LAPORAN

PEMROGRAMAN BERORIENTASI OBJEK

Nama : Muhammad Andi Syaifullah

Nim : 13020220015

Dosen : Mardiyyah Hasnawi, S.Kom.,M.T.

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS MUSLIM INDONESIA

MAKASSAR

2024

EVALUSI PRAKTIKUM I

1. Apakah perbedaan antara struktur kontrol percabangan if-else dan switch- case?

2. Kapan digunakan struktur kontrol if-else dan switch-case

3. Pada program 2, tambahkan perintah untuk memilih 2 opsi menggunakan kontrol switch..case. opsi pilihah

1=inputNilai()

Pilihan 2=inputNilaiBaru()

4. Apakah perbedaan antara struktur kontrol perulangan while dan do-while?

5. Kapan digunakan struktur kontrol for?

6. Apakah perbedaan antara Array dan ArrayList? berilah contoh masing-masing!

7. Buatlah contoh program yang mengimplementasikan HashMap dengan memasukkan nilai dan key melalui keyboad!

JAWAB

1. Perbedaan struktur control percabangan if-else dan switch case yakni sebagai berikut :

a. Penggunaan:

- If-else digunakan untuk membuat keputusan berdasarkan satu kondisi atau serangkaian kondisi yang saling terkait.

- Switch-case digunakan ketika terdapat banyak kondisi atau opsi yang berbeda dan keputusan harus dibuat berdasarkan nilai yang sama dengan beberapa nilai-nilai yang sudah ditentukan.

b. Ekspresi kondisi:

- If-else menggunakan ekspresi kondisi yang dapat menghasilkan nilai benar atau salah (true atau false).
- Switch-case menggunakan nilai yang sesuai dengan salah satu kasus atau opsi yang sudah ditentukan sebelumnya.

c. Pengecekan kondisi:

- If-else mengevaluasi kondisi secara berurutan, mulai dari kondisi pertama hingga kondisi terakhir, dan menjalankan blok kode yang sesuai dengan kondisi yang pertama kali terpenuhi.
- Switch-case mengecek nilai ekspresi dan menjalankan blok kode yang sesuai dengan nilai tersebut. Switch-case dapat lebih efisien daripada if- else ketika ada banyak kasus yang perlu diperiksa.

d. Fleksibilitas:

- If-else lebih fleksibel karena dapat menangani berbagai macam kondisi dengan lebih detail dan kompleksitas.
- Switch-case lebih cocok digunakan ketika ada banyak kasus atau opsi yang berbeda dan setiap kasus hanya memerlukan satu nilai yang dapat dibandingkan.

e. Penanganan nilai default:

- If-else dapat menangani nilai default dengan menggunakan blok else terakhir untuk menangani kondisi yang tidak terpenuhi oleh kondisi sebelumnya.
- Switch-case memiliki kasus default yang dapat digunakan untuk menangani nilai yang tidak cocok dengan kasus-kasus yang sudah didefinisikan

sebelumnya.

2. Kapan digunakan if-else dan switch-case jika sebagai berikut :

A. If-else:

- Kondisi kompleks: Jika keputusan yang dibuat bergantung pada kondisi yang kompleks atau kombinasi dari beberapa kondisi, if-else bisa lebih cocok. Misalnya, jika Anda perlu memeriksa beberapa variabel atau
  ekspresi yang saling terkait.

- Kondisi bersarang: Jika Anda perlu mengevaluasi kondisi secara bersarang (nested), if-else lebih fleksibel dalam menangani situasi ini

- Ekspresi boolean: Jika keputusan berdasarkan ekspresi boolean (true/false), if-else adalah pilihan yang tepat.

B. Switch-case:

- Banyak opsi: Jika Anda memiliki banyak opsi atau kasus yang berbeda dan keputusan bergantung pada nilai yang sama, switch-case bisa lebih efisien dan lebih mudah dibaca.

- Penanganan nilai tunggal: Jika keputusan hanya bergantung pada satu nilai atau ekspresi yang harus dibandingkan dengan beberapa nilai yang sudah ditentukan sebelumnya, switch-case cocok digunakan.

- Optimasi performa: Dalam beberapa bahasa pemrograman, switch-case dapat dioptimalkan secara internal untuk mempercepat pemrosesan kasus-kasus.

