

Nama : Yudhistira Putra Hartanto

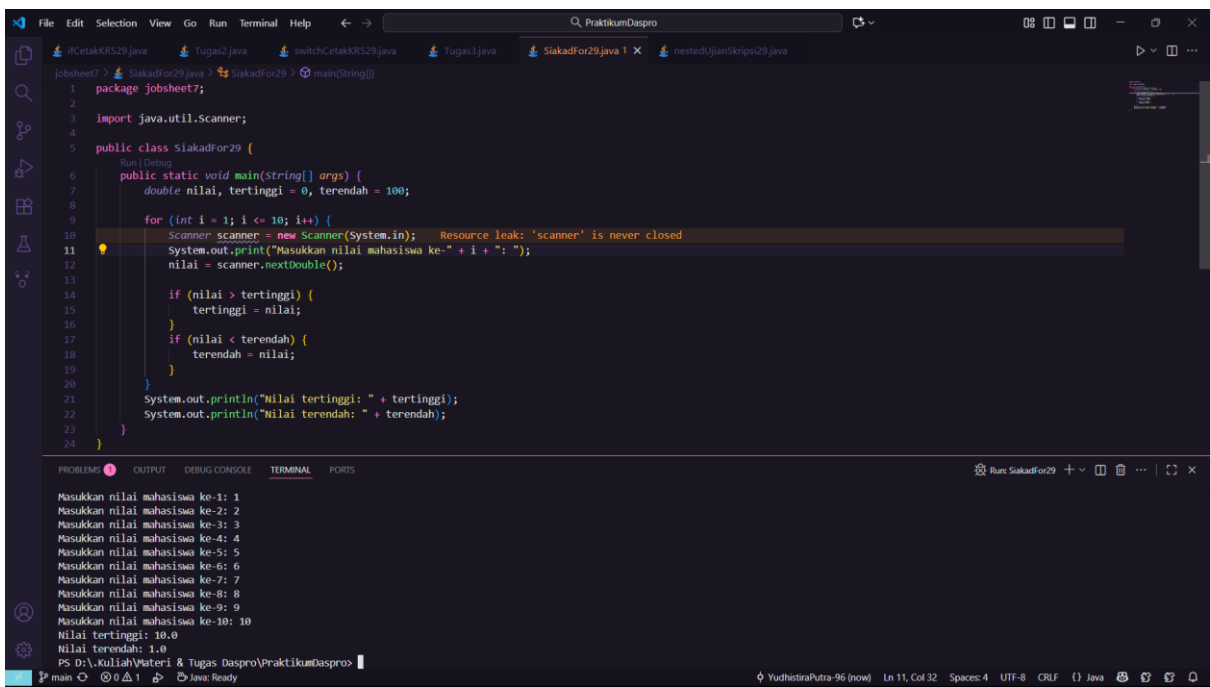
Kelas/Absen : 1G/29

NIM : 254107020083

Prodi : D-IV Teknik Informatika

Praktikum

1. Percobaan 1



Pertanyaan :

1. Sebutkan dan tunjukkan masing-masing komponen perulangan FOR pada kode program Percobaan 1!

2. Mengapa variabel tertinggi diinisialisasi 0 dan terendah diinisialisasi 100? Apa yang terjadi jika variabel tertinggi diinisialisasi 100 dan terendah diinisialisasi 0?

3. Jelaskan fungsi dan alur kerja dari potongan kode berikut!

```

if (nilai > tertinggi) {
    tertinggi = nilai;
}

if (nilai < terendah) {
    terendah = nilai;
}

```

4. Modifikasi kode program sehingga terdapat perhitungan untuk menentukan berapa mahasiswa yang lulus dan yang tidak lulus berdasarkan batas kelulusan (nilai minimal 60). Tampilkan jumlah mahasiswa lulus dan tidak lulus setelah menampilkan nilai tertinggi dan terendah!

5. Commit dan push hasil modifikasi Anda ke Github dengan pesan “Modifikasi Percobaan 1”

Jawaban :

1.

- Inisialisasi: `int i = 1`  
Inisialisasi ini untuk mendeklarasikan dan menginisialisasi variabel `i` dengan nilai 1.
- Kondisi: `i <= 10`  
Kondisi ini mengevaluasi apakah nilai `i` kurang dari atau sama dengan 10, jika TRUE, blok perulangan dieksekusi sedangkan jika FALSE, perulangan berhenti.
- Increment: `i++`  
Menambah nilai `i` sebanyak 1 setiap kali iterasi atau proses pengulangan selesai. Dimana setara dengan `i = i + 1`

2. Variabel tertinggi diinisialisasi dengan 0 agar angka yang kita input akan lebih selalu lebih besar dari 0 dan variabel terendah diinisialisasikan dengan 100 agar angka yang kita input pasti akan lebih kecil dari 100. Sehingga Inisialisasi terbalik akan menghasilkan output yang salah karena tidak ada nilai yang bisa melebihi 100 atau kurang dari 0.

