LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL 8
Jawa Inheritence (Pewarisan) 2

Disusun Oleh:

Gerald Eberhard

(105223002)

PROGRAM STUDI ILMU KOMPUTER

FAKULTAS SAINS DAN KOMPUTER

UNIVERSITAS PERTAMINA

2025

1. Pendahuluan

Studi kasus praktikum ini adalah pembuatan game petualangan berbasis teks sederhana yang memungkinkan pengguna untuk login, mendaftar, dan menjelajahi dunia terbuka. Sistem ini dirancang menggunakan konsep Pemrograman Berorientasi Objek (PBO) dalam bahasa Java untuk memodelkan entitas seperti pemain, musuh, dan ransel (inventory). Setiap entitas memiliki atribut dan perilaku spesifik, seperti data pemain (darah, mana, level, peran), pengelolaan item di ransel, dan interaksi dengan musuh (serang atau lari). Program ini memungkinkan pengguna untuk mendaftar dengan nama pengguna dan kata sandi, memilih peran (Penyihir atau Pejuang), menjelajahi dunia dengan peristiwa acak (menemukan item, musuh, atau harta), serta mengelola ransel dan ramuan.

Codingan ini terdiri dari satu kelas utama, yaitu GamePetualangan.java, yang berfungsi sebagai pusat logika permainan. Kelas ini mengelola semua fungsi seperti registrasi, login, pengelolaan ransel, petualangan, dan pertarungan. Data disimpan menggunakan HashMap untuk menyimpan informasi pengguna dan pemain dalam memori. Program ini menggunakan pendekatan berbasis teks untuk memberikan pengalaman bermain yang interaktif.

2. Variabel

No	Nama Variabel	Tipe Data	Fungsi
1	pengguna	HashMap<String, HashMap <String, String>>	Menyimpan data pengguna seperti nama pengguna, kata sandi, dan peran.
2	dataPemain	HashMap<String, Object>	Menyimpan data pemain seperti nama pengguna, peran, darah, mana, level, dan ransel.
3	scanner	Scanner	Objek untuk menerima input dari pengguna melalui terminal.
4	random	Random	Objek untuk meng hasilkan nilai acak, seperti peristiwa acak selama petualangan.
5	namaPengguna	String	Menyimpan nama pengguna yang digunakan untuk login atau registrasi.
6	peran	String	Menyimpan peran pemain (Penyihir atau Pejuang).

7	darah	double	Menyimpan nilai darah pemain.
8	mana	double	Menyimpan nilai mana pemain.
9	level	HashMap	Menyimpan item pemain seperti ramuan, makanan, dan harta.
10	ransel	integer	Variabel ini diciptakan untuk menyimpan total sudah berapa banyak object buku yang dibuat.
11	slotRanselMaks	int	Menyimpan kapasitas maksimum slot ransel (konstanta bernilai 10).
12	musuh	HashMap<String, Object>	Menyimpan data musuh seperti nama, darah, mana, dan kekuatan serang.
13	arah	String	Array yang menyimpan opsi arah pergerakan (Maju, Bergerak ke kanan, Bergerak ke kiri, Keluar dari permainan).
14	peristiwa	String	Array yang menyimpan jenis peristiwa acak (item, musuh, harta).
15	bobot	int	Array yang menyimpan bobot probabilitas untuk peristiwa acak (40, 40, 20).

3. Constructor dan Method

Nama Method	Jenis Method	Fungsi
GamePetualangan	Constructor	Secara implisit digunakan untuk membuat objek kelas utama (tidak eksplisit didefinisikan).
main	Prosedur	Menjalankan program utama, menginisialisasi data pemain, dan memanggil menu utama.
bersihkanLayar	Prosedur	Membersihkan layar terminal dengan mencetak baris kosong.
registrasi	Fungsi	Menangani proses pendaftaran pengguna baru, menyimpan nama pengguna, kata sandi, dan peran.

