

TUGAS 4

LAPORAN PEMROGRAMAN BERORIENTASI OBJEK

RAIHAN APRIANSYAH || 13020220014

EVALUASI PRAKTIKUM

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

- Kondisi:
 - if-else digunakan ketika terdapat satu kondisi yang harus dievaluasi.
 - switch-case digunakan ketika terdapat beberapa kemungkinan kondisi yang akan dievaluasi.
- Tipe Kondisi:
 - if-else dapat mengevaluasi ekspresi boolean (true/false) atau kondisi lain yang menghasilkan nilai boolean.
 - switch-case biasanya digunakan dengan nilai-nilai konstan atau enum.
- Eksekusi:
 - Dalam if-else, setiap kondisi dievaluasi secara berurutan, dan hanya satu blok pernyataan yang dieksekusi.
 - Dalam switch-case, nilai ekspresi switch dievaluasi sekali, dan kemudian pernyataan yang sesuai dengan nilai tersebut dieksekusi. Setelah itu, eksekusi dilanjutkan hingga akhir blok switch atau hingga ditemukan pernyataan break.
- Kondisi Default:
 - Dalam if-else, tidak ada yang mirip dengan kondisi default. Jika tidak ada kondisi yang benar, tidak ada blok pernyataan khusus yang dieksekusi.
 - Dalam switch-case, terdapat blok default yang dieksekusi jika tidak ada nilai case yang cocok dengan nilai switch.
- Comparasi Nilai:
 - if-else menggunakan operasi perbandingan untuk mengevaluasi kondisi.
 - switch-case membandingkan nilai ekspresi switch dengan nilai-nilai case secara langsung.

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

➤ if-else:

- Gunakan if-else ketika Anda perlu mengevaluasi kondisi yang kompleks, seperti kombinasi dari beberapa kondisi atau kondisi bersarang.
- Jika kondisi berhubungan dengan ekspresi boolean yang kompleks, if-else lebih fleksibel dalam menangani logika tersebut.
- Ketika setiap kondisi memerlukan tindakan atau pernyataan yang berbeda, if-else memungkinkan Anda untuk menentukan tindakan yang sesuai untuk setiap kondisi.

➤ switch-case:

- Gunakan switch-case ketika Anda memiliki beberapa pilihan kondisi yang akan dievaluasi berdasarkan satu ekspresi atau variabel.
- Jika Anda membandingkan nilai yang sama dengan beberapa nilai konstan atau enum, switch-case lebih cocok dan membuat kode lebih mudah dibaca.
- Jika setiap kasus memerlukan eksekusi yang sama, atau hampir sama, switch-case dapat mengurangi pengulangan kode.

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

opsi pilihan 1=inputNilai()

Pilihan 2=inputNilaiBaru()

➤ Coding :

```
1 package testnilai;
2
3 /*
4  * Author : 13020220014
5  * Nama : Raihan Agrianyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 12.04
8  */
9
10 import java.util.Scanner;
11
12 public class TestNilai {
13
14     public static void main(String[] args) {
15         HitungRata hitung = new HitungRata();
16         Scanner input = new Scanner(System.in);
17         int banyakData;
18
19         System.out.print("Masukkan Jumlah Data : ");
20         banyakData = input.nextInt();
21         int nilai[] = new int[banyakData];
22
23         System.out.println("Memasukkan");
24         System.out.println("1. Input Nilai");
25         System.out.println("2. Input Nilai Baru");
26         System.out.print("Pilih opsi: ");
27         int pilihan = input.nextInt();
28
29         switch (pilihan) {
30             case 1:
31                 System.out.print("Masukkan Nilai : ");
32                 hitung.inputNilai(nilai);
33                 System.out.print("Daftar Nilai : ");
34             case 2:
35                 System.out.print("Masukkan Nilai Baru: ");
36                 hitung.inputNilaiBaru(banyakData);
37                 System.out.print("Daftar Nilai Baru : ");
38                 hitung.cetakNilaiBaru();
39                 break;
40             default:
41                 System.out.println("Pilihan tidak valid!");
42         }
43     }
44 }
45
46
47 }
```

