

## JOBSHEET 7

### PERULANGAN 1

#### 1. Tujuan

- Mahasiswa dapat menjelaskan format penulisan program perulangan (for, while, dan do-while)
- Mahasiswa dapat mengimplementasikan flowchart perulangan menggunakan bahasa pemrograman Java

#### 2. Praktikum

##### 2.1 Percobaan 1: Studi Kasus Nilai Mahasiswa di SIAKAD – Perulangan FOR

Waktu Percobaan: 90 menit

Di dalam Sistem Informasi Akademik (SIKAD), dosen mengisi nilai mata kuliah Praktikum Dasar Pemrograman yang ditempuh oleh mahasiswa. Dosen tersebut ingin mencari nilai tertinggi dan terendah Kuis dari 10 mahasiswa di dalam satu kelas. Dosen tersebut harus memasukkan nilai dari setiap siswa, kemudian menentukan dan menampilkan nilai tertinggi dan terendah.

##### 2.1.1 Langkah-langkah Percobaan

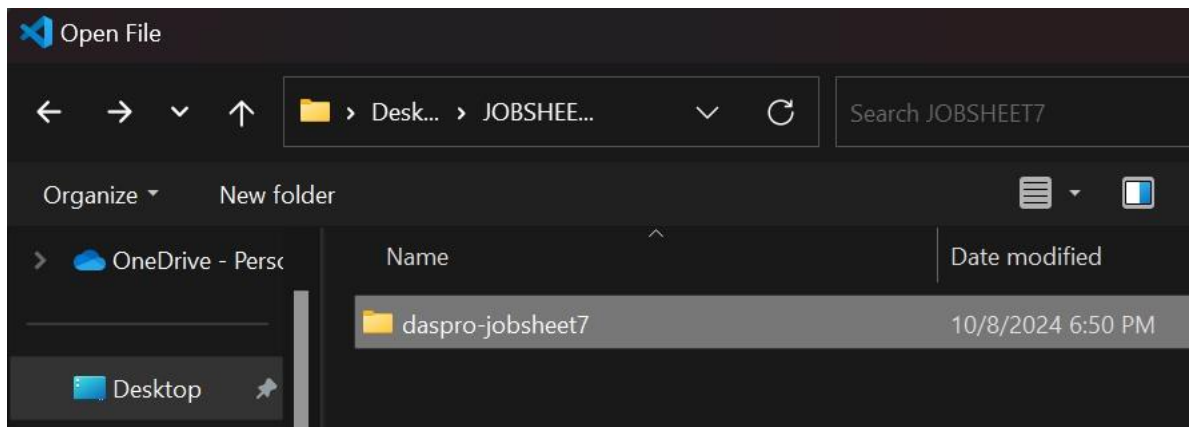
1. Buat repository baru pada akun Github Anda, beri nama daspro-jobsheet7



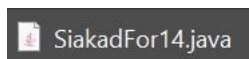
2. Lakukan cloning repository tersebut menggunakan perintah git clone dari terminal

```
C:\Users\WINDOWS 11\Desktop\JOBSHEET7>git clone https://github.com/adamarsyadfaizin/daspro-jobsheet7.git
Cloning into 'daspro-jobsheet7'...
warning: You appear to have cloned an empty repository.
```

3. Buka folder repository tersebut menggunakan Visual Studio Code



4. Buat file baru, beri nama SiakadForNoAbsen.java



5. Buatlah struktur dasar program Java yang terdiri dari fungsi main().

```
public class SiakadFor14 {  
    Run | Debug  
    public static void main(String[] args) {  
    }  
}
```

6. Tambahkan library Scanner di bagian atas (luar) class

```
import java.util.Scanner;
```

7. Buat deklarasi Scanner dengan nama variabel sc di dalam fungsi main()

```
Run | Debug  
public static void main(String[] args) {  
    Scanner sc = new Scanner(System.in);  
}
```

8. Deklarasikan variabel nilai, tertinggi, dan terendah bertipe double. Inisialisasi tertinggi dengan 0 dan terendah dengan 100

```
double nilai, tertinggi = 0, terendah = 100;
```