3.

- Fungsi:

Blok kode ini berfungsi untuk mencari dan menyimpan nilai tertinggi serta terendah dari semua nilai yang diinputkan

- Alur Kerja:

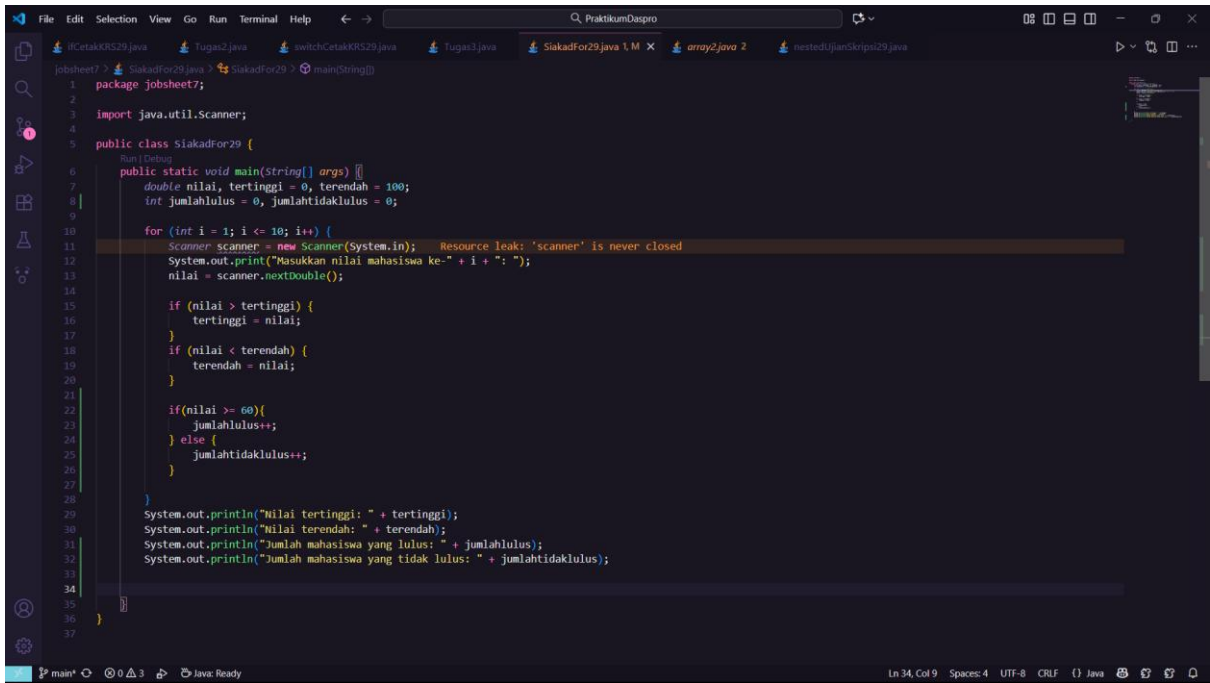
- Pengecekan Nilai Tertinggi:

if (`nilai > tertinggi`) maka akan memeriksa apakah nilai yang baru diinput lebih besar dari nilai tertinggi sementara. Jika YA, maka (`tertinggi = nilai`) akan mengupdate variabel tertinggi dengan nilai yang baru. Jika TIDAK, maka akan dilewati dan pertahankan nilai tertinggi sebelumnya