➤ Class HitungRata

```
1 package testnilai;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 12.30
8  */
9
10 import java.util.*;
11
12 public class HitungRata {
13     private double total = 0.0;
14     private ArrayList<Integer> nilaiBaru = new ArrayList<>();
15     private Scanner input = new Scanner(System.in);
16
17     public void inputNilai(int[] nilai) {
18         for (int i = 0; i < nilai.length; i++) {
19             nilai[i] = input.nextInt();
20             total += nilai[i];
21         }
22     }
23
24     public double rataNilai(int Ndata) {
25         return total / ((double) Ndata);
26     }
27
28     public void cetakNilai(int[] nilai) {
29         for (int angka : nilai) {
30             System.out.print(angka + "\t");
31         }
32         System.out.println();
33     }
34
35     public void inputNilaiBaru(int jumlah) {
36         while (jumlah > 0) {
37             nilaiBaru.add(input.nextInt());
38             jumlah--;
39         }
40     }
41
42     public void cetakNilaiBaru() {
43         ListIterator<Integer> iterator = nilaiBaru.listIterator();
44         while (iterator.hasNext()) {
45             Integer data = iterator.next();
46             if (data == null) {
47                 System.out.println("null");
48             } else {
49                 System.out.println(data.toString());
50             }
51         }
52     }
53 }
```

➤ Output

```
Masukkan Jumlah Data : 5
Menu:
1. Input Nilai
2. Input Nilai Baru
Pilih opsi: 1
Masukkan Nilai : 130
22
14
12
4
Daftar Nilai : 130      22      14      12      4
Rata Nilai : 36.4
BUILD SUCCESSFUL (total time: 25 seconds)
```

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

➤ while:

- Evaluasi kondisi dilakukan sebelum blok pernyataan dieksekusi. Jika kondisi awalnya salah, blok pernyataan tidak akan dieksekusi sama sekali.
- Cocok digunakan ketika jumlah iterasi tidak pasti dan perlu dicek kondisi sebelum masuk ke dalam perulangan.

Contoh penggunaan while:

```
int i = 0;
while (i < 5) {
    System.out.println("Nilai i: " + i);
    i++;
}
```

➤ do-while:

- Evaluasi kondisi dilakukan setelah blok pernyataan dieksekusi, sehingga setidaknya blok pernyataan akan dieksekusi sekali bahkan jika kondisi awalnya salah.
- Cocok digunakan ketika setidaknya satu iterasi harus dilakukan, meskipun kondisi awalnya salah.

Contoh penggunaan do-while:

```
int i = 0;
do {
    System.out.println("Nilai i: " + i);
    i++;
} while (i < 5);
```

5. Kapan digunakan struktur kontrol for?

- Struktur kontrol for digunakan ketika kita tahu jumlah iterasi yang akan dilakukan sebelumnya atau ketika kita ingin menggunakan loop untuk melakukan iterasi dengan baik melalui array, koleksi, atau rentang bilangan tertentu. Sintaks for juga cocok untuk melakukan inisialisasi, pengujian kondisi, dan peningkatan variabel loop dalam satu baris.

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

➤ Array:

- Array adalah struktur data dalam bahasa pemrograman yang dapat menyimpan sejumlah elemen dengan tipe data yang sama.
- Ukuran array biasanya tetap (statis), artinya setelah array dibuat, ukurannya tidak dapat diubah.
- Untuk mengakses elemen array, Anda menggunakan indeks berbasis nol.

Contoh penggunaan array dalam Java:

```
int[] numbers = {1, 2, 3, 4, 5};
System.out.println(numbers[0]);
```

➤ **ArrayList:**

- ArrayList adalah kelas dalam Java yang merupakan bagian dari kerangka kerja koleksi (collections framework) dan digunakan untuk menyimpan kumpulan elemen.
- Ukuran ArrayList dapat berubah (dinamis), artinya Anda dapat menambah atau menghapus elemen tanpa harus membuat array baru.
- ArrayList menyediakan berbagai metode untuk menambah, menghapus, atau mengakses elemen, serta untuk melakukan operasi lainnya.

