

TUGAS 4

PEMROGRAMAN BERORIENTASI OBJEK

NAMA : ACHMAD ZULFIKAR

STAMBUK : 13020220007

KELAS : A1

EVALUASI STRUKTUR KONTROL & ARRAY

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

- **if-else:**

- Dapat menangani berbagai kondisi yang kompleks dan ekspresi yang menghasilkan nilai boolean.
- Memungkinkan penggunaan operator logika seperti && (dan), || (atau), dan ! (tidak).
- Lebih fleksibel karena dapat memiliki banyak kondisi dan blok else if.

Contoh penggunaan:

```
if (kondisi1) {  
    // blok kode jika kondisi1 benar  
} else if (kondisi2) {  
    // blok kode jika kondisi2 benar  
} else {  
    // blok kode jika semua kondisi di atas salah
```

- **switch-case:**

- Ideal untuk situasi di mana satu variabel atau ekspresi dibandingkan dengan serangkaian nilai konstan.
- Lebih terstruktur dan mudah dibaca ketika ada banyak nilai yang harus dibandingkan dengan variabel yang sama.
- Tidak mendukung operator logika.
- Memerlukan break untuk mencegah eksekusi dari “jatuh” ke kasus berikutnya.

Contoh penggunaan:

```
switch (variabel) {  
    case nilai1:  
        // blok kode untuk nilai1  
        break;  
    case nilai2:  
        // blok kode untuk nilai2
```

```

        break;
    default:
        // blok kode jika tidak ada kasus yang cocok
    }

```

2. Kapan digunakan struktur kontrol if-else dan switch-case ?
Struktur kontrol **if-else** digunakan ketika kondisi harus dievaluasi secara berurutan. Jika kondisi terpenuhi, kode dalam blok **if** dieksekusi; jika tidak, blok **else** dieksekusi. **Switch-case** digunakan ketika banyak kondisi harus dievaluasi berdasarkan nilai variabel tertentu. Ini cocok untuk memilih tindakan berdasarkan nilai variabel, dengan setiap kasus mengevaluasi nilai variabel untuk tindakan yang sesuai.

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

```

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("Pilih opsi:\n1. Input Nilai Baru\n2. Input Nilai Baru\nPilihan Anda: ");
        int pilihan = input.nextInt();

```

```

switch(pilihan) {
    case 1:
        System.out.print("Masukkan Nilai Baru: ");
        hitung.inputNilai(nilai);
        System.out.print("Daftar Nilai : ");
        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. Apakah perbedaan antara struktur kontrol perulangan while dan do-while?

- **While Loop:**

Pada struktur while, kondisi dievaluasi sebelum iterasi loop pertama. Artinya, jika kondisi awalnya salah, maka blok kode dalam loop tidak akan pernah dieksekusi.

Jika kondisi terpenuhi, blok kode dalam loop akan dieksekusi. Setelah itu, kondisi akan dievaluasi lagi. Jika kondisi masih terpenuhi, iterasi akan terus berlanjut.

Jika kondisi menjadi salah, loop berakhir.

Contoh:

```

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

```

- **Do-While Loop:**

Pada struktur do-while, blok kode dalam loop dieksekusi sekali sebelum kondisi dievaluasi. Ini berarti setidaknya satu iterasi akan selalu terjadi.

Setelah blok kode dalam loop dieksekusi, kondisi akan dievaluasi. Jika kondisi terpenuhi, iterasi akan terus berlanjut. Jika kondisi tidak terpenuhi, loop berakhir.

Contoh:

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

5. Kapan digunakan struktur kontrol for?

Struktur kontrol for digunakan ketika perlu dilakukan iterasi (pengulangan) sejumlah tertentu kali atau sesuai dengan kondisi tertentu dengan langkah pengulangan yang sudah ditentukan. Umumnya, for digunakan ketika diketahui berapa kali ingin dilakukan iterasi.

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

- **Array:**

- Array adalah struktur data yang terdiri dari sejumlah elemen dengan tipe data yang sama yang disimpan dalam blok memori yang berurutan.
- Ukuran array ditentukan saat deklarasi dan tidak dapat berubah setelahnya.
- Array dapat diakses menggunakan indeks numerik, dimulai dari 0.
- Untuk array primitif, elemen-elemen array diberi nilai default (misalnya, 0 untuk int, null untuk objek), sedangkan untuk array objek, elemen-elemen awalnya adalah null.

Contoh:

```
// Deklarasi dan inisialisasi array dengan 5 elemen  
int[] numbers = new int[5];  
numbers[0] = 1;  
numbers[1] = 2;  
numbers[2] = 3 ;  
// ..
```

- **ArrayList:**

- ArrayList adalah kelas dalam Java yang menyediakan implementasi dari struktur data dinamis yang dapat digunakan untuk menyimpan kumpulan elemen.
- Ukuran ArrayList dapat berubah secara dinamis saat elemen ditambahkan atau dihapus.
- ArrayList menyediakan metode untuk menambah, menghapus, mencari, dan mengakses elemen dengan mudah.
- ArrayList hanya dapat menyimpan objek (tidak dapat menyimpan tipe data primitif), dan elemen-elemen objeknya tidak perlu memiliki tipe data yang sama.

Contoh:

```
import java.util.ArrayList;
// Deklarasi dan inisialisasi ArrayList
ArrayList<Integer> numbersList = new ArrayList<>();
numbersList.add(1);
numbersList.add(2);
numbersList.add(3);
// ...
```

7. Buatlah contoh program 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) {
        // Membuat objek Scanner untuk mendapatkan input dari pengguna
        Scanner scanner = new Scanner(System.in);

        // Membuat objek HashMap
        Map<String, Integer> hashMap = new HashMap<>();

        // Meminta pengguna untuk memasukkan pasangan kunci-nilai
        System.out.println("Masukkan pasangan kunci-nilai (tekan Enter
setelah setiap pasangan, ketik 'selesai' untuk selesai):");
        String inputKey;
        while (true) {
```

```

        System.out.print("Kunci: ");
        inputKey = scanner.nextLine();

        // Jika pengguna memasukkan 'selesai', keluar dari loop
        if (inputKey.equalsIgnoreCase("selesai")) {
            break;
        }

        // Meminta pengguna untuk memasukkan nilai yang sesuai dengan
        kunci
        System.out.print("Nilai: ");
        int inputValue = scanner.nextInt();
        scanner.nextLine(); // Membersihkan newline setelah nextInt()

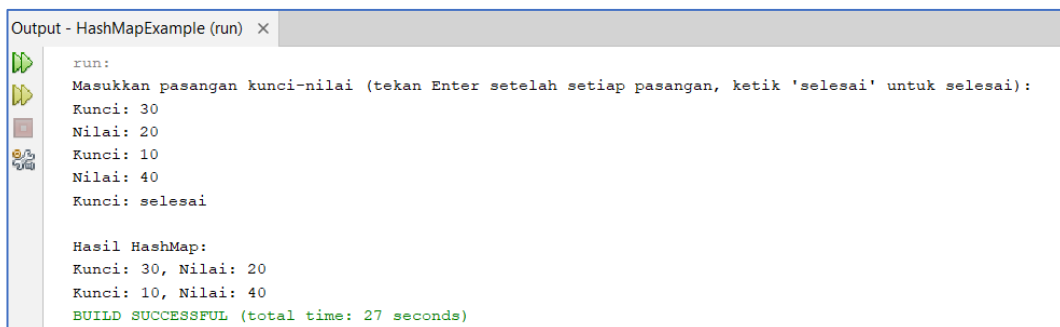
        // Menambahkan pasangan kunci-nilai ke dalam HashMap
        hashMap.put(inputKey, inputValue);
    }

    // Menampilkan hasil HashMap setelah input selesai
    System.out.println("\nHasil HashMap:");
    for (Map.Entry<String, Integer> entry : hashMap.entrySet()) {
        System.out.println("Kunci: " + entry.getKey() + ", Nilai: " +
            entry.getValue());
    }

    // Menutup objek Scanner
    scanner.close();
}
}