9. Buat struktur perulangan FOR dengan batas kondisi sesuai jumlah mahasiswa yaitu 10

```
for (int i = 1; i <= 10; i++) {  
}
```

10. Di dalam perulangan FOR tersebut, tambahkan perintah untuk memasukkan nilai mahasiswa. Setelah itu, buat dua kondisi pemilihan secara terpisah untuk mengecek nilai tertinggi dan terendah dengan membandingkan nilai masukan dengan variabel tertinggi dan variabel terendah

```
for (int i = 1; i <= 10; i++) {  
    System.out.print("Masukkan nilai mahasiswa ke-" + i + ": ");  
    nilai = sc.nextDouble();  
    if (nilai > tertinggi) {  
        tertinggi = nilai;  
    }  
    if (nilai < terendah) {  
        terendah = nilai;  
    }  
}
```

11. Di luar perulangan FOR, tampilkan nilai tertinggi dan terendah

```
}  
System.out.println("Nilai tertinggi: " + tertinggi);  
System.out.println("Nilai terendah: " + terendah);
```

12. Compile dan run program

```
Masukkan nilai mahasiswa ke-1: 98  
Masukkan nilai mahasiswa ke-2: 60  
Masukkan nilai mahasiswa ke-3: 60  
Masukkan nilai mahasiswa ke-4: 60  
Masukkan nilai mahasiswa ke-5: 40  
Masukkan nilai mahasiswa ke-6: 60  
Masukkan nilai mahasiswa ke-7: 60  
Masukkan nilai mahasiswa ke-8: 60  
Masukkan nilai mahasiswa ke-9: 60  
Masukkan nilai mahasiswa ke-10: 60  
Nilai tertinggi: 98.0  
Nilai terendah: 40.0
```

13. Commit dan push kode program ke Github

SiakadFor14.java	Percobaan_1_14	now
------------------	----------------	-----

## 2.1.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

```
Masukkan nilai mahasiswa ke-1: 76.5
Masukkan nilai mahasiswa ke-2: 82.3
Masukkan nilai mahasiswa ke-3: 62.1
Masukkan nilai mahasiswa ke-4: 88.4
Masukkan nilai mahasiswa ke-5: 65.9
Masukkan nilai mahasiswa ke-6: 67.9
Masukkan nilai mahasiswa ke-7: 90.1
Masukkan nilai mahasiswa ke-8: 55.3
Masukkan nilai mahasiswa ke-9: 73.7
Masukkan nilai mahasiswa ke-10: 78.6
Nilai tertinggi: 90.1
Nilai terendah: 55.3
```

### 2.1.3 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 kode program ke Github

### Jawaban Pertanyaan

1. 1. Inisialisasi (int i = 1) `int i = 1`

2. Kondisi ( i <= 10 ) `i <= 10`

3. Increment/Update (i++) `i++`

4. Blok kode yang diulang

```

System.out.print("Masukkan nilai mahasiswa ke-" + i + ": ");
nilai = sc.nextDouble();
if (nilai > tertinggi) {
    tertinggi = nilai;
}
if (nilai < terendah) {
    terendah = nilai;
}

```

2. Variabel tertinggi diinisialisasi dengan nilai 0 agar ketika membandingkan nilai mahasiswa yang pertama kali diinput, nilai apapun yang lebih besar dari 0 (selama dalam skala normal, misalnya 0 hingga 100) akan dianggap sebagai nilai tertinggi baru. Jika diinisialisasi dengan nilai yang lebih besar, misalnya 100, maka sistem tidak akan menemukan nilai tertinggi dengan benar karena tidak ada nilai yang lebih besar dari 100 dalam kisaran nilai yang diinputkan.

Variabel terendah diinisialisasi dengan nilai 100 untuk alasan yang serupa. Dalam kisaran nilai mahasiswa, 100 adalah nilai maksimum, sehingga nilai mahasiswa pertama yang lebih kecil dari 100 akan langsung menggantikan nilai terendah

YANG TERJADI APABILA TERTINGGI = 100 , TERENDAH = 0

Untuk tertinggi: Tidak akan pernah ada nilai mahasiswa yang lebih besar dari 100 (asumsi nilai dalam skala 0-100), sehingga variabel tertinggi akan tetap bernilai 100, meskipun mungkin nilai mahasiswa yang lebih kecil dari 100 sudah dimasukkan.

