JOBSHEET 7
PERULANGAN 1

1. Tujuan

- Mahasiswa dapat menjelaskan format penulisan program perulangan bagian 1
- Mahasiswa dapat mengimplementasikan flowchart perulangan bagian 1 menggunakan bahasa pemrograman Java

2. Praktikum

2.1 Percobaan 1: Menghitung Bilangan Kelipatan Menggunakan FOR

Waktu Percobaan: 60 menit

Pada percobaan ini dilakukan pembuatan kode program untuk menampilkan bilangan kelipatan angka tertentu dari rentang 1 sampai dengan 50 menggunakan perulangan FOR, serta menghitung total dari bilangan-bilangan tersebut.

1. Buka text editor. Buat file baru, beri nama **ForKelipatanNoAbsen.java**
2. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi **main()**
3. Tambahkan library Scanner
4. Buat deklarasi **Scanner** dengan nama **scan**
5. Buatlah variabel bertipe **int** dengan nama **kelipatan**, **jumlah**, dan **counter**. Inisialisasi variabel **jumlah** dan **counter** dengan nilai 0
6. Tambahkan kode berikut ini untuk menerima input dari keyboard

```
System.out.print("Masukkan bilangan kelipatan (1-9): ");
kelipatan = scan.nextInt();
```

7. Buat struktur perulangan FOR dengan kondisi pemilihan IF untuk menentukan bilangan kelipatan

```
for (int i = 1; i <= 50; i++) {
    if (i % kelipatan == 0) {
        total += i;
        counter++;
    }
}
```

8. Tampilkan banyaknya bilangan kelipatan dan total bilangan kelipatan pada rentang 1 sampai dengan 50.

```
System.out.printf("Banyaknya bilangan %d dari 1 sampai 50 adalah %d\n", kelipatan, counter);
System.out.printf("Total bilangan kelipatan %d dari 1 sampai 50 adalah %d\n", kelipatan, total);
```

9. Jalankan program tersebut. Cocokkan hasil compile kode program Anda dengan gambar berikut ini

```
Masukkan bilangan kelipatan (1-9): 5
Banyaknya bilangan 5 dari 1 sampai 50 adalah 10
Total bilangan kelipatan 5 dari 1 sampai 50 adalah 275
```

10. Commit dan push kode program ke github

Pertanyaan

1. Terdapat tiga komponen perulangan pada sintaks FOR. Berdasarkan Percobaan 1 tersebut, sebutkan dan tunjukkan masing-masing komponen perulangan FOR pada kode program yang telah dibuat!
2. Jelaskan alur kerja dari potongan kode program berikut!

```
for (int i = 1; i <= 50; i++) {
    if (i % kelipatan == 0) {
        total += i;
        counter++;
    }
}
```

3. Modifikasi kode program yang telah dibuat dengan menambahkan variabel baru untuk menghitung rata-rata dari seluruh bilangan kelipatan yang ditentukan! Push dan commit kode program ke github.
4. Buatlah file baru dengan nama **WhileKelipatanNoAbsen.java**. Buatlah kode program dengan tujuan serupa tetapi menggunakan WHILE. Push dan commit kode program ke github.

2.2 Percobaan 2: Menghitung Gaji Lembur Karyawan Menggunakan WHILE dan CONTINUE

Waktu Percobaan: 60 menit

Sebuah perusahaan memberikan gaji lembur kepada karyawannya setiap minggu. Gaji tersebut dihitung berdasarkan jabatan karyawan dan jumlah jam lembur dalam seminggu. Karyawan dengan jabatan “direktur” tidak mendapatkan tambahan gaji meskipun melakukan

lembur, karyawan dengan jabatan “manager” mendapatkan gaji lembur sebesar 100000 per jam, sedangkan karyawan dengan jabatan “staf” mendapatkan gaji lembur sebesar 75000 per jam. Pada percobaan ini dilakukan pembuatan kode program menggunakan WHILE dan CONTINUE untuk menghitung pengeluaran perusahaan.

