JOBSHEET 7
PERULANGAN 1

1. Tujuan

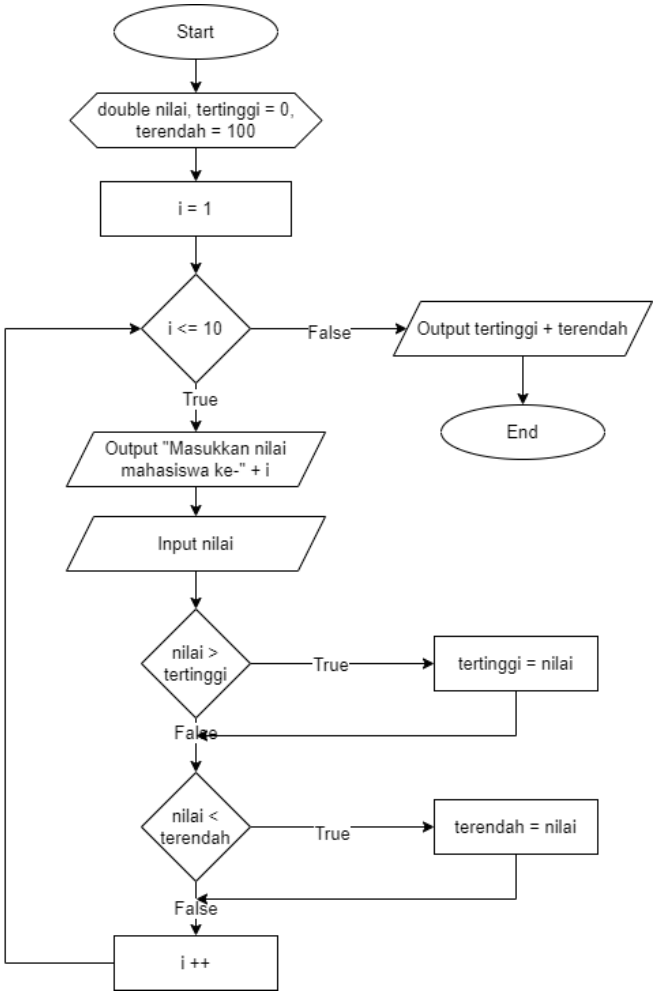
- Mahasiswa dapat menjelaskan format penulisan program perulangan (for, while, dan do-while)
- Mahasiswa dapat mengimplementasikan flowchart perulangan menggunakan bahasa pemrograman Java

2. Praktikum

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

Waktu Percobaan: 90 menit

Di dalam Sistem Informasi Akademik (SIAKAD), 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. Perhatikan flowchart berikut ini:



Berdasarkan flowchart tersebut, buat program menggunakan bahasa pemrograman Java.

2.1.1 Langkah-langkah Percobaan

1. Buat folder baru pada repositori lokal Anda, beri nama **jobsheet7**
2. Buat file baru, beri nama **SiakadForNoPresensi.java**
3. Buatlah struktur dasar program Java yang terdiri dari fungsi **main()**.
4. Tambahkan library Scanner di bagian atas (luar) class
5. Buat deklarasi Scanner dengan nama variabel **sc** di dalam fungsi main()
6. Deklarasikan variabel **nilai**, **tertinggi**, dan **terendah** bertipe double. Inisialisasi **tertinggi** dengan 0 dan **terendah** dengan 100

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

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

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

}
```

8. 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;
    }
}
```

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

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

10. Compile dan run program
11. Commit program Anda ke Github dengan pesan “Percobaan 1”

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 hasil modifikasi Anda ke Github dengan pesan "Modifikasi Percobaan 1"**

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 < nilai ≤ 100), B+ (73 < nilai ≤ 80), B (65 < nilai ≤ 73), C+ (60 < nilai ≤ 65), C (50 < nilai ≤ 60), D (39 < nilai ≤ 50), dan E (nilai ≤ 39).

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

2.2.1 Langkah-langkah Percobaan

1. Buat file baru, beri nama **SiakadWhileNoPresensi.java**
2. Buatlah struktur dasar program Java yang terdiri dari fungsi **main()**.
3. Tambahkan library Scanner di bagian atas (luar) class
4. Buat deklarasi Scanner dengan nama variabel **sc** di dalam fungsi main()
5. Deklarasikan variabel **nilai**, **jml**, dan **i** (untuk perulangan) bertipe integer. Inisialisasi **i** dengan 0 sebagai nilai awal perulangan
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
10. Commit program Anda ke Github dengan pesan “Percobaan 2”

2.2.2 Langkah-langkah Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

```
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 hasil modifikasi Anda ke Github dengan pesan “Modifikasi Percobaan 2”**

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 **KafeDoWhileNoPresensi.java**
2. Buatlah struktur dasar program Java yang terdiri dari fungsi **main()**.
3. Tambahkan library Scanner di bagian atas (luar) class
4. Buat deklarasi Scanner dengan nama variabel **sc** di dalam fungsi main()
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.
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("Masukkan nama pelanggan (ketik 'batal' untuk keluar): ");
    namaPelanggan = sc.nextLine();
    if (namaPelanggan.equalsIgnoreCase("batal")) {
        System.out.println("Transaksi dibatalkan.");
        break;
    }
    System.out.print("Jumlah kopi: ");
    kopi = sc.nextInt();
    System.out.print("Jumlah teh: ");
    teh = sc.nextInt();
    System.out.print("Jumlah roti: ");
    roti = sc.nextInt();
```

```
            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.");
```

Keterangan: sc.nextLine(); setelah print totalHarga merupakan sintaks untuk membersihkan newline dari buffer

8. Compile dan run program
9. Commit program Anda ke Github dengan pesan “Percobaan 3”

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?