Untuk terendah: Nilai mahasiswa biasanya tidak lebih kecil dari 0, sehingga variabel terendah juga akan tetap bernilai 0, meskipun semua nilai mahasiswa mungkin lebih tinggi dari 0.

```

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

```

Bagian ini memeriksa apakah nilai yang diinput oleh pengguna (nilai) lebih besar daripada nilai saat ini yang disimpan dalam variabel tertinggi.

Jika kondisi ini benar (yaitu, nilai yang dimasukkan lebih besar dari tertinggi), maka variabel tertinggi diperbarui dengan nilai baru tersebut.

```

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

```

Bagian ini memeriksa apakah nilai yang diinput oleh pengguna (nilai) lebih kecil daripada nilai saat ini yang disimpan dalam variabel terendah.

Jika kondisi ini benar (yaitu, nilai yang dimasukkan lebih kecil dari terendah), maka variabel terendah diperbarui dengan nilai baru tersebut.

4.

```
for (int i = 1; i <= 10; i++) {  
    System.out.print("Masukkan nilai mahasiswa ke-" + i + ": ");  
    nilai = sc.nextDouble();  
    if (nilai > tertinggi) {  
        tertinggi = nilai;  
    }  
    if (nilai < terendah) {  
        terendah = nilai;  
    }  
    if (nilai >= 60) {  
        lulus++;  
    }  
    if (nilai < 60) {  
        tidakLulus++;  
    }  
}  
  
System.out.println("Nilai tertinggi: " + tertinggi);  
System.out.println("Nilai terendah: " + terendah);  
System.out.println("Jumlah Mahasiswa yang lulus : " + lulus);  
System.out.println("Jumlah Mahasiswa yang tidak lulus : " + tidakLulus);
```

OUTPUT :

```
Masukkan nilai mahasiswa ke-1: 80  
Masukkan nilai mahasiswa ke-2: 60  
Masukkan nilai mahasiswa ke-3: 70  
Masukkan nilai mahasiswa ke-4: 50  
Masukkan nilai mahasiswa ke-5: 40  
Masukkan nilai mahasiswa ke-6: 76  
Masukkan nilai mahasiswa ke-7: 30  
Masukkan nilai mahasiswa ke-8: 50  
Masukkan nilai mahasiswa ke-9: 89  
Masukkan nilai mahasiswa ke-10: 60  
Nilai tertinggi: 89.0  
Nilai terendah: 30.0  
Jumlah Mahasiswa yang lulus : 6.0  
Jumlah Mahasiswa yang tidak lulus : 4.0
```

SiakadFor14Pertanyaan.java

Percobaan\_1\_Pertanyaan\_14

5.

## 2.2 Percobaan 2: Studi Kasus Nilai Mahasiswa di SIAKAD – Perulangan WHILE

Waktu Percobaan: 90 menit

Seorang dosen ingin memasukkan nilai beberapa mahasiswa ke dalam SIAKAD untuk ditentukan kategori nilai hurufnya. Program harus meminta dosen untuk memasukkan nilai setiap mahasiswa. Jika dosen memasukkan nilai yang tidak valid (negatif atau lebih dari 100), program harus mengabaikan input tersebut dan meminta dosen untuk melakukan input ulang. Selanjutnya, nilai yang valid dikelompokkan ke dalam kategori huruf A ( $80 < \text{nilai} \leq 100$ ), B+ ( $73 < \text{nilai} \leq 80$ ), B ( $65 < \text{nilai} \leq 73$ ), C+ ( $60 < \text{nilai} \leq 65$ ), C ( $50 < \text{nilai} \leq 60$ ), D ( $39 < \text{nilai} \leq 50$ ), dan E ( $\text{nilai} \leq 39$ ).