1. Buka text editor. Buat file baru, beri nama **WhileGajiNoAbsen.java**
2. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi **main()**
3. Tambahkan library Scanner
4. Buat deklarasi **Scanner** dengan nama **scan**
5. Deklarasikan variabel **jumlahKaryawan** dan **jumlahJamLembur** bertipe int serta **gajiLembur** dan **totalGajiLembur** bertipe double. Inisialisasi variabel **gajiLembur** dan **totalGajiLembur** dengan nilai 0
6. Deklarasikan variabel **jabatan** bertipe String
7. Tambahkan kode berikut ini untuk menerima input dari keyboard guna menentukan jumlah karyawan yang akan dihitung gajinya

```
System.out.print("Masukkan jumlah karyawan: ");
jumlahKaryawan = sc.nextInt();
```

8. Buat struktur perulangan WHILE dengan kondisi pemilihan IF-ELSE dan CONTINUE untuk menentukan gaji lembur berdasarkan jabatan karyawan

```
int i = 0;

while (i < jumlahKaryawan) {
    System.out.println("Pilihan jabatan - Direktur, Manajer, Karyawan");
    System.out.print("Masukkan jabatan karyawan ke-" + (i+1) + ": ");
    jabatan = sc.next();
    System.out.print("Masukkan jumlah jam lembur: ");
    jumlahJamLembur = sc.nextInt();
    i++;

    if (jabatan.equalsIgnoreCase("direktur")) {
        continue;
    } else if (jabatan.equalsIgnoreCase("manajer")) {
        gajiLembur = jumlahJamLembur * 100000;
    }
    else if (jabatan.equalsIgnoreCase("karyawan")) {
        gajiLembur = jumlahJamLembur * 75000;
    }

    totalGajiLembur += gajiLembur;
}
```

9. Tampilkan hasil perhitungan jumlah gaji

```
System.out.println("Total gaji lembur: " + totalGajiLembur);
```

10. Jalankan program tersebut. Cocokkan hasil compile kode program Anda dengan gambar berikut ini

```
Masukkan jumlah karyawan: 3
Pilihan jabatan - Direktur, Manajer, Karyawan
Masukkan jabatan karyawan ke-1: manajer
Masukkan jumlah jam lembur: 1
Pilihan jabatan - Direktur, Manajer, Karyawan
Masukkan jabatan karyawan ke-2: direktur
Masukkan jumlah jam lembur: 10
Pilihan jabatan - Direktur, Manajer, Karyawan
Masukkan jabatan karyawan ke-3: karyawan
Masukkan jumlah jam lembur: 5
Total gaji lembur: 475000.0
```

11. Push dan commit kode program ke github.

Pertanyaan

1. Tunjukkan bagian kode program yang digunakan sebagai syarat untuk menghentikan perulangan WHILE! Berapa kali perulangan dilakukan?
2. Pada potongan kode berikut,

```
if (jabatan.equalsIgnoreCase("direktur")) {
    continue;
```

Apa yang sebenarnya terjadi jika variabel **jabatan** berisi nilai “DIREKTUR”? Apa peran CONTINUE yang dituliskan di dalam sintaks perulangan?

3. Mengapa komponen update **i++** diletakkan di posisi tengah, tidak di bagian akhir statement? Pindahkan **i++** di bagian akhir, lalu jalankan kembali program dengan memasukkan “direktur” sebagai jabatan karyawan pertama. Apa yang terjadi? Jelaskan!
4. Modifikasi kode program untuk menghandle jabatan yang invalid seperti contoh berikut:

```
Masukkan jumlah karyawan: 3
Pilihan jabatan - Direktur, Manajer, Karyawan
Masukkan jabatan karyawan ke-1: manajer
Masukkan jumlah jam lembur: 10
Pilihan jabatan - Direktur, Manajer, Karyawan
Masukkan jabatan karyawan ke-2: direktur
Masukkan jumlah jam lembur: 5
Pilihan jabatan - Direktur, Manajer, Karyawan
Masukkan jabatan karyawan ke-3: pegawai
Masukkan jumlah jam lembur: 4
Jabatan invalid
Pilihan jabatan - Direktur, Manajer, Karyawan
Masukkan jabatan karyawan ke-3: karyawan
Masukkan jumlah jam lembur: 4
Total gaji lembur: 1300000.0
```

