MODUL III

SORTING AND SEARCHING

2.1 PERMASALAHAN

2.1.1 Easy Level

Kalian masih ingat dengan sosok bernama Rijal yang memiliki rental PS? Ia telah kembali pada cerita kali ini dengan banyak tokoh baru lain-nya.Di siang hari yang sangat terik ini. Rijal sedang sibuk di meja kasir rental PS miliknya ketika Eysar masuk. Eysar adalah teman dekat Rijal semasa perkuliahan dulu. Biasanya, Eysar datang ke sini dengan semangat untuk bermain PES, tapi kali ini wajahnya terlihat murung.

“Kenapa Sar? Biasanya langsung nyari stick, kok sekarang malah bengong?” tanya Rijal, penasaran.

Eysar duduk di sofa yang biasa digunakan pelanggan untuk menunggu giliran bermain. "Nggak tahu, Jal. Lagi galau Jal. Lagi nggak mood buat main" jawabnya sambil menghela napas. Rijal mengerutkan dahi. "LOL, galau kenapa tuh? Cerita dah. Kalau duduk doang di sini, nggak seru kan".

Eysar tersenyum tipis. "Kayaknya saya mau bikin playlist deh di Spotify. Buat nemenin kalau lagi galau begini". Rijal tertawa kecil. "LOL, playlist galau nih. Biar playlist-nya penuh nuansa sendu kayak hatimu?" godanya.

“LOL” kata Eysar sambil mengangguk, mengeluarkan ponsel dan membuka aplikasi Spotify yang masih kosong tanpa lagu di playlist barunya. Dengan sabar, ia mulai memilih lagu-lagu yang cocok. Rijal, yang duduk di sebelahnya sambil mengecek pemesanan pelanggan lain, sesekali memberikan rekomendasi lagu.

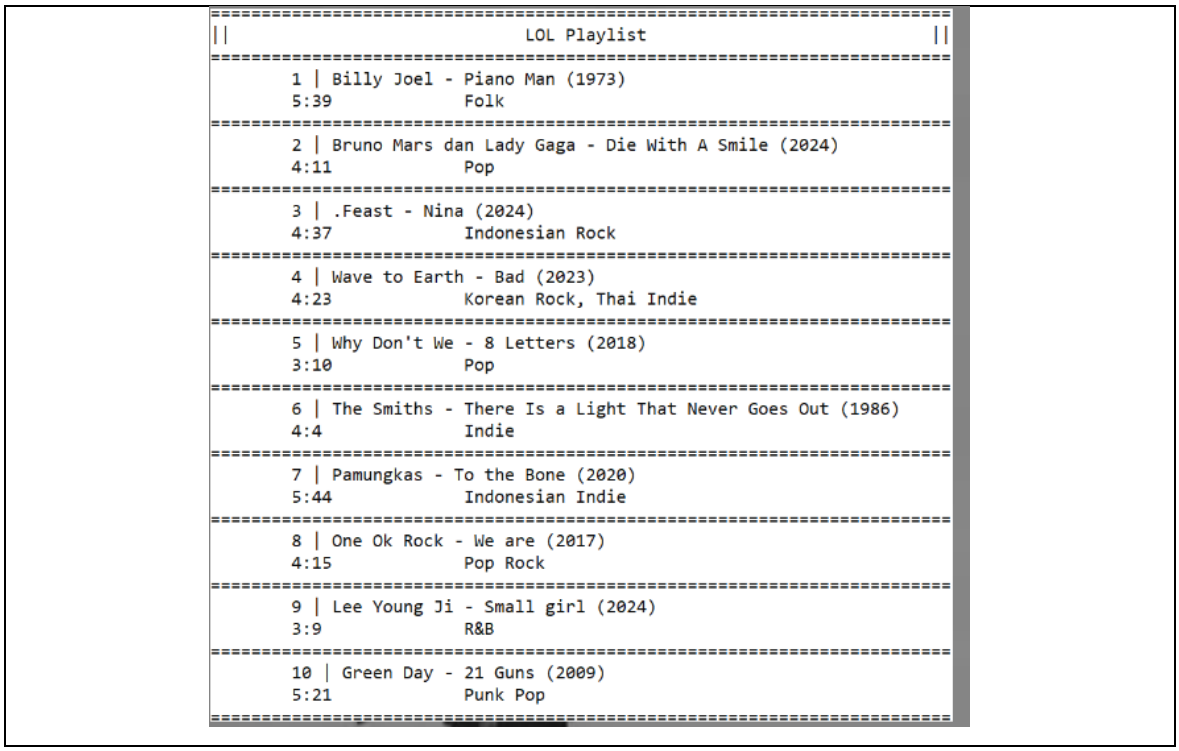
“Apa nama playlist buat galamu tu Sar?” kata Rijal yang tertarik dengan playlist Eysar. Eysar tertawa “Rijal Ganteng gimana?”. “LOL” kata Rijal sambil ikut tertawa. Eysar berkata sambil bersemangat “Nah itu aja gimana Jal? LOL Playlist?”. “LOL, iyadah” kata Rijal sambil tertawa.

```
==========================================================================
||                          LOL Playlist                                ||
==========================================================================
                There is no song in the playlist yet.
==========================================================================
```

*tampilan di saat belum ada lagu yang ditambahkan

“Saya masukin sepuluh lagu dulu ga sih, ” kata Eysar sambil menambahkan lagu-lagu pilihan-nya ke dalam playlist.

