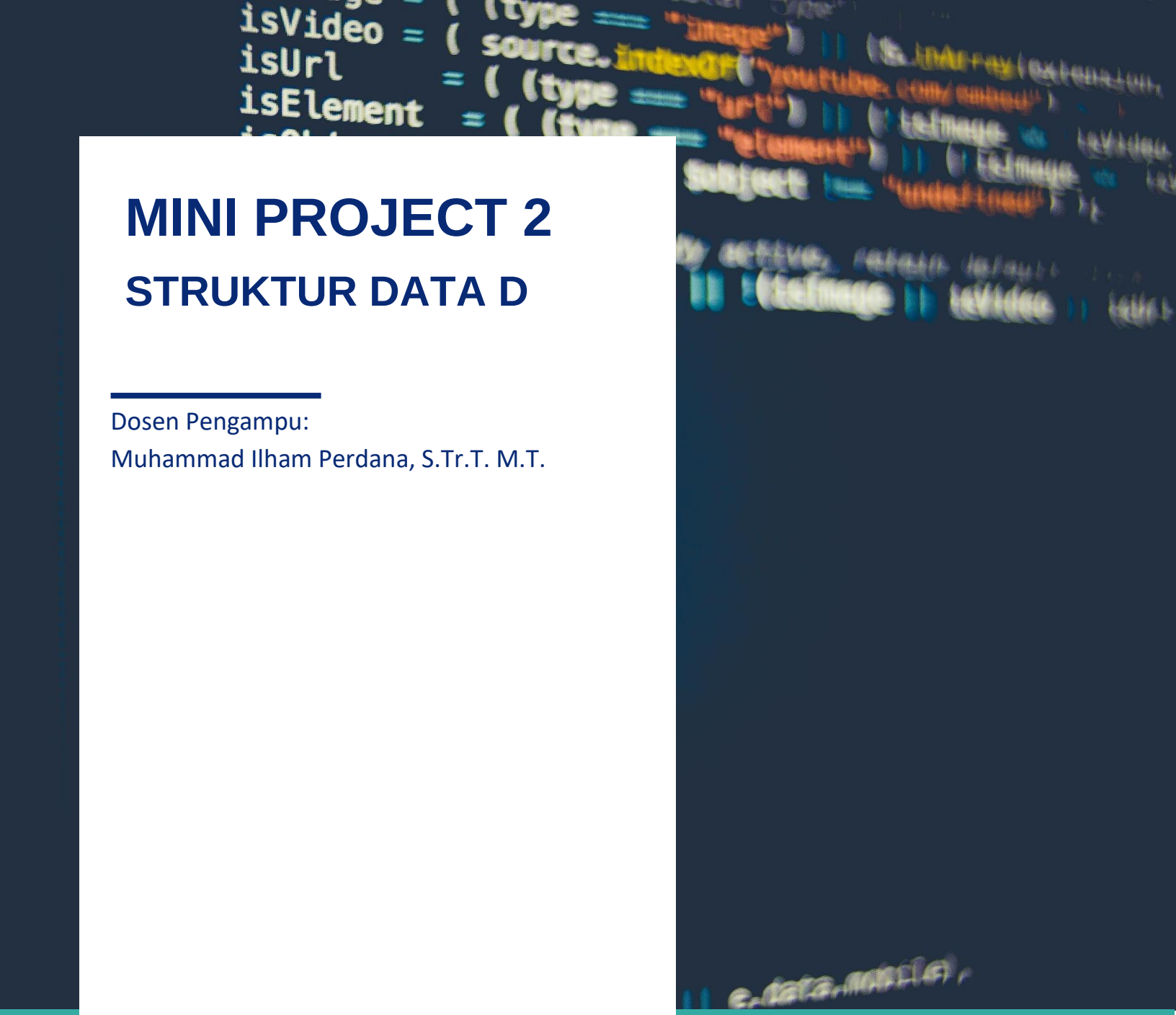


MINI PROJECT 2

STRUKTUR DATA D

Dosen Pengampu:
Muhammad Ilham Perdana, S.Tr.T. M.T.

Source Code:
https://github.com/hisyam99/mini-project-strukdat-2024/tree/main/MINI_PROJECT_2

Muhammad Hisyam Kamil
202210370311060

Daftar Isi

Program Image Color Analyzer

Source Code: https://github.com/hisyam99/mini-project-strukdat-2024/tree/main/MINI_PROJECT_2

1. Pendahuluan

Latar Belakang:

Nasib sial menimpa seorang mahasiswa DKV. Ia terlambat memasuki kelas karena malamnya terlalu asik push rank Mobile Legend dan push MMR Moskov untuk menjadi no 1 di Malang. Ia diperbolehkan masuk kelas asalkan menyelesaikan hukuman yang diberikan oleh dosen. Hukumannya adalah mengidentifikasi warna apa saja yang terdapat pada suatu gambar, menghitung berapa kali warna tersebut muncul (per-pixel), dan warna apa yang dominan. Ia kebingungan karena hukumannya sungguh sangat mustahil. Beruntungnya, tiba-tiba temannya yang merupakan mahasiswa informatika datang menghampiri. Mendengar keluhan temannya, Si mahasiswa informatika bersedia untuk membantu menyelesaikan hukuman tersebut.

