login gagal. Silakan coba lagi.");
119             }
120         }
121     }
```

Baris 113 – 118. Opsi 2 (Login). Kode ini melakukan pengecekkan dengan mencocokkan data yang diinput dengan data yang tersimpan. **Jika benar** maka

player akan disambut dan langsung masuk kedalam game. **Jika salah,** maka program menampilkan gagal login dan menyarankan *player* mencoba lagi.

```
123         //ADVENTURE
124         while (true) {
125             System.out.println(x:"\n=== Petualangan Dimulai ===");
126             System.out.println(x:"1. Maju (Lawan monster)");
127             System.out.println(x:"2. Mundur (Istirahat & cari bahan)");
128             System.out.println(x:"3. Lihat Inventori");
129             System.out.println(x:"4. Keluar");
130             System.out.print(s:"Pilih: ");
131             String aksi = input.nextLine();
```

- **Baris 124 – 130.** Kode ini **menampilkan menu petualangan game** sebelum masuk ke dalam game dengan beberapa opsi yang ada. Dengan syarat, *player* harus sudah login untuk bisa memainkan game.
- **Baris 131.** Kode ini mendeklarasikan pilihan akan langsung masuk dan dieksekusi setelah diinput sesuai urutan FIFO (First in First out) jadi memilih pilihan harus satu-satu secara berurut..

```
133             if (aksi.equals(anObject:"1")) {
134                 if (!savedPlayer.justFought) {
135                     savedPlayer.fightMonster();
136                 } else {
137                     System.out.println (x:"Kamu baru saja bertarung, istirahat dulu sebelum bertarung lagi.");
138                 }
```

Baris 133 – 137. Opsi 1. Kode ini melakukan pemanggilan pada metode *fightMonster()* untuk *player* **melawan monster untuk mendapatkan bahan dress** namun dengan **Batasan *justFought*** bahwa *player* **tidak bisa spam *fight*** dan akan **diperingatkan untuk beristirahat** terlebih dahulu untuk bertarung setelahnya.

```
139             } else if (aksi.equals(anObject:"2")) {
140                 savedPlayer.healPlayer();
141                 savedPlayer.findMaterial();
```

Baris 139 - 141. Opsi 2. Kode ini melakukan pemanggilan pada metode *healPlayer()* untuk *player* **menyembuhkan diri *player*** dan disaat bersamaan kode juga memanggil metode *findMaterial()* agar *player* dapat **mencari material/kain** untuk membuat dress.

```
142             } else if (aksi.equals(anObject:"3")) {
143                 savedPlayer.showInventory();
```

Baris 142 & Baris 143. Opsi 3. Kode ini melakukan pemanggilan pada metode *showInvetory()* untuk *player* **melihat isi inventory** miliknya untuk mengetahui berapa jumlaah dress, berapa *health player*, dan berapa jumlah kain yang ada.

```
144             } else if (aksi.equals(anObject:"4")) {
145                 System.out.println(x:"Terima kasih telah bermain!");
146                 break;
```

Baris 144 – 146. Opsi 4. Kode ini mengakhiri games dan keluar dari program.

```
147             } else {
148                 System.out.println(x:"Pilihan tidak dikenali.");
149             }
150         } input.close();
151     }
152 }
153
```

Baris 147. OPTIONAL. Muncul jika *user* salah memasukkan nomor pilihan.

Baris 150. OPTIONAL. Penutup objek *Scanner*.

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE

=== GATHDRE GAME ===

=== MAIN MENU ===
1. Register
2. Login
Pilih: 1
Masukkan username: Sofia
Masukkan password: 101
Registrasi berhasil. Silakan login.


=== MAIN MENU ===
1. Register
2. Login
Pilih: 2
Username: Sora
Password: 101
Login gagal. Silakan coba lagi.

=== MAIN MENU ===
1. Register
2. Login
Pilih: 2
Username: Sofia
Password: 101
Login berhasil.
Selamat datang, Sofia
```

Gambar 4.3 Output (Login/Registrasi)

```
=== Petualangan Dimulai ===
1. Maju (Lawan monster)
2. Mundur (Istirahat & cari bahan)
3. Lihat Inventori
4. Keluar
Pilih: 1
Monster muncul! Kamu bertarung...
Kamu berhasil mengalahkan monster! HP tersisa: 70
Kamu menang dan mendapatkan: Dress Lv.2

=== Petualangan Dimulai ===
1. Maju (Lawan monster)
2. Mundur (Istirahat & cari bahan)
3. Lihat Inventori
4. Keluar
Pilih: 1
Kamu baru saja bertarung, istirahat dulu sebelum bertarung lagi.

=== Petualangan Dimulai ===
1. Maju (Lawan monster)
2. Mundur (Istirahat & cari bahan)
3. Lihat Inventori
4. Keluar
Pilih: 2
Kamu beristirahat dan memulihkan diri. HP : 90
Dress yang ditemukan memiliki bahan : Kain Satin

=== Petualangan Dimulai ===
1. Maju (Lawan monster)
2. Mundur (Istirahat & cari bahan)
3. Lihat Inventori
4. Keluar
Pilih: 3
HP   : 90
Dress: Dress Lv.2
Jumlah Bahan: 1
Kain Satin
```

Gambar 4.4 Output (Petualangan)

V. Kesimpulan

Berdasarkan hasil dari penerapan studi kasus pada program yang telah dibuat, dapat disimpulkan bahwa sistem ini berhasil memenuhi kebutuhan untuk merancang sebuah permainan petualangan berbasis teks yang dilengkapi dengan fitur login, pertempuran, penyembuhan, serta pengelolaan inventaris secara otomatis dan teratur. Program ini mampu merekam informasi pengguna, mengatur aktivitas petualangan seperti pertempuran dan istirahat sesuai dengan perubahan kondisi kesehatan pemain, serta menyimpan koleksi barang berupa pakaian dan material tekstil. Dengan pengaturan menu yang terstruktur, sistem ini mempermudah pengguna dalam melaksanakan proses permainan dan meningkatkan pengalaman bermain yang lebih interaktif dan terarah.

Dengan ini kami menyampaikan laporan praktikum THT mengenai Pemrograman Berorientasi Objek (PBO). Apabila terdapat kesalahan atau kekurangan dalam penyampaian ini, saya mohon untuk dimaafkan. Saya menyampaikan rasa terima kasih saya.

VI. Daftar Pustaka

Muhardian, A. (2018). Tutorial pemrograman Java untuk pemula. Diakses dari https://www.petanikode.com/tutorial/java/