3. Contoh Kode program 2 :

Pada program 2, tambahkan perintah untuk memilih 2 opsi menggunakan kontrol switch..case. opsi pilihah 1=inputNilai() Pilihan 2=inputNilaiBaru()

Program 2

```
package pertemuan2.modul3.nilai;
import java.util.Scanner;
public class TestNilai {
    public static void main(String[] args) {
        HitungRata hitung = new HitungRata();
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan Jumlah Data : ");
        int banyakData = input.nextInt();
        int nilai[] = new int[banyakData];
        System.out.print("Masukkan Nilai : ");
        hitung.inputNilai(nilai);
        System.out.print("Daftar Nilai : ");
        hitung.cetakNilai(nilai);
        System.out.println("Rata Nilai : "+ hitung.rataNilai(banyakData));
        System.out.print("Masukkan Nilai Baru: ");
        hitung.inputNilaiBaru(banyakData);
        System.out.print("Daftar Nilai Baru : ");
        hitung.cetakNilaiBaru();
    }
}
```

Untuk menambahkan opsi pilihan menggunakan kontrol switch-case, Kita dapat mengubah blok kode diakhir methodmenjadi seperti berikut:

main()

```
package pertemuan2.modul3.nilai;

import java.util.Scanner;


public class TestNilai {

    public static void main(String[] args) {
```

```
        HitungRata hitung = new HitungRata();
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan Jumlah Data : ");
        int banyakData = input.nextInt();
        int nilai[] = new int[banyakData];
        System.out.print("Masukkan Nilai : ");
        hitung.inputNilai(nilai);
        System.out.print("Daftar Nilai : ");
        hitung.cetakNilai(nilai);
        System.out.println("Rata Nilai : "+
hitung.rataNilai(banyakData));

        System.out.println("Pilih Opsi:");
        System.out.println("1. Input Nilai");
        System.out.println("2. Input Nilai Baru");
        System.out.print("Pilihan Anda: ");
        int pilihan = input.nextInt();

        switch(pilihan) {
            case 1:
                System.out.print("Masukkan Nilai Baru:
");
                hitung.inputNilai(banyakData);
                System.out.print("Daftar Nilai Baru :
");
                hitung.cetakNilai(nilai);
```

```
                break;
            case 2:
                System.out.print("Masukkan Nilai Baru:
");
                hitung.inputNilaiBaru(banyakData);
                System.out.print("Daftar Nilai Baru :
");
                hitung.cetakNilaiBaru();
                break;
            default:
                System.out.println("Pilihan tidak
valid.");
        }
    }
}
```

4. Perbedaan antara while dan do-while:

- while memeriksa kondisi sebelum melakukan perulangan, sehingga jika kondisi awal tidak memenuhi syarat, blok perulangan tidak akan
dieksekusi sama sekali.

- do-while akan selalu melakukan satu iterasi minimal, karena blok perulangan akan dieksekusi terlebih dahulu sebelum memeriksa kondisi.

5. Kapan digunakan struktur kontrol for?

- For digunakan ketika kita perlu melakukan iterasi pada sekumpulan data yang telah diketahui, seperti ketika kita perlu melakukan perulangan pada array, list, atau string.

Perbedaan antara Array dan ArrayList:

- Array adalah tipe data yang digunakan untuk menyimpan beberapa nilai yang sama tipe, dan panjangnya sudah ditentukan pada saat deklarasi.
- ArrayList adalah tipe data yang juga digunakan untuk menyimpan beberapa nilai yang sama tipe, tetapi panjangnya dapat bervariasi dan dapat diubah sesuai kebutuhan.

Array :

angka = [1, 2, 3, 4, 5]

ArrayList :

from ArrayList import ArrayList

numbers = ArrayList() numbers.add(1) numbers.add(2) numbers.add(3)

Contoh program java yang mengimplementasikan HashMap dengan memasukkan nilai dan key melalui keyboard:

import java.util.HashMap; import java.util.Map; import java.util.Scanner;

public class HashMapExample { public static void main(String[] args) {

Scanner scanner = new Scanner(System.in); Map<String, Integer> map = new HashMap<>();

```
        // Memasukkan data ke dalam HashMap System.out.println("Masukkan
        jumlah data yang ingin dimasukkan:"); int jumlahData = scanner.nextInt();

        scanner.nextLine(); // Membersihkan newline



        for (int i = 0; i < jumlahData; i++) { System.out.println("Masukkan key:");
        String key = scanner.nextLine();


        System.out.println("Masukkan nilai:"); int value = scanner.nextInt();

        scanner.nextLine(); // Membersihkan newline



        map.put(key, value);

        }



        // Menampilkan data yang tersimpan dalam HashMap


        System.out.println("Data dalam HashMap:");

        for (Map.Entry<String, Integer> entry : map.entrySet()) {
        System.out.println("Key: " + entry.getKey() + ", Value: " +
        entry.getValue());

        }

        scanner.close();

        }

}
```

OUTPUT PROGRAM :

EVALUASI PRAKTIKUM II

1. Berdasarkan ke tiga program di atas Class utama, Class Orang dan Class Mahasiswa, manakah yang menunjukkan konsep pewarisan dan polimorfisme! Jelaskan konsep tersebut sesuai program tersebut!