5. Push dan commit kode program ke github

2.3 Percobaan 3: Menghitung Jatah Cuti Menggunakan DO-WHILE

Waktu Percobaan: 50 menit

Pada percobaan ini dilakukan pembuatan kode program menggunakan DO-WHILE untuk menghitung jatah cuti yang dimiliki oleh pagawai. Pegawai mempunyai jatah cuti sebanyak 5 hari. Jatah cuti akan dikurangi perhati setiap kali digunakan. Saat jatah cuti sisa 2 hari, pegawai mendapat peringatan untuk berhenti menggunakan jatah cutinya.

1. Buka text editor. Buat file baru, beri nama **DoWhileCutiNoAbsen.java**
2. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi **main()**
3. Tambahkan library Scanner
4. Buat deklarasi **Scanner** dengan nama sc
5. Buatlah variabel **jatahCuti** dan **jumlahHari** bertipe int
6. Buatlah variabel **konfirmasi** bertipe String
7. Buat struktur perulangan DO-WHILE untuk menerima input dari keyboard dan menghitung jatah cuti

```
int jatahCuti, jumlahHari;
String konfirmasi;

System.out.print("Jatah cuti: ");
jatahCuti = sc.nextInt();

do {
    System.out.print("Apakah Anda ingin mengambil cuti (y/t)? ");
    konfirmasi = sc.next();

    if (konfirmasi.equalsIgnoreCase("y")) {
        System.out.print("Jumlah hari: ");
        jumlahHari = sc.nextInt();

        if (jumlahHari <= jatahCuti) {
            jatahCuti -= jumlahHari;
            System.out.println("Sisa jatah cuti: " + jatahCuti);
        } else {
            System.out.println("Sisa jatah cuti Anda tidak mencukupi");
            break;
        }
    }
} while (jatahCuti > 0);
```

8. Jalankan program tersebut. Cocokkan hasil *running* program yang sudah Anda buat apakah sudah sesuai dengan tampilan berikut

```
Jatah cuti: 12
Apakah Anda ingin mengambil cuti (y/t)? y
Jumlah hari: 4
Sisa jatah cuti: 8
Apakah Anda ingin mengambil cuti (y/t)? y
Jumlah hari: 5
Sisa jatah cuti: 3
Apakah Anda ingin mengambil cuti (y/t)? y
Jumlah hari: 4
Sisa jatah cuti Anda tidak mencukupi
```

9. Push dan commit kode program ke github

Pertanyaan

1. Apa kegunaan sintaks BREAK di dalam sintaks perulangan?
2. Modifikasi kode program sehingga jika jumlah hari cuti yang ingin diambil lebih besar daripada jatah yang tersisa, program tidak berhenti sehingga pengguna masih memiliki kesempatan untuk mengisikan jumlah hari sesuai jatah cuti.
3. Push dan commit kode program ke github
4. Pada saat input konfirmasi, ketikkan “t”, apa yang terjadi? Mengapa demikian?
5. Modifikasi kode program sehingga saat pengguna mengetikkan “t” sebagai input konfirmasi, maka program akan berhenti
6. Push dan commit kode program ke github

3. Tugas

Waktu Percobaan : 130 Menit

- Implementasikan flowchart yang telah dibuat pada tugas pertemuan 7 Matakuliah Dasar Pemrograman terkait project ke dalam kode program
- Push dan commit hasil kode program anda ke repository project Anda
- Catatan: tugas hanya boleh menerapkan materi dari pertemuan 1 hingga pertemuan 7.