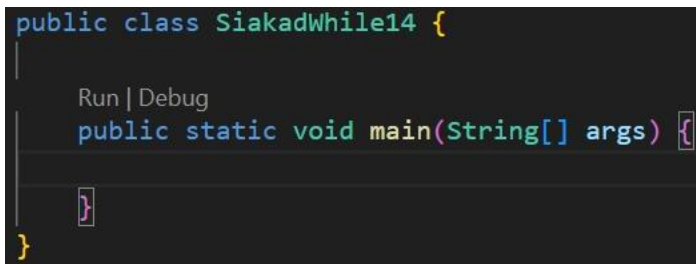
Berdasarkan studi kasus tersebut, buat program menggunakan bahasa pemrograman Java.

### 2.2.1 Langkah-langkah Percobaan

1. Buat file baru, beri nama SiakadWhileNoAbsen.java

A small screenshot showing a file icon and the text "SiakadWhile14.java".

2. Buatlah struktur dasar program Java yang terdiri dari fungsi main().

A screenshot of a code editor showing the basic structure of a Java class. The code is: 

```
public class SiakadWhile14 {  
    Run | Debug  
    public static void main(String[] args) {  
    }  
}
```

3. Tambahkan library Scanner di bagian atas (luar) class

A screenshot of a code editor showing the import statement: 

```
import java.util.Scanner;
```

4. Buat deklarasi Scanner dengan nama variabel sc di dalam fungsi main()

A screenshot of a code editor showing the declaration of a Scanner object: 

```
Scanner sc = new Scanner(System.in);
```

5. Deklarasikan variabel nilai, jml, dan i (untuk perulangan) bertipe integer. Inisialisasi i dengan 0 sebagai nilai awal perulangan

A screenshot of a code editor showing the declaration and initialization of three integer variables: 

```
int nilai, jml, i = 0;
```

6. Tuliskan kode program untuk menerima input banyaknya mahasiswa yang disimpan ke variabel jml. Dengan demikian, batas perulangan akan dinamis sesuai masukan dari



pengguna melalui keyboard.

```
System.out.print(s:"Masukkan jumlah mahasiswa: ");  
jml = sc.nextInt();
```

7. Buat struktur perulangan WHILE dengan batas kondisi sesuai jumlah mahasiswa yaitu

5. Perhatikan simbol yang digunakan adalah < karena perulangan variabel i dimulai

dari 0, bukan 1

```
while (i < jml) {  
    |  
    i++;  
}
```

8. Di dalam perulangan WHILE tersebut, tambahkan perintah untuk memasukkan nilai mahasiswa. Setelah itu, buat kondisi pemilihan IF untuk mengecek valid atau tidaknya nilai yang dimasukkan, dengan syarat nilai harus berada pada rentang 0 hingga 100.

Kemudian tambahkan kondisi pemilihan IF-ELSE IF-ELSE untuk menampilkan kategori nilai huruf berdasarkan ketentuan.



```

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

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

    if (nilai > 80 && nilai <= 100) {
        System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah A");
    } 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++;
}

```

#### 9. Compile dan run program

```

Masukkan jumlah mahasiswa: 3
Masukkan nilai mahasiswa ke-1: 80
Nilai mahasiswa ke-1 adalah B+
Masukkan nilai mahasiswa ke-2: 72
Nilai mahasiswa ke-2 adalah B
Masukkan nilai mahasiswa ke-3: 63
Nilai mahasiswa ke-3 adalah C+

```

#### 10. Commit dan push kode program ke Github

SiakadWhile14.java

Percobaan\_2\_14

### 2.2.2 Langkah-langkah Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

```

Masukkan jumlah mahasiswa: 5
Masukkan nilai mahasiswa ke-1: 85
Nilai mahasiswa ke-1 adalah A
Masukkan nilai mahasiswa ke-2: 63
Nilai mahasiswa ke-2 adalah C+
Masukkan nilai mahasiswa ke-3: 101
Nilai tidak valid. Masukkan lagi nilai yang valid!
Masukkan nilai mahasiswa ke-3: 23
Nilai mahasiswa ke-3 adalah E
Masukkan nilai mahasiswa ke-4: -15
Nilai tidak valid. Masukkan lagi nilai yang valid!
Masukkan nilai mahasiswa ke-4: 70
Nilai mahasiswa ke-4 adalah B
Masukkan nilai mahasiswa ke-5: 55
Nilai mahasiswa ke-5 adalah C

