Andini W.

202310370311453

Ketika program dijalankan, menu berikut akan ditampilkan:

```
Menu:
1. Tambah Siswa
2. Urutkan dan Tampilkan berdasarkan Nama
3. Urutkan dan Tampilkan berdasarkan Nilai
4. Keluar
Pilihan:
```

Pengguna diminta memasukkan angka sesuai opsi 1,2,3, atau 4. Input akan dibaca menggunakan **Scanner** dan disimpam dalam variable **pilihan**

Menu 1 : Tambah Siswa

Memanggil method tambahSiswa

```
static void tambahSiswa() {  1 usage  new *
    Scanner scanner = new Scanner(System.in);
    System.out.println("Masukkan data siswa. Ketik 'selesai' untuk berhenti.");

    while (true) {
        System.out.print("Masukkan nama siswa: ");
        String nama = scanner.nextLine();
        if (nama.equalsIgnoreCase( anotherString: "selesai")) {
            break;
        }
        System.out.print("Masukkan nilai siswa: ");
        int nilai = scanner.nextInt();
        scanner.nextLine(); // Membaca newline setelah nilai
        daftarSiswa.add(new Siswa(nama, nilai));
        System.out.println("Data siswa berhasil ditambahkan.");
    }
}
```

Program meminta pengguna memasukkan nama siswa dan nilai siswa, program akan terus meminta data hingga pengguna mengetik **selesai**

Data disimpan ke ArrayList **daftarSiswa.add(new Siswa(nama, nilai));** Setelah selesai program akan Kembali ke menu utama.

Menu 2 : Urutkan berdasarkan nama

Memanggil method urutkanBerdasarkanNama

```
static void urutkanBerdasarkanNama() {  1 usage  new *
    Collections.sort(daftarSiswa, new Comparator<Siswa>() {  new *
        @Override  new *
        public int compare(Siswa s1, Siswa s2) {
            return s1.nama.compareTo(s2.nama);
        }
    });
    tampilkanDataSiswa();
}
```

Collections.sort digunakan untuk mengurutkan **daftarSiswa** berdasarkan atribut nama.

Comparator membandingkan dua nama siswa (**s1.nama dan s2.nama**) secara alfabet.

Data yang sudah diurutkan ditampilkan dengan memanggil **tampilkanDataSiswa.**

Menu 3 : Urutkan berdarsakan nilai

Memanggil method urutkanBerdasarkanNilai

```
static void urutkanBerdasarkanNilai() {  1 usage  new *
    Collections.sort(daftarSiswa, new Comparator<Siswa>() {  new *
        @Override  new *
        public int compare(Siswa s1, Siswa s2) {
            return s2.nilai - s1.nilai; // Urut dari nilai terbesar
        }
    });
    tampilkanDataSiswa();
}
```

Collections.sort digunakan untuk mengurutkan **daftarSiswa** berdasarkan atribut nilai.
Comparator membandingkan nilai siswa (**s2.nilai - s1.nilai**) sehingga data diurutkan secara menurun.
Data yang sudah diurutkan ditampilkan dengan memanggil **tampilkanDataSiswa**.

Menu 4 : Keluar
Program akan menampilkan pesan “Keluar dari program”, Program berhenti.

- java.util.ArrayList digunakan untuk menyimpan daftar elemen secara dinamis.
- java.util.Collections melakukan operasi seperti pengurutan, pencarian, pengacakan, dan lainnya.
- java.util.Comparator digunakan untuk mendefinisikan aturan pengurutan objek secara khusus (custom).