*Note : gunakan double linked list.



```
==========================================================================
||                          LOL Playlist                                ||
==========================================================================
       1 | Billy Joel - Piano Man (1973)
       5:39            Folk
==========================================================================
       2 | Bruno Mars dan Lady Gaga - Die With A Smile (2024)
       4:11            Pop
==========================================================================
       3 | .Feast - Nina (2024)
       4:37            Indonesian Rock
==========================================================================
       4 | Wave to Earth - Bad (2023)
       4:23            Korean Rock, Thai Indie
==========================================================================
       5 | Why Don't We - 8 Letters (2018)
       3:10            Pop
==========================================================================
       6 | The Smiths - There Is a Light That Never Goes Out (1986)
       4:4             Indie
==========================================================================
       7 | Pamungkas - To the Bone (2020)
       5:44            Indonesian Indie
==========================================================================
       8 | One Ok Rock - We are (2017)
       4:15            Pop Rock
==========================================================================
       9 | Lee Young Ji - Small girl (2024)
       3:9             R&B
==========================================================================
       10 | Green Day - 21 Guns (2009)
       5:21            Punk Pop
==========================================================================
```

* tampilan sesudah ada lagu yang ditambahkan

*Note : atributnya ada judul lagu, nama artis, tahun rilis, durasi (dalam menit dan detik), dan genre.

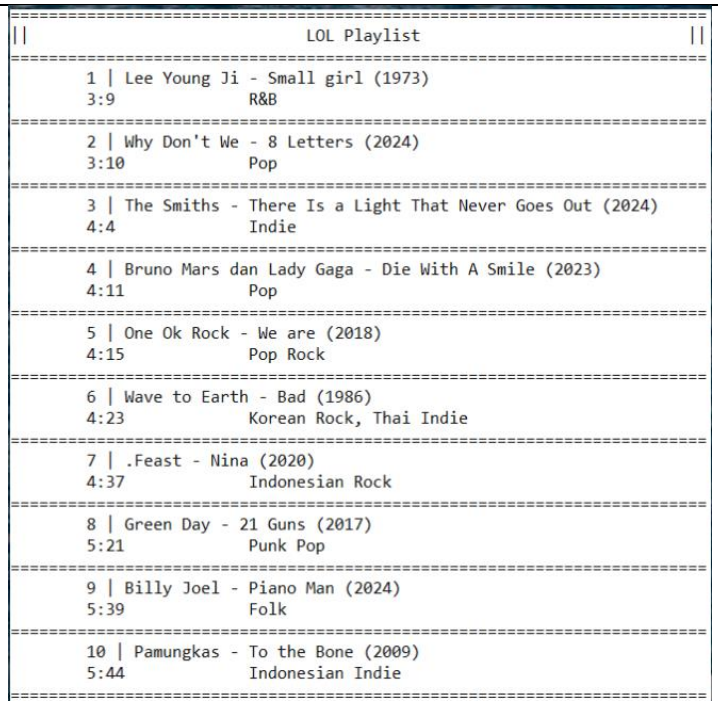
Setelah playlistnya terisi dengan sepuluh lagu, Eysar mulai berpikir untuk mengurutkan lagu-lagu ini dengan lebih spesifik lagi. Kali ini, ia ingin mengurutkannya berdasarkan durasi

lagu agar alurnya terasa lebih pas.

Rijal, yang sedang membereskan stik PS di rak, mendekat lagi setelah mendengar keinginan Eysar. “Kalau urutin dari durasi, berarti lagu paling pendek dulu, baru yang lebih panjang kan?”.

Eysar mengangguk. Dengan sedikit bantuan dari Rijal, ia memeriksa durasi setiap lagu di playlist dan mulai menyusun ulang. Satu per satu, lagu-lagu tersebut diurutkan dari yang terpendek hingga yang terpanjang. Setelah selesai, playlist itu terlihat lebih rapi dan teratur.

*Note : Mengurutkan menggunakan algoritma Bubble Sort (Asc) berdasarkan durasi menit dan detik



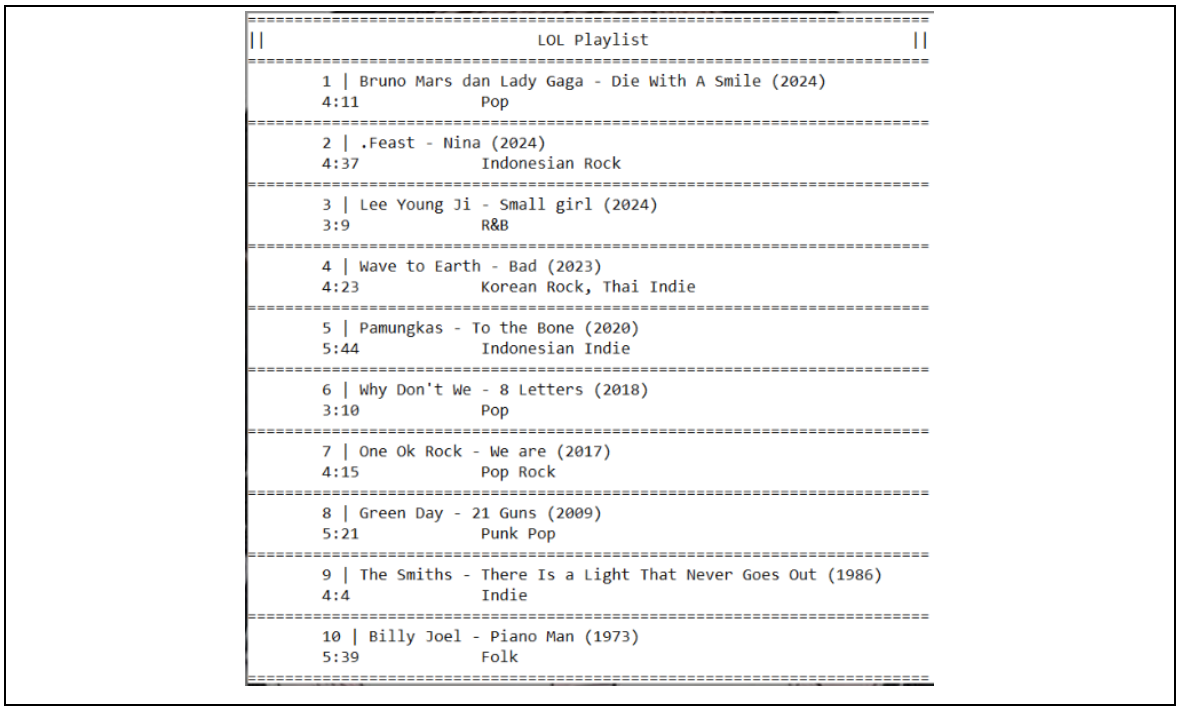
```
==========================================================================
||                          LOL Playlist                                ||
==========================================================================
        1 | Lee Young Ji - Small girl (1973)
        3:9             R&B
==========================================================================
        2 | Why Don't We - 8 Letters (2024)
        3:10            Pop
==========================================================================
        3 | The Smiths - There Is a Light That Never Goes Out (2024)
        4:4             Indie
==========================================================================
        4 | Bruno Mars dan Lady Gaga - Die With A Smile (2023)
        4:11            Pop
==========================================================================
        5 | One Ok Rock - We are (2018)
        4:15            Pop Rock
==========================================================================
        6 | Wave to Earth - Bad (1986)
        4:23            Korean Rock, Thai Indie
==========================================================================
        7 | .Feast - Nina (2020)
        4:37            Indonesian Rock
==========================================================================
        8 | Green Day - 21 Guns (2017)
        5:21            Punk Pop
==========================================================================
        9 | Billy Joel - Piano Man (2024)
        5:39            Folk
==========================================================================
        10 | Pamungkas - To the Bone (2009)
        5:44            Indonesian Indie
==========================================================================
```