		Mengembalikan nilai boolean untuk keberhasilan registrasi.
login	Fungsi	Menangani proses login pengguna, memverifikasi nama pengguna dan kata sandi. Mengembalikan nilai boolean untuk keberhasilan login.
bukaRansel	Fungsi	Menampilkan informasi pemain (darah, mana, level) dan isi ransel. Mengembalikan nilai boolean untuk menentukan apakah kembali ke permainan.
gunakanRamuan	Prosedur	Menggunakan ramuan untuk memulihkan darah (+50) dan mana (+10) pemain jika tersedia.
bertarung	Prosedur	Menangani pertarungan dengan musuh, memberikan opsi untuk menyerang atau lari, dan mengelola darah pemain serta musuh.
menuUtama	Prosedur	Menampilkan menu utama (Masuk, Daftar, Keluar) dan mengatur alur program berdasarkan pilihan pengguna.
menuPermainan	Prosedur	Menampilkan menu permainan (Buka Ransel, Mulai Petualangan, Gunakan Ramuan, Keluar) dan mengatur alur permainan.

4. **Dokumentasi dan Pembahasan Code**

Program pada THT kali ini berfokus pada pembuatan game petualangan berbasis teks menggunakan paradigma PBO. Berikut adalah langkah-langkah utama dalam kode:

Pertama, program menginisialisasi data pemain dalam metode main. Data seperti darah, mana, level, dan ransel disimpan dalam HashMap bernama dataPemain. Hal ini dijalankan dalam codingan berikut:

```
Run main | Debug main | Run | Debug | Windsurf: Refactor | Explain | Generate Javadoc | ×
12  public static void main(String[] args) {
13      // Inisialisasi data pemain
14      dataPemain.put(key:"namaPengguna", value:"");
15      dataPemain.put(key:"peran", value:"");
16      dataPemain.put(key:"darah", value:100.0);
17      dataPemain.put(key:"mana", value:100.0);
18      dataPemain.put(key:"level", value:1.0);
19      HashMap<String, Integer> ransel = new HashMap<>();
20      ransel.put(key:"ramuan", value:3);
21      ransel.put(key:"makanan", value:0);
22      ransel.put(key:"harta", value:0);
23      dataPemain.put(key:"ransel", ransel);
24      dataPemain.put(key:"slotRanselMaks", value:10);
25
26      System.out.println(x:"Selamat datang di Game Petualangan!");
27      menuUtama();
28  }
29
```

Kemudian, program memungkinkan pengguna untuk mendaftar melalui metode registrasi. Sebelum pengguna mendaftar sebelumnya program akan memanggil method bersihkan layar untuk mencetak 50 baris kosong untuk memberikan jarak dan sedikit kesan ekslusif. Metode ini akan meminta nama pengguna, kata sandi, dan peran, lalu menyimpan data ke dalam variabel pengguna. Berikut ini codingan menu utama login:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
private static void menuUtama() {
    while (true) {
        bersihkanLayar();
        System.out.println(x:"1. Masuk");
        System.out.println(x:"2. Daftar");
        System.out.println(x:"3. Keluar");
        System.out.print(s:"Masukkan pilihan Anda: ");
        String pilihan = scanner.nextLine();
        if (pilihan.equals(anObject:"1")) {
            if (login()) {
                menuPermainan();
            }
        } else if (pilihan.equals(anObject:"2")) {
            registrasi();
        } else if (pilihan.equals(anObject:"3")) {
            System.out.println(x:"Selamat tinggal!");
            break;
        } else {
            System.out.println(x:"Pilihan tidak valid!");
            System.out.println(x:"Tekan Enter untuk melanjutkan...");
            scanner.nextLine();
        }
    }
}
```

Berikut ini codingan untuk program registrasi:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
private static void bersihkanLayar() {
    for (int i = 0; i < 50; i++) {
        System.out.println();
    }
}

Windsurf: Refactor | Explain | Generate Javadoc | ×
private static boolean registrasi() {
    bersihkanLayar();
    System.out.println(x:"=== Daftar ===");
    System.out.print(s:"Masukkan nama pengguna: ");
    String namaPengguna = scanner.nextLine();
    if (pengguna.containsKey(namaPengguna)) {
        System.out.println(x:"Nama pengguna sudah ada!");
        return false;
    }
    System.out.print(s:"Masukkan kata sandi: ");
    String kataSandi = scanner.nextLine();
    System.out.println(x:"Pilih peran Anda:");
    System.out.println(x:"1. Penyihir");
    System.out.println(x:"2. Pejuang");
    System.out.print(s:"Pilihan Anda: ");
    String pilihanPeran = scanner.nextLine();
    String peran = pilihanPeran.equals(anObject:"1") ? "Penyihir" : pilihanPeran.equals(anObject:"2") ? "Pejuang" : "Pejuang";
    HashMap<String, String> data = new HashMap<>();
    data.put(key:"kataSandi", kataSandi);
    data.put(key:"peran", peran);
    pengguna.put(namaPengguna, data);
    System.out.println(x:"Pendaftaran berhasil!");
    System.out.println(x:"Tekan Enter untuk melanjutkan...");
    scanner.nextLine();
    return true;
}
```