Contoh penggunaan ArrayList dalam Java:

```
import java.util.ArrayList;

ArrayList<Integer> numbers = new ArrayList<>();

numbers.add(1);

numbers.add(2);

numbers.add(3);

System.out.println(numbers.get(0));
```

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

➤ **Codingan :**

```

1
2
3  /*
4   @author 13020220014
5   Nama : Raihan Apriansyah
6   Materi : Tugas 4 Evaluasi Praktikum 1
7   Waktu : 27/03/2024, 13.04
8   */
9
10 import java.util.HashMap;
11 import java.util.Map;
12 import java.util.Scanner;
13
14 public class ContohHash {
15     public static void main(String[] args) {
16         Scanner scanner = new Scanner(System.in);
17         Map<String, Integer> hashMap = new HashMap<>();
18
19         System.out.println("Masukkan jumlah data yang ingin dimasukkan:");
20         int jumlahData = scanner.nextInt();
21         scanner.nextLine();
22
23         for (int i = 0; i < jumlahData; i++) {
24             System.out.println("Masukkan kunci (key) ke-" + (i + 1) + ":");
25             String key = scanner.nextLine();
26             System.out.println("Masukkan nilai ke-" + (i + 1) + ":");
27             int value = scanner.nextInt();
28             scanner.nextLine();
29
30             hashMap.put(key, value);
31         }
32         System.out.println("Isi HashMap:");
33         for (Map.Entry<String, Integer> entry : hashMap.entrySet()) {
34             System.out.println("Key: " + entry.getKey() + ", Value: " + entry.getValue());
35         }
36     }
37 }

```

➤ Output

```
Masukkan jumlah data yang ingin dimasukkan:
2
Masukkan kunci (key) ke-1:
130
Masukkan nilai ke-1:
22
Masukkan kunci (key) ke-2:
14
Masukkan nilai ke-2:
12
Isi HashMap:
Key: 14, Value: 12
Key: 130, Value: 22
BUILD SUCCESSFUL (total time: 17 seconds)
```

EVALUASI PRAKTIKUM

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!

➤ Dalam ketiga program di atas, konsep pewarisan dan polimorfisme terlihat pada kelas Mahasiswa.

➤ Pewarisan (Inheritance):

Kelas Mahasiswa merupakan subkelas dari kelas Orang, yang berarti Mahasiswa mewarisi semua atribut dan metode dari kelas Orang. Pada konstruktor tanpa parameter di kelas Mahasiswa, terdapat pemanggilan `super()` yang mengacu pada konstruktor kelas induk (Orang), sehingga konstruktor kelas induk akan dieksekusi terlebih dahulu sebelum konstruktor kelas Mahasiswa. Dengan kata lain, kelas Mahasiswa mengambil semua karakteristik yang dimiliki oleh kelas Orang dan menambahkan atribut tambahan stb.

➤ Polimorfisme:

Terdapat dua versi dari konstruktor kelas Mahasiswa: satu tanpa parameter dan satu dengan dua parameter (nama dan stb). Konsep polimorfisme memungkinkan penggunaan metode dengan nama yang sama di kelas yang berbeda. Dalam hal ini, metode konstruktor `Mahasiswa()` tanpa parameter adalah contoh polimorfisme, di mana kelas Mahasiswa memiliki versi konstruktor yang berbeda dari kelas induknya (Orang). Meskipun kelas Mahasiswa mewarisi konstruktor tanpa parameter dari

kelas Orang, kelas Mahasiswa juga memiliki versi konstruktor tambahan yang memungkinkan inisialisasi atribut nama dan stb sekaligus.

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

➤ Utama.java

```
1 package Pertemuan2Modul413020220014;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 14.01
8  */
9
10 public class Utama {
11     public static void main(String[] args) {
12         Orang orang = new Orang();
13         orang.nama = "Raihan";
14         System.out.println("Stb : " + orang.nama);
15
16         Orang.info();
17
18         Mahasiswa mahasiswa = new Mahasiswa();
19         mahasiswa.nama = "Apriansyah";
20         System.out.println("Stb Mahasiswa : " + mahasiswa.nama);
21
22         Mahasiswa.info();
23     }
24 }
25
```

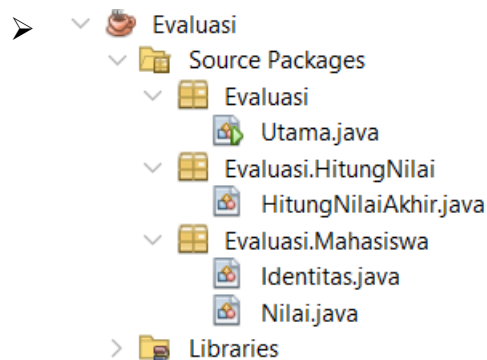
➤ Orang.java

```
1 package Pertemuan2Modul413020220014;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 14.01
8  */
9
10 public class Orang {
11     public String nama;
12
13     public Orang() {
14         this.nama = "Aminah";
15     }
16
17     public Orang(String nama) {
18         this.nama = nama;
19     }
20
21     public static void info() {
22         System.out.println("Ini adalah class Orang");
23     }
24 }
25
```