* tampilan sesudah lagu di urutkan berdasarkan menit dan detik

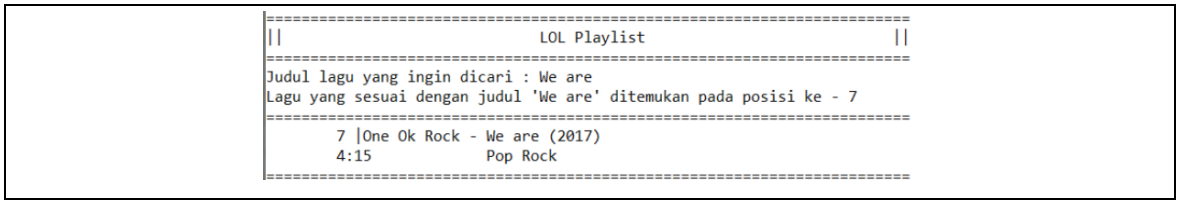
Namun, Eysar tiba-tiba punya ide lain. "Jal, kalau saya urutinnya berdasarkan lagu terbaru yang rilis". Rijal tersenyum mengerti. "Jadi urutannya berdasarkan tahun rilis terbaru kan Sar?".

Eysar mengangguk lagi. Ia pun mengaktifkan opsi pengurutan otomatis berdasarkan tahun rilis, sehingga lagu-lagu terbaru akan berada di uturan teratas playlist.

*Note : Mengurutkan menggunakan algoritma Selection Sort (Desc) berdasarkan tahun rilis.



```
==========================================================================
||                          LOL Playlist                                ||
==========================================================================
       1 | Bruno Mars dan Lady Gaga - Die With A Smile (2024)
       4:11             Pop
==========================================================================
       2 | .Feast - Nina (2024)
       4:37             Indonesian Rock
==========================================================================
       3 | Lee Young Ji - Small girl (2024)
       3:9              R&B
==========================================================================
       4 | Wave to Earth - Bad (2023)
       4:23             Korean Rock, Thai Indie
==========================================================================
       5 | Pamungkas - To the Bone (2020)
       5:44             Indonesian Indie
==========================================================================
       6 | Why Don't We - 8 Letters (2018)
       3:10             Pop
==========================================================================
       7 | One Ok Rock - We are (2017)
       4:15             Pop Rock
==========================================================================
       8 | Green Day - 21 Guns (2009)
       5:21             Punk Pop
==========================================================================
       9 | The Smiths - There Is a Light That Never Goes Out (1986)
       4:4              Indie
==========================================================================
       10 | Billy Joel - Piano Man (1973)
       5:39             Folk
==========================================================================
```



```
==========================================================================
||                          LOL Playlist                                ||
==========================================================================
Judul lagu yang ingin dicari : We are
Lagu yang sesuai dengan judul 'We are' ditemukan pada posisi ke - 7
==========================================================================
       7 |One Ok Rock - We are (2017)
       4:15             Pop Rock
==========================================================================
```