```

- Output :



```

Output - HashMapExample (run) x
run:
Masukkan pasangan kunci-nilai (tekan Enter setelah setiap pasangan, ketik 'selesai' untuk selesai):
Kunci: 30
Nilai: 20
Kunci: 10
Nilai: 40
Kunci: selesai

Hasil HashMap:
Kunci: 30, Nilai: 20
Kunci: 10, Nilai: 40
BUILD SUCCESSFUL (total time: 27 seconds)

```

EVALUASI KONSEP PEMROGRAMAN BERBASIS OBJEK

NAMA : ACHMAD ZULFIKAR

STAMBUK : 13020220007

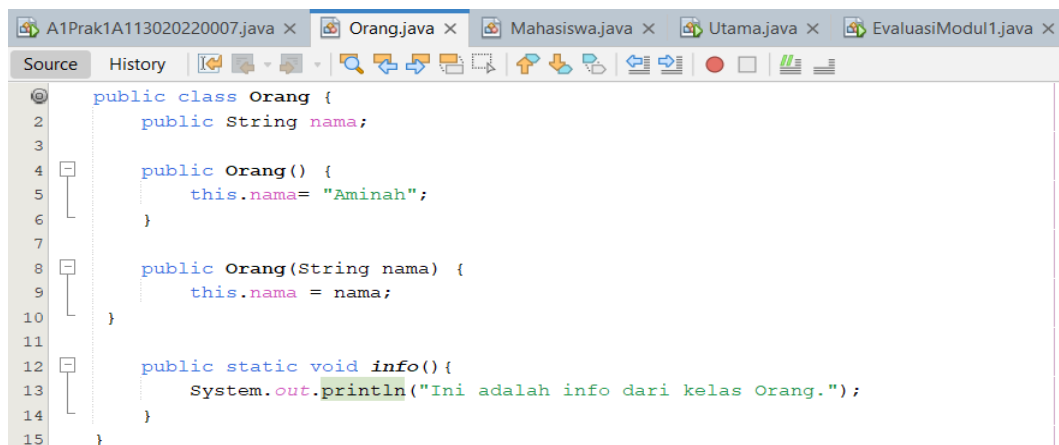
KELAS : A1

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 tersebut, konsep pewarisan dan polimorfisme terdapat pada kelas Mahasiswa. Pewarisan terjadi karena kelas Mahasiswa adalah subkelas dari kelas Orang, yang berarti semua atribut dan metode dari kelas Orang diwarisi oleh kelas Mahasiswa. Dalam konstruktor tanpa parameter di kelas Mahasiswa, digunakan pemanggilan `super()` untuk merujuk konstruktor dari kelas induk (Orang), sehingga konstruktor kelas induk dieksekusi sebelum konstruktor kelas Mahasiswa. Dengan demikian, kelas Mahasiswa memperoleh semua karakteristik dari kelas Orang dan menambahkan atribut stb.

Polimorfisme terjadi dengan adanya dua versi konstruktor di kelas Mahasiswa: satu tanpa parameter dan satu dengan dua parameter (nama dan stb). Polimorfisme memungkinkan penggunaan metode dengan nama yang sama di kelas yang berbeda. Dalam kasus ini, konstruktor `Mahasiswa()` tanpa parameter adalah contoh polimorfisme, karena meskipun diwarisi dari kelas Orang, kelas Mahasiswa juga memiliki versi konstruktor tambahan yang memungkinkan inisialisasi atribut nama dan stb secara bersamaan.