```

### 2.2.3 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 kode program ke Github

### Jawaban Pertanyaan

1. A. Jika nilai kurang dari 0 atau lebih dari 100 maka akan mengeluarkan output "Nilai tidak valid Masukkan lagi nilai yang valid"

B. Dengan menggunakan `continue`, iterasi tidak akan ditingkatkan (`i` tidak akan bertambah) ketika nilai tidak valid, sehingga mahasiswa yang sama diminta untuk memasukkan nilai kembali hingga yang valid diberikan.

2. Sintaks `i++` ditempatkan di akhir perulangan `while` karena tujuan dari variabel `i` adalah untuk menghitung jumlah mahasiswa dan memastikan setiap mahasiswa diminta memasukkan nilai yang benar. Variabel `i` bertindak sebagai counter untuk melacak mahasiswa ke-berapa yang sedang diproses.

Jika `i++` dituliskan di awal perulangan, maka `i` akan bertambah sebelum memproses input nilai mahasiswa contoh output :

Misalkan ada 3 mahasiswa, ketika mahasiswa pertama memasukkan nilai, `i` langsung menjadi 2 sebelum nilai mahasiswa pertama divalidasi, dan proses langsung lompat ke mahasiswa kedua. Mahasiswa pertama mungkin terlewat.

3. 20 kali

```
Masukkan jumlah mahasiswa: 3
Masukkan nilai mahasiswa ke-1: 90
Nilai mahasiswa ke-1 adalah A
Bagus, pertahankan nilainya
```

4. Masukkan nilai mahasiswa ke-2: 50

```
if (nilai > 80 && nilai <= 100) {
    System.out.println("Nilai mahasiswa ke-" + (i + 1) + " adalah A");
    System.out.println(x:"Bagus, pertahankan nilainya");
}
```

5.  SiakadWhile14Pertanyaan.java

Percobaan\_2\_Pertanyaan\_14

## 2.3 Percobaan 3: Studi Kasus Transaksi di Kafe – Perulangan DO-WHILE

Waktu Percobaan: 60 menit

Di sebuah kafe, kasir ingin memproses transaksi beberapa pelanggan. Pelanggan dapat membeli lebih dari satu item (kopi dengan harga Rp 12.000, teh dengan harga Rp 7.000, dan roti dengan harga Rp 20.000), dan kasir akan terus memasukkan jumlah pembelian untuk setiap pelanggan. Jika ada pelanggan yang memutuskan untuk membatalkan transaksi (dengan memasukkan "batal"), maka kasir akan menghentikan input transaksi dan program berhenti.

Berdasarkan studi kasus tersebut, buat program menggunakan bahasa pemrograman Java.

### 2.3.1 Langkah-langkah Percobaan

1. Buat file baru, beri nama `KafeDoWhileNoAbsen.java`

 KafeDoWhile14.java

2. Buatlah struktur dasar program Java yang terdiri dari fungsi main().

```
public class KafeDoWhile14 {  
    Run | Debug  
    public static void main(String[] args) {  
    }  
}
```

3. Tambahkan library Scanner di bagian atas (luar) class

```
import java.util.Scanner;
```

4. Buat deklarasi Scanner dengan nama variabel sc di dalam fungsi main()

```
public static void main(String[] args) {  
    Scanner sc = new Scanner(System.in);  
}
```

5. Deklarasikan variabel kopi, teh, dan roti bertipe integer untuk menampung banyaknya item yang dibeli pelanggan, serta namaPelanggan bertipe String. Deklarasi dan inisialisasi hargaKopi dengan 12000, hargaTeh dengan 7000, hargaRoti dengan 20000.

```
int kopi, teh, roti, hargaKopi = 12000, hargaTeh = 7000, hargaRoti = 20000;  
String namaPelanggan;
```