2.2 HASIL PERCOBAAN

2.2.1 Hard Level

1. Algoritma
    a. Inisialisasi program.
    b. Menambah Lagu.
    c. Menampilkan antrian.
    d. Bubble Sort (Durasi).
    e. Selection Sort (Tahun).
    f. Menampilkan Playlist.
    g. Mencari Lagu.
    h. Menampilkan Playlist Setelah Perubahan.
2. *Source Code*

```
class Song {
    String title;
    String artist;
    int year;
    double duration;
    String genre;
    Song prev, next;

    public Song(String artist, String title, int year, double
duration, String genre) {
        this.artist = artist;
        this.title = title;
        this.year = year;
        this.duration = duration;
        this.genre = genre;
        this.prev = null;
        this.next = null;
    }
}

class Playlist {
    Song head, tail;
    int size;

    public Playlist() {
        this.head = null;
        this.tail = null;
        this.size = 0;
    }

    public void addSong(Song song) {
        if (head == null) {
            head = song;
            tail = song;
        } else {
            tail.next = song;
            song.prev = tail;
            tail = song;
        }
```

```
            size++;
        }

        public void bubbleSortByDuration() {
            if (head == null) return;

            boolean swapped;
            do {
                swapped = false;
                Song current = head;
                while (current != null && current.next != null) {
                    if (current.duration > current.next.duration) {
                        String tempTitle = current.title;
                        String tempArtist = current.artist;
                        int tempYear = current.year;
                        double tempDuration = current.duration;
                        String tempGenre = current.genre;

                        current.title = current.next.title;
                        current.artist = current.next.artist;
                        current.year = current.next.year;
                        current.duration = current.next.duration;
                        current.genre = current.next.genre;

                        current.next.title = tempTitle;
                        current.next.artist = tempArtist;
                        current.next.year = tempYear;
                        current.next.duration = tempDuration;
                        current.next.genre = tempGenre;

                        swapped = true;
                    }
                    current = current.next;
                }
            } while (swapped);
        }

        public void selectionSortByYear() {
            if (head == null) return;

            Song current = head;
            while (current != null) {
                Song maxNode = current;
                Song check = current.next;
                while (check != null) {
                    if (check.year > maxNode.year) {
                        maxNode = check;
                    }
                    check = check.next;
                }
                if (maxNode != current) {
                    String tempTitle = current.title;
                    String tempArtist = current.artist;
                    int tempYear = current.year;
                    double tempDuration = current.duration;
                    String tempGenre = current.genre;

                    current.title = maxNode.title;
                    current.artist = maxNode.artist;
                    current.year = maxNode.year;
                    current.duration = maxNode.duration;
```

```
                    current.genre = maxNode.genre;

                    maxNode.title = tempTitle;
                    maxNode.artist = tempArtist;
                    maxNode.year = tempYear;
                    maxNode.duration = tempDuration;
                    maxNode.genre = tempGenre;
                }
                current = current.next;
            }
        }

        public void display() {

    System.out.println("==========================================
    ==============");
            System.out.println("||                          LOL
    Playlist                    ||");

    System.out.println("==========================================
    ==============");

            Song current = head;
            int index = 1;
            while (current != null) {
                String line1 = index + " | " + current.artist + " -
    " + current.title + " (" + current.year + ")";
                String line2 = current.duration + "       " +
    current.genre;

                System.out.println(line1);
                System.out.println(line2);

    System.out.println("==========================================
    ==============");

                current = current.next;
                index++;
            }
        }

        public void searchByTitle(String searchTitle) {
            int index = 1;
            Song current = head;
            while (current != null) {
                if (current.title.equalsIgnoreCase(searchTitle)) {

    System.out.println("==========================================
    ==============");
                    System.out.printf("Song with title '%s' found at
    position %d\n", searchTitle, index);

    System.out.println("==========================================
    ==============");

                    String line1 = index + " | " + current.artist +
    " - " +  current.title + " (" + current.year + ")";
                    String line2 = current.duration + "
    " + current.genre;
                    System.out.println(line1);
                    System.out.println(line2);
```

```
System.out.println("==========================================
==============");
                return;
            }
            current = current.next;
            index++;
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Playlist playlist = new Playlist();

        playlist.addSong(new Song("Billy Joel", "Piano Man",
1973, 5.39, "Folk"));
        playlist.addSong(new Song("Bruno Mars", "Lady Gaga",
2024, 4.11, "Pop"));
        playlist.addSong(new Song(".Feast", "Nina", 2024, 4.37,
"Indonesia Rock"));
        playlist.addSong(new Song("Wave to Earth", "Bad", 2023,
4.23, "Korean Rock, Thai Indie"));
        playlist.addSong(new Song("Why Don't We", "8 Lettes",
1970, 4.03, "Rock"));
        playlist.addSong(new Song("The Smiths", "There Is a
Light That Never Goes Out", 1986, 4.4, "Indie"));
        playlist.addSong(new Song("Pamungkas", "To the Bone",
2020, 5.44, "Indonesian Indie"));
        playlist.addSong(new Song("One Ok Rock", "We are", 2017,
4.15, "Pop Rock"));
        playlist.addSong(new Song("Lee Young Ji", "Smart Girl",
2024, 3.9, "R&B"));
        playlist.addSong(new Song("Green Day", "21 Gungs", 2009,
5.21, "Punk Pop"));
        playlist.addSong(new Song("Adele", "Someone Like You",
2011, 4.45, "Pop"));
        playlist.addSong(new Song("Ed Sheeran", "Shape of You",
2017, 4.24, "Pop"));
        playlist.addSong(new Song("Coldplay", "Fix You", 2005,
4.55, "Alternative Rock"));
        playlist.addSong(new Song("Taylor Swift", "Shake It
Off", 2014, 3.39, "Pop"));
        playlist.addSong(new Song("Kendrick Lamar", "HUMBLE.",
2017, 2.57, "Hip-Hop"));
        playlist.addSong(new Song("Sia", "Chandelier", 2014,
3.53, "Pop"));
        playlist.addSong(new Song("Maroon 5", "Sugar", 2014,
5.02, "Pop-Rock"));
        playlist.addSong(new Song("Post Malone", "Rockstar",
2017, 3.38, "Hip-Hop"));
        playlist.addSong(new Song("Drake", "God's Plan", 2018,
3.19, "Hip-Hop"));
        playlist.addSong(new Song("The Weeknd", "Blinding
Lights", 2020, 3.20, "Pop"));
        playlist.addSong(new Song("Beyoncé", "Halo", 2008, 4.21,
"Pop-R&B"));
        playlist.addSong(new Song("Billie Eilish", "Bad Guy",
2019, 3.14, "Pop"));
        playlist.addSong(new Song("Imagine Dragons", "Believer",
2017, 3.24, "Rock"));
```

```
        playlist.addSong(new Song("Dua Lipa", "Don't Start Now",
2019, 3.03, "Pop"));
        playlist.addSong(new Song("Drake", "In My Feelings",
2018, 3.37, "Hip-Hop"));
        playlist.addSong(new Song("Shawn Mendes", "Senorita",
2019, 3.11, "Pop"));
        playlist.addSong(new Song("Kanye West", "Stronger",
2007, 5.11, "Hip-Hop"));
        playlist.addSong(new Song("Rihanna", "Diamonds", 2012,
3.45, "Pop"));
        playlist.addSong(new Song("Lana Del Rey", "Summertime
Sadness", 2012, 4.25, "Pop"));
        playlist.addSong(new Song("Snoop Dogg", "Drop It Like
It's Hot", 2004, 4.30, "Hip-Hop"));
        playlist.addSong(new Song("Lil Nas X", "Old Town Road",
2019, 2.37, "Country-Rap"));
        playlist.addSong(new Song("Travis Scott", "SICKO MODE",
2018, 5.12, "Hip-Hop"));
        playlist.addSong(new Song("Halsey", "Without Me", 2018,
3.21, "Pop"));
        playlist.addSong(new Song("Katy Perry", "Roar", 2013,
3.42, "Pop"));
        playlist.addSong(new Song("The Chainsmokers", "Closer",
2016, 4.05, "Pop"));
        playlist.addSong(new Song("Marshmello", "Happier", 2018,
3.34, "Electronic"));
        playlist.addSong(new Song("Zedd", "Stay", 2017, 3.30,
"Electronic"));
        playlist.addSong(new Song("Post Malone", "Circles",
2019, 3.35, "Pop-Rock"));
        playlist.addSong(new Song("Ariana Grande", "No Tears
Left to Cry", 2018, 3.25, "Pop"));
        playlist.addSong(new Song("Khalid", "Talk", 2019, 3.17,
"R&B"));
        playlist.addSong(new Song("Megan Thee Stallion", "Savage
Remix", 2020, 4.03, "Hip-Hop"));
        playlist.addSong(new Song("BTS", "Dynamite", 2020, 3.19,
"K-Pop"));
        playlist.addSong(new Song("Drake", "Nonstop", 2018,
3.58, "Hip-Hop"));
        playlist.addSong(new Song("Lizzo", "Truth Hurts", 2017,
2.53, "Pop"));
        playlist.addSong(new Song("Camila Cabello", "Havana",
2017, 3.37, "Pop"));
        playlist.addSong(new Song("Lil Baby", "Drip Too Hard",
2018, 3.04, "Hip-Hop"));
        playlist.addSong(new Song("Maroon 5", "Girls Like You",
2018, 3.30, "Pop-Rock"));
        playlist.addSong(new Song("Charlie Puth", "Attention",
2017, 3.31, "Pop"));
        playlist.addSong(new Song("Shakira", "Hips Don't Lie",
2006, 3.38, "Pop"));
        playlist.addSong(new Song("OneRepublic", "Counting
Stars", 2013, 4.17, "Pop-Rock"));

        System.out.println("\nTampilan Setelah Lagu
Ditambahkan:");
        playlist.display();

        long startTimeBubbleSort = System.nanoTime();
        playlist.bubbleSortByDuration();
```

```
        long endTimeBubbleSort = System.nanoTime();
        System.out.println("\nTampilan Sesudah Lagi Di Urutkan
Berdasarkan Menit dan Detik:");
        playlist.display();
        System.out.println("\nBubble Sort memakan waktu: " +
(endTimeBubbleSort - startTimeBubbleSort) + " nanodetik.");

        long startTimeSelectionSort = System.nanoTime();
        playlist.selectionSortByYear();
        long endTimeSelectionSort = System.nanoTime();
        System.out.println("\nTampilan Sesudah Lagu Di Urutkan
Berdasarkan Tahun :");
        playlist.display();
        System.out.println("\nSelection Sort memakan waktu: " +
(endTimeSelectionSort - startTimeSelectionSort) + "
nanodetik.");

        String searchTitle = "We Are";

System.out.println("\n================================");
        System.out.println("Judul lagu yang ingin dicari: " +
searchTitle);

        Song current = playlist.head;
        int index = 1;
        boolean found = false;

        while (current != null) {
            if (current.title.equalsIgnoreCase(searchTitle)) {
                found = true;
                System.out.println("Lagu dengan judul '" +
searchTitle + "' ditemukan pada posisi ke-" + index);

System.out.println("================================");
                String line1 = index + " | " + current.artist +
" - " + current.title + " (" + current.year + ")";
                String line2 = current.duration + " menit    " +
current.genre;
                System.out.println(line1);
                System.out.println(line2);

System.out.println("================================");
                break;
            }
            current = current.next;
            index++;
        }
        if (!found) {
            System.out.println("Lagu dengan judul '" +
searchTitle + "' tidak ditemukan.");
        }
    }
}
```