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



```
A1Prak1A113020220007.java x Orang.java x Mahasiswa.java x Utama.java x EvaluasiModul1.java x
Source History
1 public class Orang {
2     public String nama;
3
4     public Orang() {
5         this.nama = "Aminah";
6     }
7
8     public Orang(String nama) {
9         this.nama = nama;
10    }
11
12    public static void info() {
13        System.out.println("Ini adalah info dari kelas Orang.");
14    }
15 }
```

The screenshot shows an IDE with two open Java files. The first file, `Mahasiswa.java`, defines a class `Mahasiswa` that extends `Orang`. It has a private attribute `stb` and two constructors: a no-argument constructor that calls `super()` and initializes `stb` to "1302002134", and a parameterized constructor that takes `stb` and `nama` as arguments. There is also a static method `info()` that prints a message. The second file, `Utama.java`, contains a `main` method that calls `Orang.info()` to demonstrate the inheritance.

```
1 public class Mahasiswa extends Orang {
2     private String stb;
3
4     public Mahasiswa() {
5         super();
6         this.stb = "1302002134"; //stambuk anda
7     }
8
9     public Mahasiswa(String stb, String nama) {
10        this.nama = nama;
11        this.stb = stb;
12    }
13
14    public static void info() {
15        System.out.println("Ini adalah info dari kelas Mahasiswa");
16    }
17 }

1 public class Utama {
2     public static void main(String[] args) {
3         Orang.info(); // Pemanggilan method info() dari kelas orang
4     }
5 }
6 }
```

3. pemanggilan method `info()` pada program utama (Class utama)! 3. Buatlah sebuah project dengan nama project stambuk anda dan buatlah pengorganisasian package dan class seperti berikut.



- ❖ Setelah mengerjakan soal nomor 3, Lengkapi Program berikut :


```
...va  Utamajava x  EvaluasiModul1.java x  InputOutput.java x  HashMapExample.java x  HitungNilaiAkhir.java x  Identitas.java x  Nilai.java

Source  History  [Icons]

1  package Evaluasi.Mahasiswa;
2
3  /**
4   * @author stambuk : 13020220007
5   * Nama : ACHMAD ZULFIKAR
6   * MATERI : MODUL 4, KONSEP PEMROGRAMAN BERBASIS OBJEK
7   * WAKTU : RABU , 27-MARET-2024
8   */
9
10 public class Identitas {
11     private String nama;
12     private String stambuk;
13
14     // Constructor
15     public Identitas(String nama, String stambuk) {
16         this.nama = nama;
17         this.stambuk = stambuk;
18     }
19
20     // Getter untuk nama
21     public String getNama() {
22         return nama;
23     }
24
25     // Setter untuk nama
26     public void setNama(String nama) {
27         this.nama = nama;
28     }
29
30     // Getter untuk stambuk
31     public String getStambuk() {
32         return stambuk;
33     }
34
35     // Setter untuk stambuk
36     public void setStambuk(String stambuk) {
37         this.stambuk = stambuk;
38     }
39 }
```

```
...va  Utamajava x  EvaluasiModul1.java x  InputOutput.java x  HashMapExample.java x  HitungNilaiAkhir.java x  Identitas.java x  Nilai.java x  Utamajava x

Source  History  [Icons]

1  package Evaluasi.Mahasiswa;
2
3  /**
4   * @author stambuk : 13020220007
5   * Nama : ACHMAD ZULFIKAR
6   * MATERI : MODUL 4, KONSEP PEMROGRAMAN BERBASIS OBJEK
7   * WAKTU : RABU , 27-MARET-2024
8   */
9
10 public class Nilai {
11     private int tugas1;
12     private int tugas2;
13     private int mid;
14     private int uas;
15
16     // Constructor
17     public Nilai(int tugas1, int tugas2, int mid, int uas) {
18         this.tugas1 = tugas1;
19         this.tugas2 = tugas2;
20         this.mid = mid;
21         this.uas = uas;
22     }
23
24     // Getter untuk tugas1
25     public int getTugas1() {
26         return tugas1;
27     }
28
29     // Setter untuk tugas1
30     public void setTugas1(int tugas1) {
31         this.tugas1 = tugas1;
32     }
33
34     // Getter untuk tugas2
35     public int getTugas2() {
36         return tugas2;
37     }
38 }
```

```

34 // Getter untuk tugas2
35 public int getTugas2() {
36     return tugas2;
37 }
38
39 // Setter untuk tugas2
40 public void setTugas2(int tugas2) {
41     this.tugas2 = tugas2;
42 }
43
44 // Getter untuk mid
45 public int getMid() {
46     return mid;
47 }
48
49 // Setter untuk mid
50 public void setMid(int mid) {
51     this.mid = mid;
52 }
53
54 // Getter untuk uas
55 public int getUas() {
56     return uas;
57 }
58
59 // Setter untuk uas
60 public void setUas(int uas) {
61     this.uas = uas;
62 }
63 }