6. Buat struktur perulangan DO-WHILE dengan kondisi true

```
do {  
    } while (true);
```

7. Di dalam perulangan DO-WHILE tersebut, tambahkan perintah untuk memasukkan namaPelanggan. Kemudian tambahkan kondisi IF untuk mengecek isi variabel namaPelanggan. Selanjutnya, tambahkan perintah untuk memasukkan banyaknya item yang dibeli pelanggan untuk setiap menu, apabila masukan nama pelanggan bukan "batal". Hitung total harga pembelian dan tampilkan hasilnya.

```

do {
    System.out.print(s:"Masukkan nama pelanggan (ketik 'batal' untuk keluar): ");
    namaPelanggan = sc.nextLine();
    if (namaPelanggan.equalsIgnoreCase(anotherString:"batal")) {
        System.out.println(x:"Transaksi dibatalkan");
        break;
    }
    System.out.print(s:"Jumlah kopi: ");
    kopi = sc.nextInt();
    System.out.print(s:"Jumlah teh: ");
    teh = sc.nextInt();
    System.out.print(s:"Jumlah roti: ");
    roti = sc.nextInt();
    int totalHarga = (kopi * hargaKopi) + (teh * hargaTeh) + (roti * hargaRoti);
    System.out.println("Total yang harus dibayar: Rp " + totalHarga);
    sc.nextLine();
} while (true);

System.out.println(x:"Semua transaksi selesai.");

```

8. Compile dan run program

```

Masukkan nama pelanggan (ketik 'batal' untuk keluar): Adam
Jumlah kopi: 3
Jumlah teh: 2
Jumlah roti: 1
Total yang harus dibayar: Rp 70000

```

9. Commit dan push kode program ke Github

KafeDoWhile14.java	Percobaan_3_14	now
--------------------	----------------	-----

### 2.3.2 Langkah-langkah Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

```

Masukkan nama pelanggan (ketik 'batal' untuk keluar): Rena
Jumlah kopi: 3
Jumlah teh: 0
Jumlah roti: 1
Total yang harus dibayar: Rp 56000
Masukkan nama pelanggan (ketik 'batal' untuk keluar): Yuni
Jumlah kopi: 1
Jumlah teh: 4
Jumlah roti: 2
Total yang harus dibayar: Rp 80000
Masukkan nama pelanggan (ketik 'batal' untuk keluar): BATAL
Transaksi dibatalkan
Semua transaksi selesai.

```

### 2.3.3 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 Pertanyaan

1. 1 kali
2. jika memasukkan input batal pada variable namaPelanggan
3. Penggunaan nilai true pada kondisi perulangan do-while dalam kode ini berfungsi untuk membuat loop berjalan tanpa batas (infinite loop), hingga ditemukan kondisi yang secara eksplisit menghentikan perulangan (dalam hal ini, menggunakan break).
4. Perulangan do-while pada kode tersebut tetap berjalan meskipun tidak ada komponen inisialisasi dan update karena perulangan do-while tidak membutuhkan inisialisasi atau update secara eksplisit untuk berjalan. Perulangan ini hanya bergantung pada kondisi boolean di bagian while. Dalam kasus ini, kondisi boolean selalu diatur menjadi true, sehingga loop akan terus berjalan sampai ada intervensi (seperti pernyataan break).

### 3. Tugas

Waktu Percobaan : 120 Menit

1. Seorang pengelola bioskop ingin membuat program untuk menghitung total penjualan tiket dalam satu hari. Tiket dijual dengan harga Rp 50.000 per tiket. Program harus menghitung total tiket yang terjual dan total harga penjualan tiket selama satu hari dengan ketentuan sebagai berikut:
  - Jika pelanggan membeli lebih dari 4 tiket, pelanggan mendapatkan diskon 10%.
  - Jika pelanggan membeli lebih dari 10 tiket, pelanggan mendapatkan diskon 15%.
  - Jika input jumlah tiket tidak valid (negatif), program akan mengabaikan input tersebut dan meminta input ulang.