2.3 ANALISIS DATA

2.3.1 *Easy* Level

```
class Song {
    String title;
    String artist;
    int year;
    double duration;
    String genre;
    Song prev, next;

    public Song(String artist, String title, int year, double
duration, String genre) {
        this.artist = artist;
        this.title = title;
        this.year = year;
        this.duration = duration;
        this.genre = genre;
        this.prev = null;
        this.next = null;
    }
}
```

Kelas Song ini digunakan untuk mendefinisikan sebuah objek lagu dengan berbagai atribut yang menjelaskan karakteristik lagu tersebut. Di dalam kelas ini, terdapat beberapa atribut penting: title untuk menyimpan judul lagu, artist untuk menyimpan nama penyanyi atau band, year untuk menyimpan tahun rilis lagu, duration untuk menyimpan durasi lagu dalam menit, dan genre untuk menyimpan genre musik lagu tersebut (seperti Pop, Rock, Hip-Hop, dll). Selain itu, ada dua atribut tambahan, yaitu prev dan next, yang masing-masing digunakan untuk menunjuk ke lagu sebelumnya dan lagu berikutnya. Atribut-atribut prev dan next ini adalah bagian dari mekanisme untuk membangun struktur data yang disebut *linked list*, yang memungkinkan lagu-lagu di playlist disusun secara berurutan dan dapat diakses satu per satu.