Selanjutnya, setelah login melalui metode login, pengguna dapat memulai petualangan dengan metode mulaiPetualangan. Metode ini memungkinkan pemain memilih arah dan menghadapi peristiwa acak. Berikut ini code pemograman login:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
private static boolean login() {
    bersihkanLayar();
    System.out.println(x:"=== Masuk ===");
    System.out.print(s:"Masukkan nama pengguna: ");
    String namaPengguna = scanner.nextLine();
    System.out.print(s:"Masukkan kata sandi: ");
    String kataSandi = scanner.nextLine();
    if (pengguna.containsKey(namaPengguna) && pengguna.get(namaPengguna).get(key:"kataSandi").equals(kataSandi)) {
        dataPemain.put(key:"namaPengguna", namaPengguna);
        dataPemain.put(key:"peran", pengguna.get(namaPengguna).get(key:"peran"));
        System.out.println(x:"Masuk berhasil!");
        System.out.println(x:"Tekan Enter untuk melanjutkan...");
        scanner.nextLine();
        return true;
    } else {
        System.out.println(x:"Nama pengguna atau kata sandi salah!");
        System.out.println(x:"Tekan Enter untuk melanjutkan...");
        scanner.nextLine();
        return false;
    }
}
```

Berikut ini ada codingan untuk menciptakan menu pilihan yang tercipta setelah pemain berhasil login:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
private static void menuPermainan() {
    while (true) {
        bersihkanLayar();
        System.out.println(x:"1. Buka Ransel");
        System.out.println(x:"2. Mulai Petualangan");
        System.out.println(x:"3. Gunakan Ramuan");
        System.out.println(x:"4. Keluar");
        System.out.print(s:"Masukkan pilihan Anda: ");
        String pilihan = scanner.nextLine();
        if (pilihan.equals(anObject:"1")) {
            if (!bukaRansel()) {
                break;
            }
        } else if (pilihan.equals(anObject:"2")) {
            mulaiPetualangan();
        } else if (pilihan.equals(anObject:"3")) {
            gunakanRamuan();
        } else if (pilihan.equals(anObject:"4")) {
            break;
        } else {
            System.out.println(x:"Pilihan tidak valid!");
            System.out.println(x:"Tekan Enter untuk melanjutkan...");
            scanner.nextLine();
        }
    }
}
```