3. Tugas

Waktu Percobaan : 120 Menit

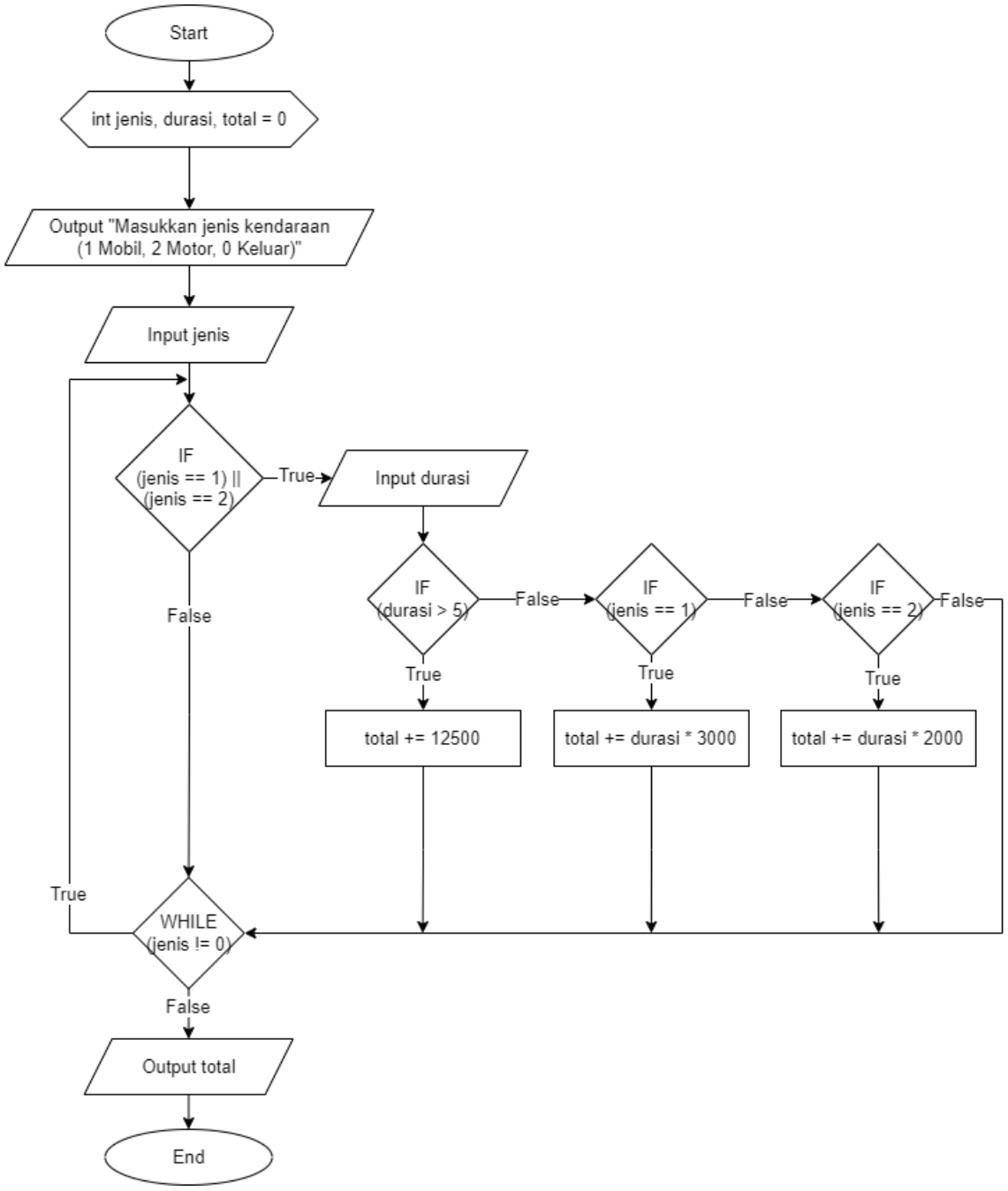
1. Seorang pengelola bioskop ingin membuat program untuk menghitung total penjualan tiket dalam satu hari. Tiket dijual dengan harga Rp 50.000 per tiket. Program harus menghitung total tiket yang terjual dan total harga penjualan tiket selama satu hari dengan ketentuan sebagai berikut:
   - Jika pelanggan 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.

Commit dan push program Anda ke Github dengan pesan "Tugas 1"

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! **Commit dan push program Anda ke Github dengan pesan "Tugas 2"**