Deskripsi Project:

Diberikan suatu gambar. Buatlah sebuah program untuk mengidentifikasi jenis warna apa saja yang muncul pada suatu gambar, hitung frekuensi kemunculan warna-warna tersebut, dan tentukan warna apa yang dominan. Jangan lupa aplikasikan struktur data hashmap.

Syarat:

1. Implementasikan program menggunakan Java dengan memanfaatkan struktur data HashMap untuk menyimpan informasi tentang frekuensi kemunculan setiap warna pada gambar.
2. Program harus dapat memuat gambar dalam format tertentu (misalnya, JPEG atau PNG).

3. Hitung frekuensi kemunculan setiap warna pada gambar dan simpan informasi tersebut dalam HashMap.
4. Tampilkan warna apa saja yang ada pada gambar tersebut
5. Tentukan warna yang paling dominan berdasarkan frekuensi kemunculannya.
6. Gunakan gambar yang tersedia pada file RAR ini. Untuk NIM ganjil gunakan Gambar 1 dan NIM Genap gunakan gambar 2. Perhatikan untuk hanya menggunakan 2 gambar ini. Jumlah dan warna dominan salah = Nilai Tugas 50%
7. Berikan deskripsi pada setiap baris program

2. Pembahasan

Konstruktor ImageColorAnalyzer()

```
/**
 * Konstruktor untuk ImageColorAnalyzer.
 */
public ImageColorAnalyzer() {
    this.colorFrequencyMap = new HashMap<>(); // Inisialisasi HashMap untuk
frekuensi warna
}
```

- Deskripsi: Konstruktor inisialisasi objek ImageColorAnalyzer.
- Penjelasan Detail:
  - this.colorFrequencyMap = new HashMap<>();: Membuat objek HashMap untuk menyimpan frekuensi warna. this digunakan untuk merujuk ke variabel instans colorFrequencyMap dari kelas saat ini.
- Implementasi: Menciptakan objek HashMap yang akan digunakan untuk menyimpan frekuensi masing-masing warna dalam gambar.

Method readAndProcessImage(String imagePath)

```
/**
 * Metode untuk membaca dan memproses gambar.
 * @param imagePath Path gambar yang akan dianalisis.
 * @throws IOException Jika terjadi kesalahan saat membaca gambar.
 */
public void readAndProcessImage(String imagePath) throws IOException {
    File file = new File(imagePath); // Membuat objek File berdasarkan path gambar
    // Gambar yang akan dianalisis
    BufferedImage image = ImageIO.read(file); // Membaca gambar dari file

    // Memproses setiap piksel dalam gambar
    for (int x = 0; x < image.getWidth(); x++) {
        for (int y = 0; y < image.getHeight(); y++) {
            // Mendapatkan nilai RGB dari piksel pada posisi (x, y)
            // Fungsi getRGB() mengembalikan nilai dalam format ARGB, kita gunakan
```

```
& 0xFFFFFF untuk memisahkan kanal warna saja
            int color = image.getRGB(x, y) & 0xFFFFFF; // Mask alpha channel

            // Memasukkan atau memperbarui frekuensi warna dalam HashMap
            // Menggunakan metode getOrDefault untuk mendapatkan nilai saat ini
dari HashMap
            // Jika warna belum ada dalam HashMap, default-nya adalah 0
            colorFrequencyMap.put(color, colorFrequencyMap.getOrDefault(color, 0) +
1);
        }
    }
}
```

- Deskripsi: Metode ini membaca gambar dari path yang diberikan, mengekstraksi warna dari setiap piksel, dan menghitung frekuensi masing-masing warna.
- Penjelasan Detail:
  - File file = new File(imagePath);: Membuat objek File untuk merepresentasikan gambar yang akan dibaca.
  - BufferedImage image = ImageIO.read(file);: Membaca gambar dari file dan menyimpannya dalam objek BufferedImage.
  - for (int x = 0; x < image.getWidth(); x++) { ... }: Looping melalui semua piksel di gambar, dengan x dan y sebagai koordinat.
  - int color = image.getRGB(x, y) & 0xFFFFFF;: Mendapatkan nilai warna piksel pada koordinat (x, y), mengabaikan saluran alpha.
  - colorFrequencyMap.put(color, colorFrequencyMap.getOrDefault(color, 0) + 1);: Menambahkan atau memperbarui frekuensi warna dalam HashMap.
- Implementasi: Membaca gambar, mengekstraksi warna piksel, dan menghitung frekuensi warna dalam gambar.