➤ Mahasiswa.java

```
1 package Pertemuan2Modul413020220014;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 14.01
8  */
9
10 public class Mahasiswa extends Orang {
11     private String stb;
12
13     public Mahasiswa() {
14         super();
15         this.stb = "13020220014";
16     }
17
18     public Mahasiswa(String stb, String nama) {
19         this.nama = nama;
20         this.stb = stb;
21     }
22
23     public static void info() {
24         System.out.println("Ini adalah class Mahasiswa");
25     }
26 }
27
```

3. Buatlah sebuah project dengan nama project stambuk anda dan buatlah pengorganisasian package dan class seperti berikut!



Setelah Itu Lengkapi Program pada soal yang telah diberikan

➤ Identitas.java

```
1 package Evaluasi.Mahasiswa;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 15.25
8  */
9
10 public class Identitas {
11     public String nama;
12     public String stambuk;
13
14     public void setName(String nama) {
15         this.nama = nama;
16     }
17
18     public String getName() {
19         return nama;
20     }
21
22     public void setStambuk(String stambuk) {
23         this.stambuk = stambuk;
24     }
25
26     public String getStambuk() {
27         return stambuk;
28     }
29 }
30
```

➤ Nilai.java

```
1 package Evaluasi.Mahasiswa;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 15.10
8  */
9
10 public class Nilai {
11     private int tugas1;
12     private int tugas2;
13     private int mid;
14     private int uas;
15
16
17     public void setTugas1(int tugas1) {
18         this.tugas1 = tugas1;
19     }
20
21     public int getTugas1() {
22         return tugas1;
23     }
24
25     public void setTugas2(int tugas2) {
26         this.tugas2 = tugas2;
27     }
28
29     public int getTugas2() {
30         return tugas2;
31     }
32
33     public void setMid(int mid) {
34         this.mid = mid;
35     }
36
37     public int getMid() {
38         return mid;
39     }
40
41
42     public void setUas(int uas) {
43         this.uas = uas;
44     }
45
46     public int getUas() {
47         return uas;
48     }
49 }
```

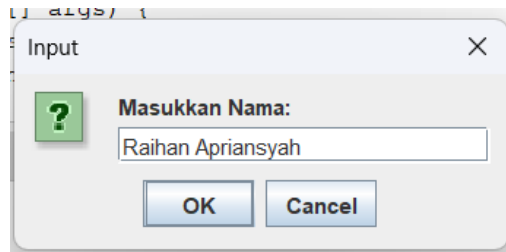
➤ HitungNilaiAkhir.java

```
1 package Evaluasi.HitungNilai;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 15.30
8  */
9
10 public class HitungNilaiAkhir {
11     public double nilaiTugas(int tugas1, int tugas2) {
12         double rataTugas = (tugas1 + tugas2) / 2.0;
13         return rataTugas;
14     }
15
16     public double nilaiAkhir(double tugas, int mid, int uas) {
17         double nilaiAkhir = (tugas * 0.4) + (mid * 0.3) + (uas * 0.3);
18         return nilaiAkhir;
19     }
20 }
```