```

```

1 /**
2  * @author stambuk : 13020220007
3  * Nama : ACHMAD ZULFIKAR
4  * MATERI : MODUL 4, KONSEP PEMROGRAMAN BERBASIS OBJEK
5  * WAKTU : RABU , 27-MARET-2024
6  */
7
8
9 package Evaluasi.HitungNilai;
10
11 public class HitungNilaiAkhir {
12     public double nilaiTugas(int tugas1, int tugas2){
13         return (tugas1 + tugas2) / 2.0;
14     }
15
16     // method untuk menghitung nilai akhir
17     public double nilaiAkhir(double tugas, int mid, int uas) {
18         return (tugas * 0.4) + (mid * 0.3) + (uas * 0.3);
19     }
20 }

```

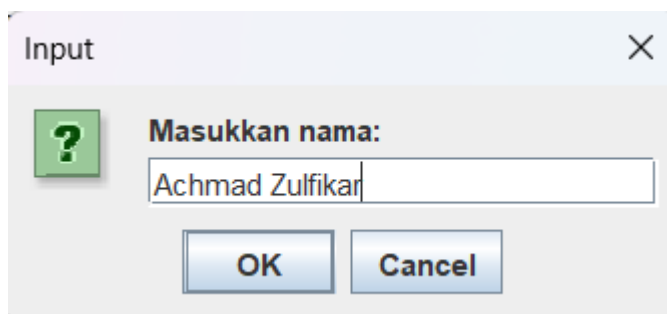
```

1 /**
2  * @author stambuk : 13020220007
3  * Nama : ACHMAD ZULFIKAR
4  * MATERI : MODUL 4, KONSEP PEMROGRAMAN BERBASIS OBJEK
5  * WAKTU : RABU , 27-MARET-2024
6  */
7
8 package Evaluasii;
9
10 import Evaluasi.HitungNilai.HitungNilaiAkhir;
11 import Evaluasi.Mahasiswa.Identitas;
12 import Evaluasi.Mahasiswa.Nilai;
13
14 import java.io.BufferedReader;
15 import java.io.IOException;
16 import java.io.InputStreamReader;
17 import java.util.Scanner;
18
19 import javax.swing.JOptionPane;
20
21 public class Utama {
22     public static void main(String[] args) throws IOException {
23         Scanner scanner = new Scanner(System.in);
24         BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
25
26         // Input nama dan stambuk menggunakan Scanner
27         System.out.print("Masukkan nama: ");
28         String nama = scanner.nextLine();
29         System.out.print("Masukkan stambuk: ");
30         String stambuk = scanner.nextLine();