   Jawab:

- Pewarisan (Inheritance): Pewarisan terjadi ketika sebuah kelas mewarisi sifat-sifat (properti dan metode) dari kelas lain. Dalam hal ini, kelas Mahasiswa mewarisi sifat-sifat dari kelas Orang.
- Dalam Program 3 (Mahasiswa.java), perhatikan bahwa kelas Mahasiswa menggunakan kata kunci extends Orang. Ini menunjukkan bahwa kelas Mahasiswa mewarisi sifat-sifat dari kelas Orang. Dengan demikian, kelas Mahasiswa memiliki akses ke properti nama dari kelas Orang.

Contoh Penggunaan Pewarisan :

```
public class Mahasiswa extends Orang { private String
stb;

// Konstruktor  public Mahasiswa() {

super();

this.stb = "1302002134"; // Stambuk anda

}

public Mahasiswa(String stb, String nama) { this.nama =
nama;

this.stb = stb;

 }

}
```

-

Polimorfisme:

Polimorfisme adalah kemampuan suatu objek untuk memiliki banyak bentuk atau perilaku yang berbeda tergantung pada konteks penggunaannya. Dalam hal ini, polimorfisme dapat dilihat dalam konstruktor kelas Orang dan Mahasiswa.

Dalam Program 1 (Orang.java), terdapat dua konstruktor: satu tanpa parameter dan satu dengan parameter nama. Konstruktor tanpa parameter digunakan untuk menginisialisasi nama dengan nilai default "Aminah". Konstruktor dengan parameter nama memungkinkan pengguna untuk menginisialisasi nama sesuai dengan nilai yang diberikan.

Dalam Program 3 (Mahasiswa.java), terdapat dua konstruktor juga: satu tanpa parameter yang memanggil konstruktor tanpa parameter dari kelas induk (Orang), dan satu dengan parameter stb dan nama yang menginisialisasi properti nama dan stb sesuai dengan nilai yang diberikan.

Contoh penggunaan polimorfisme:

```
// Program 1 :
Orang.java
public class
Orang {
  public String nama;

  // Konstruktor
  tanpa parameter
  public Orang() {
    this.nama= "Aminah";
  }
```

```
  // Konstruktor dengan
  parameter nama public
  Orang(String nama) {
    this.nama = nama;
  }
}

// Program 3 : Mahasiswa.java
public class Mahasiswa
  extends Orang { private
  String stb;

  // Konstruktor tanpa parameter yang memanggil
konstruktor Orang tanpa parameter
  public
    Mahasiswa
    () {
    super();
    this.stb = "1302002134"; // Stambuk anda
  }

  // Konstruktor dengan parameter
  stb dan nama public
  Mahasiswa(String stb, String
  nama) {
    this.nama
```

-

```
        = nama;
        this.stb
        = stb;
    }
}
```

Dengan demikian, konsep pewarisan (inheritance) terjadi antara " kelas Orang " dan " Mahasiswa ", sedangkan konsep polimorfisme terjadi dalam penggunaan konstruktor yang memiliki banyak bentuk dan perilaku berbeda tergantung pada konteksnya.

2. Tambahkan static pada method info() Class Orang dan Class Mahasiswa kemudian lakukan pemanggilan method info() pada program utama (Class utama)!

   Jawab

   Berikut adalah pembaruan pada kelas Orang dan Mahasiswa dengan menambahkan metode info() yang bersifat statis (static), serta pemanggilan metode info() pada kelas Utama:

A. Pembaharuan pada " class orang "

```
public class Orang {
        public String nama;

        public
          Orang() {
          this.nama
          =
          "Aminah";
        }
```

```
    public
      Orang(String
      nama) {
      this.nama =
      nama;
    }

    public static void info() {
System.out.println("Ini adalah kelas Orang.");
    }

}
```

A. Pembaharuan pada " Class Mahasiswa "

```
Mahasiswa extends Orang {
  private String stb;

  public
    Mahasiswa()
    { super();
    this.stb = "1302002134";
  }

  public Mahasiswa(String stb, String nama) {

    this.nama =
    nama;
    this.stb =
    stb;
  }

public static void info() {
  System.out.println("Ini adalah kelas Mahasiswa.");
  }

}
```

A. Pembaharuan pada Metode info () pada "Class utama"

-

```
public class Utama {
  public static void main(String[] args) {
    Orang.info(); // Memanggil metode info()
    dari kelas Orang Mahasiswa.info(); //
    Memanggil metode info() dari kelas
Mahasiswa
  }
}
```

OUTPUT PROGRAM :

```
Ini adalah kelas Orang.
Ini adalah kelas Mahasiswa.
```