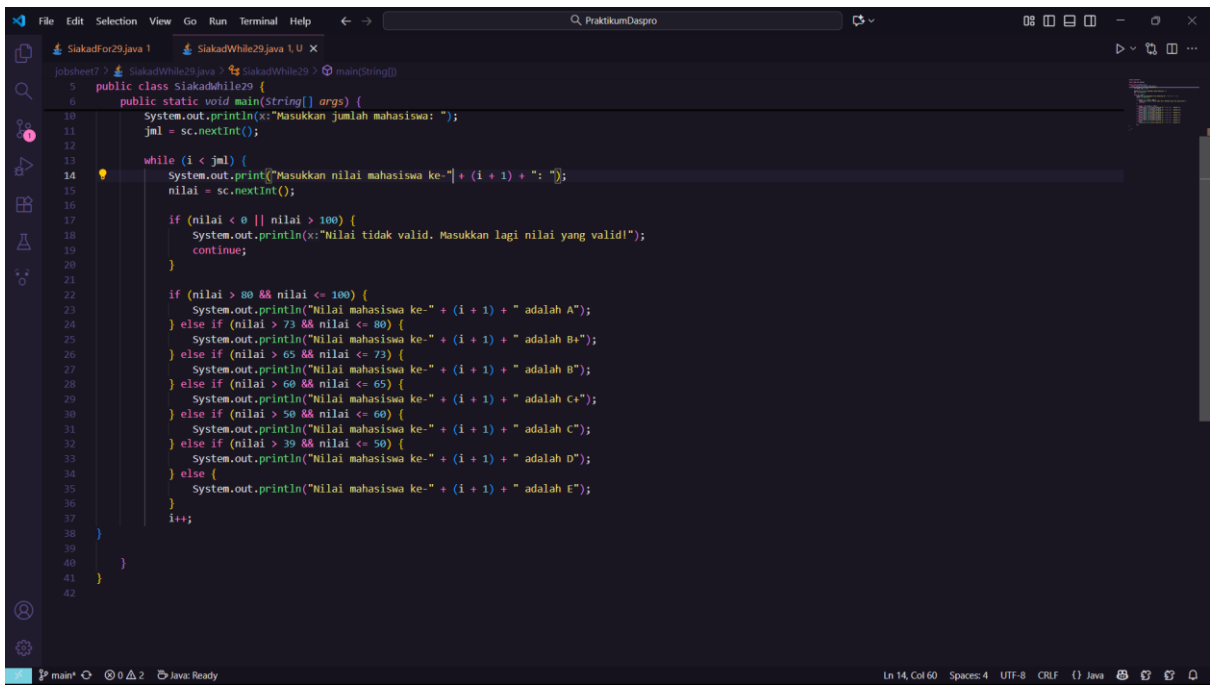
- Pengecekan Nilai Terendah:

if (`nilai < terendah`) maka akan memeriksa apakah nilai yang baru diinput lebih kecil dari nilai terendah sementara. Jika YA, maka (`terendah = nilai`) akan mengupdate variabel terendah dengan nilai yang baru. Jika TIDAK, maka akan dilewati dan pertahankan nilai terendah sebelumnya

4.



## 2. Percobaan 2



Pertanyaan :

1. Pada potongan kode berikut, tentukan maksud dan kegunaan dari sintaks berikut:

```
if (nilai < 0 || nilai > 100) {
    System.out.println(x:"Nilai tidak valid. Masukkan lagi nilai yang valid!");
    continue;
}
```

a. `nilai < 0 || nilai > 100`

b. `continue`

2. Mengapa sintaks `i++` dituliskan di akhir perulangan WHILE? Apa yang terjadi jika posisinya dituliskan di awal perulangan WHILE?

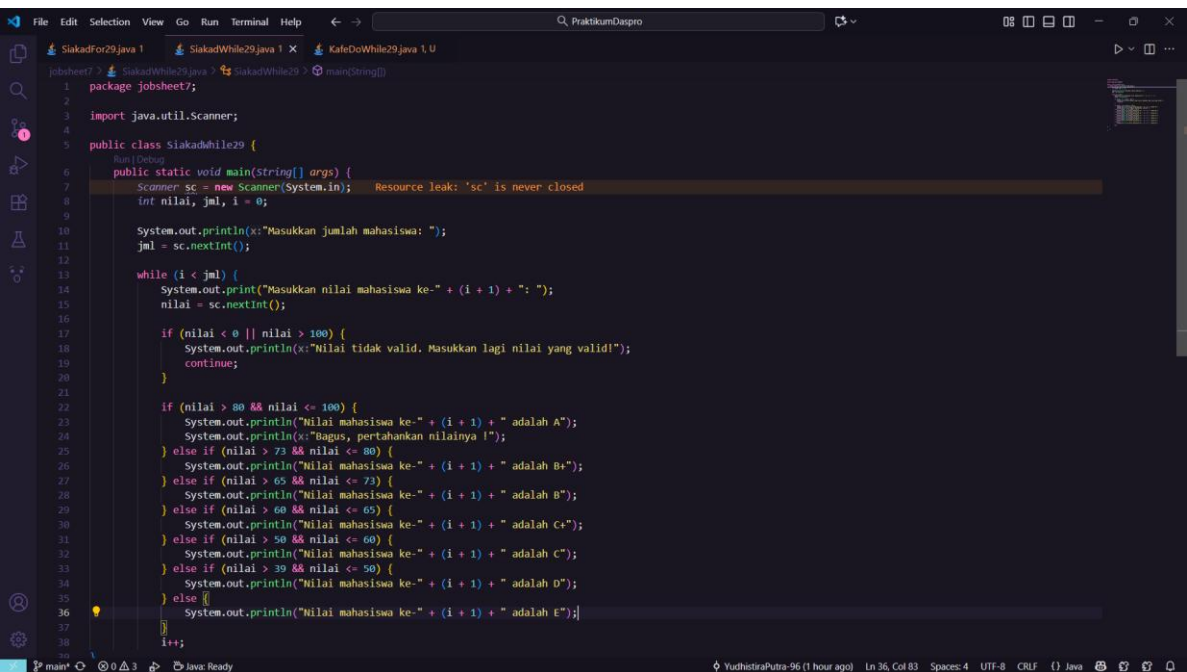
3. Apabila jumlah mahasiswa yang dimasukkan adalah 19, berapa kali perulangan WHILE akan berjalan?
4. Modifikasi kode program sehingga apabila terdapat mahasiswa yang mendapat nilai A, program menampilkan pesan tambahan "Bagus, pertahankan nilainya"!
5. Commit dan push hasil modifikasi Anda ke Github dengan pesan “Modifikasi Percobaan 2”