```

```
...va Utama.java X EvaluasiModul1.java X InputOutput.java X HashMapExample.java X HitungNilaiAkhir.java X Identitas.java X Nilai.java X Utama.java X
Source History
20
21 public class Utama {
22     public static void main(String[] args) throws IOException {
23         Scanner scanner = new Scanner(System.in);
24         BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
25
26         // Input nama dan stambuk menggunakan Scanner
27         System.out.print("Masukkan nama: ");
28         String nama = scanner.nextLine();
29         System.out.print("Masukkan stambuk: ");
30         String stambuk = scanner.nextLine();
31
32         // Input tugas1, tugas2, mid, dan uas menggunakan BufferedReader
33         System.out.print("Masukkan nilai tugas 1: ");
34         int tugas1 = Integer.parseInt(reader.readLine());
35         System.out.print("Masukkan nilai tugas 2: ");
36         int tugas2 = Integer.parseInt(reader.readLine());
37         System.out.print("Masukkan nilai mid: ");
38         int mid = Integer.parseInt(reader.readLine());
39         System.out.print("Masukkan nilai uas: ");
40         int uas = Integer.parseInt(reader.readLine());
41
42         // Inisialisasi objek Identitas dan Nilai
43         Identitas identitas = new Identitas(nama, stambuk);
44         Nilai nilai = new Nilai(tugas1, tugas2, mid, uas);
45
46         // Menghitung nilai tugas dan nilai akhir
47         HitungNilaiAkhir hitungNilaiAkhir = new HitungNilaiAkhir();
48         double tugas = hitungNilaiAkhir.nilaiTugas(tugas1, tugas2);
49         double na = hitungNilaiAkhir.nilaiAkhir(tugas, mid, uas);
50
51         // Menampilkan hasil menggunakan JOptionPane
52         JOptionPane.showMessageDialog(null, "Nama: " + identitas.getNama() + "\nStambuk: " + identitas.getStambuk());
53         JOptionPane.showMessageDialog(null, "Nilai tugas: " + tugas + "\nNilai akhir: " + na);
54     }
55 }
```

- Output :

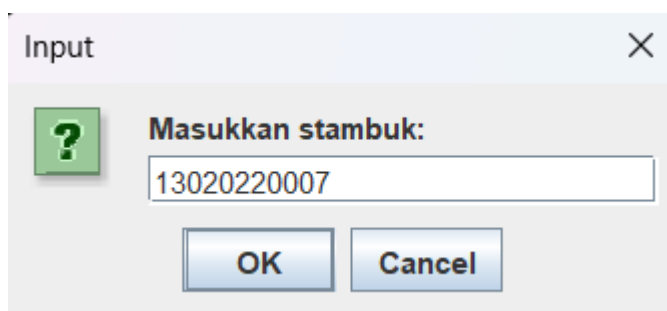


Input

Masukkan nama:

Achmad Zulfikar

OK Cancel

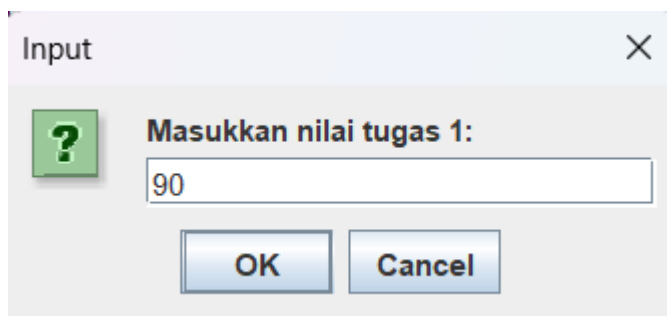


Input

Masukkan stambuk:

13020220007

OK Cancel




Input

Masukkan nilai tugas 1:


90

OK Cancel


Input ×

 **Masukkan nilai tugas 2:**


Input ×

 **Masukkan nilai mid:**


Input ×

 **Masukkan nilai uas:**

Message ×

 **Nama: Achmad Zulfikar**
Stambuk: 13020220007

Message ×

 **Nilai tugas: 87.5**
Nilai akhir: 91.4