Konstruktor di dalam kelas Song digunakan untuk menginisialisasi objek baru dari kelas ini. Konstruktor menerima lima parameter—artist, title, year, duration, dan genre—yang akan diambil nilainya dan disimpan di dalam objek Song yang baru dibuat. Selain itu, konstruktor ini juga menetapkan nilai prev dan next menjadi null, karena saat objek lagu pertama kali dibuat, lagu tersebut tidak memiliki hubungan dengan lagu lain di playlist (belum ada lagu sebelumnya atau berikutnya). Dengan cara ini, setiap objek Song akan menyimpan informasi lengkap tentang lagu serta memiliki kemampuan untuk terhubung

dengan lagu lainnya dalam struktur playlist, yang memungkinkan kita untuk menyusun dan menavigasi koleksi lagu dengan mudah.

```
class Playlist {
    Song head, tail;
    int size;

    public Playlist() {
        this.head = null;
        this.tail = null;
        this.size = 0;
    }

    public void addSong(Song song) {
        if (head == null) {
            head = song;
            tail = song;
        } else {
            tail.next = song;
            song.prev = tail;
            tail = song;
        }
        size++;
    }

    public void bubbleSortByDuration() {
        if (head == null) return;

        boolean swapped;
        do {
            swapped = false;
            Song current = head;
            while (current != null && current.next != null) {
                if (current.duration > current.next.duration) {
                    String tempTitle = current.title;
                    String tempArtist = current.artist;
                    int tempYear = current.year;
                    double tempDuration = current.duration;
                    String tempGenre = current.genre;

                    current.title = current.next.title;
                    current.artist = current.next.artist;
                    current.year = current.next.year;
                    current.duration = current.next.duration;
                    current.genre = current.next.genre;

                    current.next.title = tempTitle;
                    current.next.artist = tempArtist;
```

```
                    current.next.year = tempYear;
                    current.next.duration = tempDuration;
                    current.next.genre = tempGenre;

                    swapped = true;
                }
                current = current.next;
            }
        } while (swapped);
    }

    public void selectionSortByYear() {
        if (head == null) return;

        Song current = head;
        while (current != null) {
            Song maxNode = current;
            Song check = current.next;
            while (check != null) {
                if (check.year > maxNode.year) {
                    maxNode = check;
                }
                check = check.next;
            }
            if (maxNode != current) {
                String tempTitle = current.title;
                String tempArtist = current.artist;
                int tempYear = current.year;
                double tempDuration = current.duration;
                String tempGenre = current.genre;

                current.title = maxNode.title;
                current.artist = maxNode.artist;
                current.year = maxNode.year;
                current.duration = maxNode.duration;
                current.genre = maxNode.genre;

                maxNode.title = tempTitle;
                maxNode.artist = tempArtist;
                maxNode.year = tempYear;
                maxNode.duration = tempDuration;
                maxNode.genre = tempGenre;
            }
            current = current.next;
        }
    }

    public void display() {
```

```
System.out.println("================================================
==========");
        System.out.println("||                          LOL Playlist
||");

System.out.println("================================================
==========");

        Song current = head;
        int index = 1;
        while (current != null) {
            String line1 = index + " | " + current.artist + " - " +
current.title + " (" + current.year + ")";
            String line2 = current.duration + "       " +
current.genre;

            System.out.println(line1);
            System.out.println(line2);

System.out.println("================================================
==========");

            current = current.next;
            index++;
        }
    }

    public void searchByTitle(String searchTitle) {
        int index = 1;
        Song current = head;
        while (current != null) {
            if (current.title.equalsIgnoreCase(searchTitle)) {

System.out.println("================================================
==========");
                System.out.printf("Song with title '%s' found at
position %d\n", searchTitle, index);

System.out.println("================================================
==========");

                String line1 = index + " | " + current.artist + " -
" +  current.title + " (" + current.year + ")";
                String line2 = current.duration + "
" + current.genre;
                System.out.println(line1);
                System.out.println(line2);
```

```
System.out.println("==========================================
==========");
                return;
            }
            current = current.next;
            index++;
        }
    }
}
```

Kode yang diberikan adalah implementasi dari kelas Playlist yang berfungsi untuk mengelola koleksi lagu dalam sebuah playlist. Playlist ini disusun menggunakan struktur data *doubly linked list*, di mana setiap lagu diwakili oleh objek Song, yang berisi atribut-atribut seperti judul lagu, nama artis, tahun rilis, durasi lagu, dan genre. Setiap lagu memiliki dua referensi, yaitu prev yang mengarah ke lagu sebelumnya dan next yang mengarah ke lagu berikutnya. Kelas Playlist itu sendiri memiliki atribut head, tail, dan size, yang masing-masing menunjukkan lagu pertama, lagu terakhir, dan jumlah lagu dalam playlist.

Playlist ini memungkinkan pengguna untuk menambah lagu baru menggunakan metode addSong(), yang akan menambahkan lagu ke akhir playlist. Jika playlist kosong, lagu pertama yang ditambahkan akan menjadi lagu pertama dan terakhir, sementara jika sudah ada lagu lain, lagu baru akan disambungkan ke lagu terakhir melalui referensi next dan prev. Selain itu, playlist juga menyediakan dua algoritma pengurutan lagu, yaitu bubbleSortByDuration() untuk mengurutkan lagu berdasarkan durasi dari yang terpendek hingga yang terpanjang, dan selectionSortByYear() untuk mengurutkan lagu berdasarkan tahun rilis dari yang paling lama hingga yang paling baru.