Apabila pemain telah siap untuk bermain maka pemain akan memilih untuk memulai petualangan. Hal itu akan diakomodasi oleh program codingan berikut:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
123     private static void mulaiPetualangan() {
124         String[] arah = {"1. Maju", "2. Bergerak ke kanan", "3. Bergerak ke kiri", "4. Keluar dari Permainan"};
125         while (true) {
126             bersihkanLayar();
127             System.out.println("Darah Anda: " + dataPemain.get(key:"darah"));
128             System.out.println("Mana Anda: " + dataPemain.get(key:"mana"));
129             System.out.println("Level Anda: " + dataPemain.get(key:"level") + "\n");
130             for (String dir : arah) {
131                 System.out.println(dir);
132             }
133             System.out.print(s:"\nMasukkan pilihan Anda: ");
134             String pilihan = scanner.nextLine();
135             if (pilihan.equals(anObject:"4")) {
136                 break;
137             }
138             if (pilihan.equals(anObject:"1") || pilihan.equals(anObject:"2") || pilihan.equals(anObject:"3")) {
139                 String[] peristiwa = {"item", "musuh", "harta"};
140                 int[] bobot = {40, 40, 20};
141                 String peristiwaTerpilih = peristiwa[random.nextInt(peristiwa.length)];
142                 if (peristiwaTerpilih.equals(anObject:"item")) {
143                     String itemDitemukan = random.nextBoolean() ? "ramuan" : "makanan";
144                     HashMap<String, Integer> ransel = (HashMap<String, Integer>) dataPemain.get(key:"ransel");
145                     ransel.put(itemDitemukan, ransel.get(itemDitemukan) + 1);
146                     System.out.println("Anda menemukan " + itemDitemukan + "!");
147                 } else if (peristiwaTerpilih.equals(anObject:"musuh")) {
148                     HashMap<String, Object> musuh = new HashMap<>();
149                     musuh.put(key:"nama", random.nextBoolean() ? "Monster Bayangan" : "Perompak Goblin");
150                     musuh.put(key:"darah", 30.0 + random.nextDouble() * 20.0);
151                     musuh.put(key:"mana", 10.0 + random.nextDouble() * 20.0);
152                     musuh.put(key:"kekuatanSerang", 5.0 + random.nextDouble() * 10.0);
153                     bertarung(musuh);
154                 } else {
155                     HashMap<String, Integer> ransel = (HashMap<String, Integer>) dataPemain.get(key:"ransel");
156                     ransel.put(key:"harta", ransel.get(key:"harta") + 1);
157                     System.out.println(x:"Anda menemukan Harta!");
158                 }
159                 System.out.println(x:"Tekan Enter untuk melanjutkan...");
160                 scanner.nextLine();
161             } else {
162                 System.out.println(x:"Pilihan tidak valid!");
163                 System.out.println(x:"Tekan Enter untuk melanjutkan...");
164                 scanner.nextLine();
165             }
166         }
167     }
```

Jika pemain bertemu musuh, metode bertarung akan dipanggil untuk menangani pertarungan. Pemain dapat memilih untuk menyerang atau melarikan diri. Berikut codingan program untuk bertarung:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
private static void bertarung(HashMap<String, Object> musuh) {
    while ((Double) musuh.get(key:"darah") > 0 && (Double) dataPemain.get(key:"darah") > 0) {
        bersihkanLayar();
        System.out.println("Musuh: " + musuh.get(key:"nama"));
        System.out.println("Darah Musuh: " + musuh.get(key:"darah"));
        System.out.println("Mana Musuh: " + musuh.get(key:"mana"));
        System.out.println("Kekuatan Serang Musuh: " + musuh.get(key:"kekuatanSerang"));
        System.out.println("Darah Anda: " + dataPemain.get(key:"darah") + "\n");
        System.out.println(x:"1. Serang");
        System.out.println(x:"2. Lari");
        System.out.print(s:"\nMasukkan pilihan Anda: ");
        String pilihan = scanner.nextLine();
        if (pilihan.equals(anObject:"1")) {
            double kerusakanKeMusuh = dataPemain.get(key:"peran").equals(obj:"Penyihir") ? 25.0 : 20.0;
            double kerusakanKePemain = (Double) musuh.get(key:"kekuatanSerang");
            musuh.put(key:"darah", (Double) musuh.get(key:"darah") - kerusakanKeMusuh);
            dataPemain.put(key:"darah", (Double) dataPemain.get(key:"darah") - kerusakanKePemain);
            System.out.println("Anda memukul musuh sebesar " + kerusakanKeMusuh + " kerusakan");
            if ((Double) musuh.get(key:"darah") > 0) {
                System.out.println("Musuh memukul Anda sebesar " + kerusakanKePemain + " kerusakan");
            } else {
                System.out.println(x:"Anda mengalahkan musuh!");
                dataPemain.put(key:"level", (Double) dataPemain.get(key:"level") + 1);
                dataPemain.put(key:"darah", (Double) dataPemain.get(key:"darah") + 50);
                dataPemain.put(key:"mana", (Double) dataPemain.get(key:"mana") + 30);
                System.out.println(x:"Naik Level!");
                break;
            }
        } else if (pilihan.equals(anObject:"2")) {
            if (random.nextDouble() < 0.7) {
                System.out.println(x:"Anda berhasil melarikan diri!");
                break;
            } else {
                System.out.println(x:"Gagal melarikan diri!");
                dataPemain.put(key:"darah", (Double) dataPemain.get(key:"darah") - (Double) musuh.get(key:"kekuatanSerang"));
                System.out.println("Musuh memukul Anda sebesar " + musuh.get(key:"kekuatanSerang") + " kerusakan");
            }
        } else {
            System.out.println(x:"Pilihan tidak valid!");
        }
        System.out.println(x:"Tekan Enter untuk melanjutkan...");
        scanner.nextLine();
    }
    if ((Double) dataPemain.get(key:"darah") <= 0) {
        System.out.println(x:"Permainan Berakhir! Anda telah dikalahkan.");
        System.exit(status:0);
    }
}
```