Jawaban :

- 1.
- `nilai < 0 || nilai > 100`
    - Maksud  
Ini adalah kondisi yang memeriksa apakah nilai yang diinput berada di luar rentang yang valid, atau bisa dibilang untuk mengecek apakah nilai yang diinputkan itu lebih kecil dari 0 ATAU lebih besar dari 100.
    - Kegunaan  
Sebagai atau untuk validasi input supaya memastikan nilai berada antara 0-100.
  - Continue
    - Maksud  
Keyword untuk melanjutkan ke iterasi berikutnya tanpa mengeksekusi kode setelahnya atau bisa dibilang menskip kode setelahnya agar tidak dieksekusi.
    - Kegunaan  
Melewati sisa kode dalam perulangan saat ini dan langsung melanjutkan ke iterasi berikutnya.
2. `i++` berfungsi sebagai counter untuk mencatat berapa banyak mahasiswa yang sudah diproses. Dengan menempatkannya di akhir, setiap mahasiswa yang berhasil diproses (nilai valid) akan dihitung. Dan jika nilai tidak valid (karena continue), `i++` tidak dieksekusi sehingga mahasiswa yang sama akan diulang.

3. 19 kali, dan bisa lebih jika ada input yang tidak valid

4.



```
package jobsheet7;

import java.util.Scanner;

public class SiakadWhile29 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int nilai, jml, i = 0;

        System.out.println("Masukkan jumlah mahasiswa: ");
        jml = sc.nextInt();

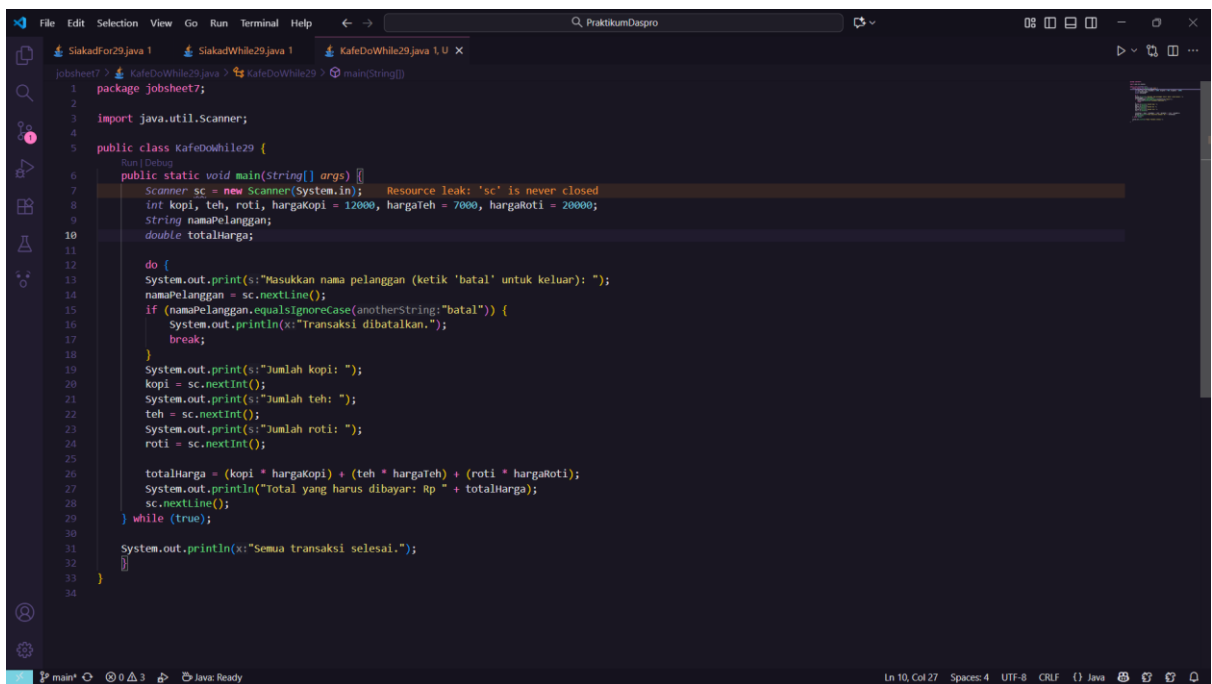
        while (i < jml) {
            System.out.print("Masukkan nilai mahasiswa ke-" + (i + 1) + ": ");
            nilai = sc.nextInt();

            if (nilai < 0 || nilai > 100) {
                System.out.println("Nilai tidak valid. Masukkan lagi nilai yang valid!");
                continue;
            }

            if (nilai > 80 && nilai <= 100) {
                System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah A");
                System.out.println("Bagus, pertahankan nilainya!");
            } else if (nilai > 73 && nilai <= 80) {
                System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah B+");
            } else if (nilai > 65 && nilai <= 73) {
                System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah B");
            } else if (nilai > 60 && nilai <= 65) {
                System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah C+");
            } else if (nilai > 50 && nilai <= 60) {
                System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah C");
            } else if (nilai > 39 && nilai <= 50) {
                System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah D");
            } else {
                System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah E");
            }

            i++;
        }
    }
}
```