Metode display() digunakan untuk menampilkan seluruh playlist, menampilkan informasi lagu seperti nama artis, judul lagu, tahun rilis, durasi, dan genre. Sementara itu, metode searchByTitle() memungkinkan pengguna mencari lagu berdasarkan judul. Jika lagu dengan judul yang dicari ditemukan, informasi lagu tersebut akan ditampilkan. Kelebihan dari penggunaan *doubly linked list* di sini adalah kemampuannya untuk menavigasi playlist secara dua arah dan efisiensi dalam menambah atau menghapus lagu. Namun, algoritma pengurutan yang digunakan, yaitu *Bubble Sort* dan *Selection Sort*, memiliki kompleksitas waktu $O(n^2)$, yang bisa membuat pengurutan playlist besar menjadi kurang efisien.

```
public class Main {
    public static void main(String[] args) {
        Playlist playlist = new Playlist();
```

```
        playlist.addSong(new Song("Billy Joel", "Piano Man", 1973,
5.39, "Folk"));
        playlist.addSong(new Song("Bruno Mars", "Lady Gaga", 2024,
4.11, "Pop"));
        playlist.addSong(new Song(".Feast", "Nina", 2024, 4.37,
"Indonesia Rock"));
        playlist.addSong(new Song("Wave to Earth", "Bad", 2023,
4.23, "Korean Rock, Thai Indie"));
        playlist.addSong(new Song("Why Don't We", "8 Lettes", 1970,
4.03, "Rock"));
        playlist.addSong(new Song("The Smiths", "There Is a Light
That Never Goes Out", 1986, 4.4, "Indie"));
        playlist.addSong(new Song("Pamungkas", "To the Bone", 2020,
5.44, "Indonesian Indie"));
        playlist.addSong(new Song("One Ok Rock", "We are", 2017,
4.15, "Pop Rock"));
        playlist.addSong(new Song("Lee Young Ji", "Smart Girl",
2024, 3.9, "R&B"));
        playlist.addSong(new Song("Green Day", "21 Gungs", 2009,
5.21, "Punk Pop"));
        playlist.addSong(new Song("Adele", "Someone Like You", 2011,
4.45, "Pop"));
        playlist.addSong(new Song("Ed Sheeran", "Shape of You",
2017, 4.24, "Pop"));
        playlist.addSong(new Song("Coldplay", "Fix You", 2005, 4.55,
"Alternative Rock"));
        playlist.addSong(new Song("Taylor Swift", "Shake It Off",
2014, 3.39, "Pop"));
        playlist.addSong(new Song("Kendrick Lamar", "HUMBLE.", 2017,
2.57, "Hip-Hop"));
        playlist.addSong(new Song("Sia", "Chandelier", 2014, 3.53,
"Pop"));
        playlist.addSong(new Song("Maroon 5", "Sugar", 2014, 5.02,
"Pop-Rock"));
        playlist.addSong(new Song("Post Malone", "Rockstar", 2017,
3.38, "Hip-Hop"));
        playlist.addSong(new Song("Drake", "God's Plan", 2018, 3.19,
"Hip-Hop"));
        playlist.addSong(new Song("The Weeknd", "Blinding Lights",
2020, 3.20, "Pop"));
        playlist.addSong(new Song("Beyoncé", "Halo", 2008, 4.21,
"Pop-R&B"));
        playlist.addSong(new Song("Billie Eilish", "Bad Guy", 2019,
3.14, "Pop"));
        playlist.addSong(new Song("Imagine Dragons", "Believer",
2017, 3.24, "Rock"));
        playlist.addSong(new Song("Dua Lipa", "Don't Start Now",
2019, 3.03, "Pop"));
```

```
        playlist.addSong(new Song("Drake", "In My Feelings", 2018,
3.37, "Hip-Hop"));
        playlist.addSong(new Song("Shawn Mendes", "Senorita", 2019,
3.11, "Pop"));
        playlist.addSong(new Song("Kanye West", "Stronger", 2007,
5.11, "Hip-Hop"));
        playlist.addSong(new Song("Rihanna", "Diamonds", 2012, 3.45,
"Pop"));
        playlist.addSong(new Song("Lana Del Rey", "Summertime
Sadness", 2012, 4.25, "Pop"));
        playlist.addSong(new Song("Snoop Dogg", "Drop It Like It's
Hot", 2004, 4.30, "Hip-Hop"));
        playlist.addSong(new Song("Lil Nas X", "Old Town Road",
2019, 2.37, "Country-Rap"));
        playlist.addSong(new Song("Travis Scott", "SICKO MODE",
2018, 5.12, "Hip-Hop"));
        playlist.addSong(new Song("Halsey", "Without Me", 2018,
3.21, "Pop"));
        playlist.addSong(new Song("Katy Perry", "Roar", 2013, 3.42,
"Pop"));
        playlist.addSong(new Song("The Chainsmokers", "Closer",
2016, 4.05, "Pop"));
        playlist.addSong(new Song("Marshmello", "Happier", 2018,
3.34, "Electronic"));
        playlist.addSong(new Song("Zedd", "Stay", 2017, 3.30,
"Electronic"));
        playlist.addSong(new Song("Post Malone", "Circles", 2019,
3.35, "Pop-Rock"));
        playlist.addSong(new Song("Ariana Grande", "No Tears Left to
Cry", 2018, 3.25, "Pop"));
        playlist.addSong(new Song("Khalid", "Talk", 2019, 3.17,
"R&B"));
        playlist.addSong(new Song("Megan Thee Stallion", "Savage
Remix", 2020, 4.03, "Hip-Hop"));
        playlist.addSong(new Song("BTS", "Dynamite", 2020, 3.19, "K-
Pop"));
        playlist.addSong(new Song("Drake", "Nonstop", 2018, 3.58,
"Hip-Hop"));
        playlist.addSong(new Song("Lizzo", "Truth Hurts", 2017,
2.53, "Pop"));
        playlist.addSong(new Song("Camila Cabello", "Havana", 2017,
3.37, "Pop"));
        playlist.addSong(new Song("Lil Baby", "Drip Too Hard", 2018,
3.04, "Hip-Hop"));
        playlist.addSong(new Song("Maroon 5", "Girls Like You",
2018, 3.30, "Pop-Rock"));
        playlist.addSong(new Song("Charlie Puth", "Attention", 2017,
3.31, "Pop"));
```

```
        playlist.addSong(new Song("Shakira", "Hips Don't Lie", 2006,
3.38, "Pop"));
        playlist.addSong(new Song("OneRepublic", "Counting Stars",
2013, 4.17, "Pop-Rock"));

        System.out.println("\nTampilan Setelah Lagu Ditambahkan:");
        playlist.display();

        long startTimeBubbleSort = System.nanoTime();
        playlist.bubbleSortByDuration();
        long endTimeBubbleSort = System.nanoTime();
        System.out.println("\nTampilan Sesudah Lagi Di Urutkan
Berdasarkan Menit dan Detik:");
        playlist.display();
        System.out.println("\nBubble Sort memakan waktu: " +
(endTimeBubbleSort - startTimeBubbleSort) + " nanodetik.");

        long startTimeSelectionSort = System.nanoTime();
        playlist.selectionSortByYear();
        long endTimeSelectionSort = System.nanoTime();
        System.out.println("\nTampilan Sesudah Lagu Di Urutkan
Berdasarkan Tahun :");
        playlist.display();
        System.out.println("\nSelection Sort memakan waktu: " +
(endTimeSelectionSort - startTimeSelectionSort) + " nanodetik.");

        String searchTitle = "We Are";
        System.out.println("\n================================");
        System.out.println("Judul lagu yang ingin dicari: " +
searchTitle);

        Song current = playlist.head;
        int index = 1;
        boolean found = false;

        while (current != null) {
            if (current.title.equalsIgnoreCase(searchTitle)) {
                found = true;
                System.out.println("Lagu dengan judul '" +
searchTitle + "' ditemukan pada posisi ke-" + index);

System.out.println("================================");
                String line1 = index + " | " + current.artist + " -
" + current.title + " (" + current.year + ")";
                String line2 = current.duration + " menit    " +
current.genre;
                System.out.println(line1);
                System.out.println(line2);
```

```
System.out.println("================================");
                break;
            }
            current = current.next;
            index++;
        }
        if (!found) {
            System.out.println("Lagu dengan judul '" + searchTitle +
"' tidak ditemukan.");
        }
    }
}
```