Pemain juga dapat mengelola ransel dan menggunakan ramuan melalui metode bukaRansel dan gunakanRamuan. Metode ini memungkinkan pemain untuk melihat status dan memulihkan darah serta mana. Berikut codingan program mengelola ransel:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
private static boolean bukaRansel() {
    bersihkanLayar();
    System.out.println(x:"**** Informasi Data ****");
    System.out.println("Darah = " + dataPemain.get(key:"darah"));
    System.out.println("Mana = " + dataPemain.get(key:"mana"));
    System.out.println("Level = " + dataPemain.get(key:"level"));
    System.out.println(x:"\nItem di Ransel:");
    HashMap<String, Integer> ransel = (HashMap<String, Integer>) dataPemain.get(key:"ransel");
    for (String item : ransel.keySet()) {
        System.out.println(item.substring(beginIndex:0, endIndex:1).toUpperCase() + item.substring(beginIndex:1) + " = " + ransel.get(item));
    }
    int slotTerpakai = ransel.values().stream().mapToInt(Integer::intValue).sum();
    int slotKosong = (int) dataPemain.get(key:"slotRanselMaks") - slotTerpakai;
    for (int i = 0; i < slotKosong; i++) {
        System.out.println("Slot " + (i + 1) + ": kosong");
    }
    System.out.print(s:"\nKembali ke permainan (ya/tidak)? ");
    return scanner.nextLine().toLowerCase().equals(anObject:"ya");
}
```

Berikut codingan untuk menggunakan ramuan:

```
Windsurf: Refactor | Explain | Generate Javadoc | ×
private static void gunakanRamuan() {
    bersihkanLayar();
    HashMap<String, Integer> ransel = (HashMap<String, Integer>) dataPemain.get(key:"ransel");
    if (ransel.get(key:"ramuan") > 0) {
        ransel.put(key:"ramuan", ransel.get(key:"ramuan") - 1);
        dataPemain.put(key:"darah", (double) dataPemain.get(key:"darah") + 50.0);
        dataPemain.put(key:"mana", (double) dataPemain.get(key:"mana") + 10.0);
        System.out.println(x:"Ramuan digunakan dengan sukses!");
        System.out.println("Darah Anda: " + dataPemain.get(key:"darah"));
        System.out.println("Mana Anda: " + dataPemain.get(key:"mana"));
        System.out.println("Level Anda: " + dataPemain.get(key:"level"));
    } else {
        System.out.println(x:"Tidak ada ramuan yang tersedia!");
    }
    System.out.println(x:"Tekan Enter untuk melanjutkan...");
    scanner.nextLine();
}
```

5. Kesimpulan

Praktikum ini berhasil mengimplementasikan game petualangan berbasis teks sederhana menggunakan konsep Pemrograman Berorientasi Objek seperti enkapsulasi (melalui atribut private dan metode akses) serta modularitas dalam satu kelas utama. Program ini memenuhi kebutuhan studi kasus dengan mengelola data pengguna, pemain, dan interaksi permainan secara terstruktur menggunakan HashMap. Pengalaman ini sangat membantu saya memahami penerapan materi PBO dalam menyelesaikan masalah nyata, seperti pembuatan game interaktif. Sistem ini dapat menangani registrasi, login, petualangan, pertarungan, dan pengelolaan inventory dengan baik. Sekian dan terima kasih.

6. Daftar Pustaka

Modul 8: Java Inheritence (Pewarisan) 2