Catatan: Perulangan dapat menggunakan for, while, atau do-while. Penambahan break atau continue jika diperlukan

2. Perhatikan flowchart berikut!



Sebuah tempat parkir ingin membuat program untuk menghitung total pembayaran parkir dari beberapa kendaraan. Tarif parkir adalah Rp 3.000 per jam untuk mobil dan Rp 2.000 per jam untuk motor. Namun, jika durasi parkir lebih dari 5 jam, diberikan tarif tetap sebesar Rp 12.500 untuk semua kendaraan. Program akan terus meminta masukan selama input bukan 0. Implementasikan flowchart tersebut ke dalam bentuk kode program Java!

Jawaban Tugas

```
Masukkan jumlah pelanggan hari ini : 4
Masukkan jumlah tiket yang dibeli oleh pelanggan ke-0: 5
Dapat diskon 10%
Masukkan jumlah tiket yang dibeli oleh pelanggan ke-1: 12
Dapat diskon 15%
Masukkan jumlah tiket yang dibeli oleh pelanggan ke-2: 2
Masukkan jumlah tiket yang dibeli oleh pelanggan ke-3: -2
Input jumlah tiket tidak valid, Tolong masukkan lagi
Masukkan jumlah tiket yang dibeli oleh pelanggan ke-4: 8
Dapat diskon 10%
1. Total Penjualan hari ini : 1440000.0
```

CODENYA :

```
import java.util.Scanner;

public class TugasNo1_14 {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double hargaTiket = 50000, diskon = 0, jmlPelanggan, i = 0,
jmlTiketYgDibeli, total = 0, totalPenjualan;
        int _i = (int) i;
        System.out.print("Masukkan jumlah pelanggan hari ini : ");
        jmlPelanggan = sc.nextDouble();
        while (jmlPelanggan > i) {
            System.out.print("Masukkan jumlah tiket yang dibeli oleh pelanggan
ke-" + (_i ++ ) + (": "));
            jmlTiketYgDibeli = sc.nextDouble();
            if (jmlTiketYgDibeli < 0) {
                System.out.println("Input jumlah tiket tidak valid, Tolong
masukkan lagi");
                continue;
            }
            if (jmlTiketYgDibeli > 10) {
```



```

        System.out.println("Dapat diskon 15%");
        total = (jmlTiketYgDibeli * hargaTiket) - (0.15 *
(jmlTiketYgDibeli * hargaTiket));
    }
    else if (jmlTiketYgDibeli > 4) {
        System.out.println("Dapat diskon 10%");
        total = (jmlTiketYgDibeli * hargaTiket) - (0.10 *
(jmlTiketYgDibeli * hargaTiket));
    }
    else if (jmlTiketYgDibeli <= 4 && jmlTiketYgDibeli > 0) {
        total = jmlTiketYgDibeli * hargaTiket;
    }
    i++;
}
totalPenjualan = total * jmlPelanggan;
System.out.println("Total Penjualan hari ini : " + totalPenjualan);
}
}

```

```

Masukkan jenis kendaraan (1 Mobil, 2 Motor, 0 Keluar) : 1
Masukkan durasi : 5
Masukkan jenis kendaraan (1 Mobil, 2 Motor, 0 Keluar) : 2
Masukkan durasi : 3
Masukkan jenis kendaraan (1 Mobil, 2 Motor, 0 Keluar) : 0
2. Total pembayaran parkir: Rp 21000

```

CODENYA:

```

import java.util.Scanner;

public class TugasNo2_14 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int jenis, durasi;
        int total = 0;
        System.out.print("Masukkan jenis kendaraan (1 Mobil, 2 Motor, 0 Keluar) :
");
        jenis = scanner.nextInt();
        while (jenis != 0) {
            if (jenis == 1 || jenis == 2) {
                System.out.print("Masukkan durasi : ");
                durasi = scanner.nextInt();
                if (durasi > 5) {
                    total += 12500;
                } else {

```

```
        if (jenis == 1) {
            total += durasi * 3000;
        } else if (jenis == 2) {
            total += durasi * 2000;
        }
    }
} else {
    System.out.println("Jenis kendaraan tidak valid. Silakan coba
lagi.");
}
System.out.print("Masukkan jenis kendaraan (1 Mobil, 2 Motor, 0
Keluar) : ");
jenis = scanner.nextInt();
}
System.out.println("Total pembayaran parkir: Rp " + total);
scanner.close();
}
}
```