Program yang diberikan bertujuan untuk mengelola dan memanipulasi sebuah playlist musik. Di dalamnya, terdapat kelas Song yang menyimpan informasi tentang lagu, seperti nama artis, judul lagu, tahun rilis, durasi, dan genre, serta dua referensi yaitu prev dan next yang membentuk struktur data *doubly linked list* untuk menghubungkan setiap lagu. Kelas Playlist berfungsi untuk mengelola kumpulan lagu-lagu tersebut dengan menyediakan berbagai metode, seperti menambahkan lagu, menampilkan playlist, serta mengurutkan playlist berdasarkan durasi atau tahun rilis. Program ini pertama-tama menginisialisasi sebuah playlist dan menambahkan berbagai lagu ke dalamnya. Setelah itu, playlist ditampilkan sebelum dan sesudah diurutkan menggunakan dua algoritma pengurutan yang berbeda: Bubble Sort untuk mengurutkan berdasarkan durasi dan Selection Sort untuk mengurutkan berdasarkan tahun rilis. Setiap pengurutan disertai dengan penghitungan waktu eksekusinya dalam satuan nanodetik, memberikan gambaran tentang kecepatan masing-masing algoritma. Program juga menyediakan fungsi pencarian untuk menemukan lagu berdasarkan judul yang diberikan pengguna. Jika lagu ditemukan, informasi mengenai lagu tersebut akan ditampilkan beserta posisinya dalam playlist. Secara keseluruhan, program ini menggabungkan pengelolaan playlist, pengurutan data, serta pencarian elemen dalam satu sistem yang terstruktur.

README

Analisis terhadap dua algoritma pengurutan yang digunakan dalam program ini, yaitu **Bubble Sort** untuk pengurutan berdasarkan durasi lagu dan **Selection Sort** untuk pengurutan berdasarkan tahun rilis, menunjukkan bahwa kedua algoritma ini memiliki kekurangan dalam hal efisiensi ketika diterapkan pada dataset yang lebih besar. **Bubble Sort** memiliki kompleksitas waktu $O(n^2)$, yang menyebabkan waktu eksekusinya tumbuh secara eksponensial seiring dengan bertambahnya jumlah elemen, menjadikannya kurang efisien untuk data yang lebih besar. Meskipun demikian, Bubble Sort cukup mudah diimplementasikan dan bisa efektif jika dataset kecil atau hampir terurut. Di sisi lain, **Selection Sort**, meskipun juga memiliki kompleksitas waktu $O(n^2)$, lebih efisien dalam hal jumlah pertukaran yang dilakukan, karena hanya melakukan satu pertukaran per iterasi. Namun, kedua algoritma ini kurang optimal untuk data dalam jumlah besar. Untuk mengatasi masalah ini, algoritma pengurutan yang lebih efisien seperti **Merge Sort** atau **Quick Sort**, yang memiliki kompleksitas waktu O(n log n), dapat digunakan untuk mempercepat proses pengurutan pada dataset besar, menghasilkan solusi yang lebih efektif dan efisien.