### 3. Percobaan 3



```
1 package jobsheet7;
2
3 import java.util.Scanner;
4
5 public class KafeDoWhile29 {
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         int kopi, teh, roti, hargaKopi = 12000, hargaTeh = 7000, hargaRoti = 20000;
9         String namaPelanggan;
10        double totalHarga;
11
12        do {
13            System.out.print("Masukkan nama pelanggan (ketik 'batal' untuk keluar): ");
14            namaPelanggan = sc.nextLine();
15            if (namaPelanggan.equalsIgnoreCase("batal")) {
16                System.out.println("Transaksi dibatalkan.");
17                break;
18            }
19            System.out.print("Jumlah kopi: ");
20            kopi = sc.nextInt();
21            System.out.print("Jumlah teh: ");
22            teh = sc.nextInt();
23            System.out.print("Jumlah roti: ");
24            roti = sc.nextInt();
25
26            totalHarga = (kopi * hargaKopi) + (teh * hargaTeh) + (roti * hargaRoti);
27            System.out.println("Total yang harus dibayar: Rp " + totalHarga);
28            sc.nextLine();
29        } while (true);
30
31        System.out.println("Semua transaksi selesai.");
32    }
33 }
34
```

- Pertanyaan :
1. Pada penggunaan DO-WHILE ini, apabila nama pelanggan yang dimasukkan pertama kali adalah “batal”, maka berapa kali perulangan dilakukan?
  2. Sebutkan kondisi berhenti yang digunakan pada perulangan DO-WHILE tersebut!
  3. Apa fungsi dari penggunaan nilai true pada kondisi DO-WHILE?
  4. Mengapa perulangan DO-WHILE tersebut tetap berjalan meskipun tidak ada komponen inisialisasi dan update?

- Jawaban :
1. 1 kali perulangan karena perulangan DO-WHILE selalu mengeksekusi minimal 1 kali, bahkan jika kondisi berhenti terpenuhi.
  2. Ketika user menginput nama pelanggan dengan teks "batal" (case-insensitive atau menghiraukan besar-kecilnya huruf).
  3. Membuat perulangan infinite loop (perulangan tak terbatas) atau dengan nilai true maka perulangan akan terus berjalan, karena keadaannya True(benar) hingga beberapa kondisi menghentikannya seperti ketika true diubah menjadi false atau menggunakan break.
  4. Karena struktur DO-WHILE memiliki karakteristik khusus, seperti DO-WHILE selalu mengeksekusi blok kode minimal sekali sebelum mengecek kondisi, while(true) membuat perulangan tidak pernah berhenti secara natural, dan program ini tidak menghitung jumlah iterasi, tetapi berjalan berdasarkan input user. Sehingga penghentian dilakukan dengan break statement, bukan melalui kondisi perulangan.