➤ Utama.java

```
1 package Evaluasi;
2
3 /*
4  * @author 13020220014
5  * Nama : Raihan Apriansyah
6  * Materi : Tugas 4 Evaluasi Praktikum 1
7  * Waktu : 27/03/2024, 15.40
8  */
9
10 import Evaluasi.HitungNilai.HitungNilaiAkhir;
11 import Evaluasi.Mahasiswa.*;
12
13 import javax.swing.JOptionPane;
14
15
16 public class Utama {
17
18     public static void main(String[] args) {
19         String nama = JOptionPane.showInputDialog("Masukkan Nama:");
20         String stambuk = JOptionPane.showInputDialog("Masukkan Stambuk:");
21
22         int tugas1 = Integer.parseInt(JOptionPane.showInputDialog("Masukkan Tugas 1:"));
23         int tugas2 = Integer.parseInt(JOptionPane.showInputDialog("Masukkan Tugas 2:"));
24         int mid = Integer.parseInt(JOptionPane.showInputDialog("Masukkan Nilai Mid:"));
25         int uas = Integer.parseInt(JOptionPane.showInputDialog("Masukkan Nilai UAS:"));
26
27         Identitas mhs = new Identitas();
28         mhs.setNama(nama);
29         mhs.setStambuk(stambuk);
30
31         HitungNilaiAkhir hitungNilai = new HitungNilaiAkhir();
32         double tugas = hitungNilai.nilaiTugas(tugas1, tugas2);
33
34         double na = hitungNilai.nilaiAkhir(tugas, mid, uas);
35
36         JOptionPane.showMessageDialog(null, "Nama: " + mhs.getNama() + "\nStambuk: " + mhs.getStambuk());
37
38         JOptionPane.showMessageDialog(null, "Tugas: " + tugas + "\nMid: " + mid + "\nUAS: " + uas + "\nNilai Akhir: " + na);
39     }
40 }
```

➤ Output

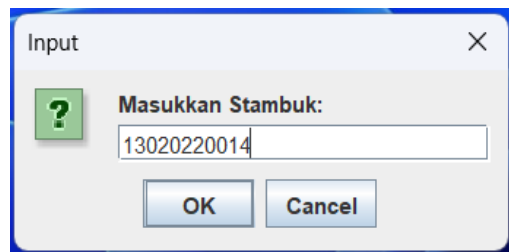


Input

Masukkan Nama:

Raihan Apriansyah

OK Cancel

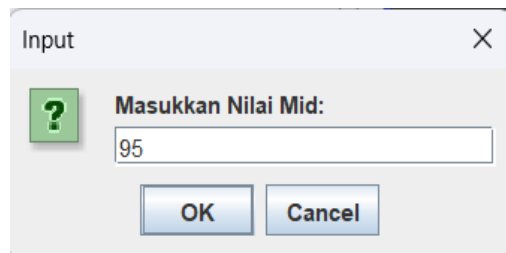


Input

Masukkan Stambuk:

13020220014

OK Cancel

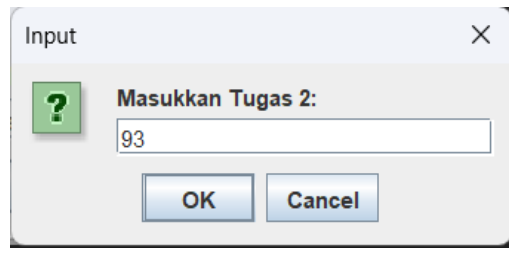


Input

Masukkan Nilai Mid:

95

OK Cancel

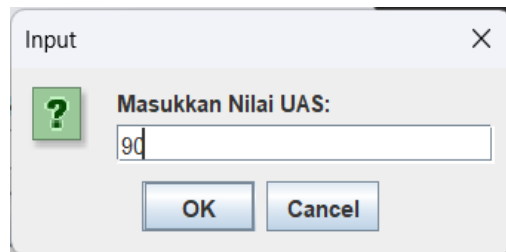


Input

Masukkan Tugas 2:

93

OK Cancel

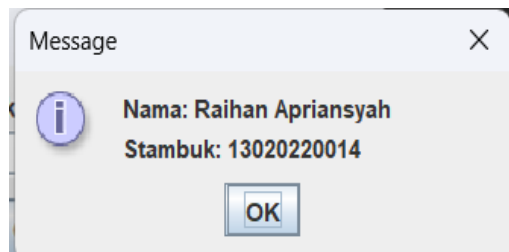


Input

Masukkan Nilai UAS:

90

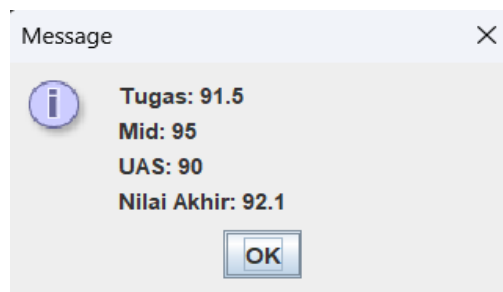
OK Cancel



Message

Nama: Raihan Apriansyah
Stambuk: 13020220014

OK



Message

Tugas: 91.5
Mid: 95
UAS: 90
Nilai Akhir: 92.1

OK