Method getDominantColor()

```
/**
 * Metode untuk mendapatkan warna dominan berdasarkan frekuensinya.
 * @return Warna dominan dalam format integer RGB.
 */
public int getDominantColor() {
    int dominantColor = 0; // Variabel untuk menyimpan warna dominan
    int maxFrequency = 0; // Variabel untuk menyimpan frekuensi maksimum

    // Menemukan warna dominan dengan frekuensi tertinggi
    for (Map.Entry<Integer, Integer> entry : colorFrequencyMap.entrySet()) {
        // Jika frekuensi warna saat ini lebih besar dari frekuensi maksimum yang
tersimpan
        if (entry.getValue() > maxFrequency) {
            // Perbarui frekuensi maksimum dengan frekuensi warna saat ini
            maxFrequency = entry.getValue();
            // Perbarui warna dominan dengan warna saat ini
            dominantColor = entry.getKey();
```

```
        }
    }

    return dominantColor; // Mengembalikan warna dominan
}
```

- Deskripsi: Metode ini mencari warna dengan frekuensi terbanyak dalam gambar, yang dianggap sebagai warna dominan.
- Penjelasan Detail:
  - for (Map.Entry<Integer, Integer> entry : colorFrequencyMap.entrySet()) { ... }: Looping melalui semua entri dalam HashMap.
  - if (entry.getValue() > maxFrequency) { ... }: Memeriksa apakah frekuensi warna saat ini lebih besar dari frekuensi maksimum yang tersimpan.
  - maxFrequency = entry.getValue();: Memperbarui frekuensi maksimum dengan frekuensi warna saat ini.
  - dominantColor = entry.getKey();: Memperbarui warna dominan dengan warna saat ini.
- Implementasi: Mencari warna dominan dalam gambar berdasarkan frekuensinya.

Method getMaxFrequency()

```
/**
 * Metode untuk mendapatkan frekuensi maksimum dari warna dominan.
 * @return Frekuensi maksimum dari warna dominan.
 */
public int getMaxFrequency() {
    int maxFrequency = 0; // Variabel untuk menyimpan frekuensi maksimum

    // Iterasi melalui nilai frekuensi dalam HashMap
    for (int frequency : colorFrequencyMap.values()) {
        if (frequency > maxFrequency) {
            maxFrequency = frequency;
        }
    }

    return maxFrequency; // Mengembalikan frekuensi maksimum
}
```

- Deskripsi: Metode ini mengembalikan frekuensi maksimum dari warna dominan yang ditemukan dalam gambar.
- Penjelasan Detail:
  - for (int frequency : colorFrequencyMap.values()) { ... }: Looping melalui semua nilai frekuensi dalam HashMap.
  - if (frequency > maxFrequency) { ... }: Memeriksa apakah frekuensi warna saat ini lebih besar dari frekuensi maksimum yang tersimpan.

- maxFrequency = frequency;: Memperbarui frekuensi maksimum dengan frekuensi warna saat ini.

- Implementasi: Mendapatkan frekuensi maksimum dari warna dominan dalam gambar.

Method displayResults()

```
/**
 * Metode untuk menampilkan hasil analisis warna.
 */
public void displayResults() {
    int dominantColor = getDominantColor(); // Dapatkan warna dominan
    int maxFrequency = getMaxFrequency(); // Dapatkan frekuensi maksimum

    // Tampilkan hasil dalam format heksadesimal
    System.out.printf("Warna dominan pada gambar adalah: #%06X%n", dominantColor);
    // Menampilkan jumlah frekuensi warna dominan
    System.out.printf("Jumlah frekuensi warna #%06X: %d Pixel%n", dominantColor,
maxFrequency);
}
```

- Deskripsi: Metode ini menampilkan hasil analisis warna ke konsol.
- Penjelasan Detail:
  - int dominantColor = getDominantColor();: Mendapatkan warna dominan.
  - int maxFrequency = getMaxFrequency();: Mendapatkan frekuensi maksimum warna dominan.
  - System.out.printf("Warna dominan pada gambar adalah: #%06X%n", dominantColor);: Menampilkan warna dominan dalam format heksadesimal.
  - System.out.printf("Jumlah frekuensi warna #%06X: %d Pixel%n", dominantColor, maxFrequency);: Menampilkan jumlah frekuensi warna dominan.
- Implementasi: Menampilkan hasil analisis warna ke konsol.

Method main(String[] args)

```
/**
 * Metode utama untuk menjalankan program.
 * @param args Argumen baris perintah, di sini tidak digunakan.
 */
public static void main(String[] args) {
    ImageColorAnalyzer colorAnalyzer = new ImageColorAnalyzer(); // Buat objek
analisis warna
    Scanner scanner = new Scanner(System.in); // Membuat objek Scanner untuk input

    System.out.println("Selamat datang di Program Deteksi Warna Dominan pada
Gambar!");
```

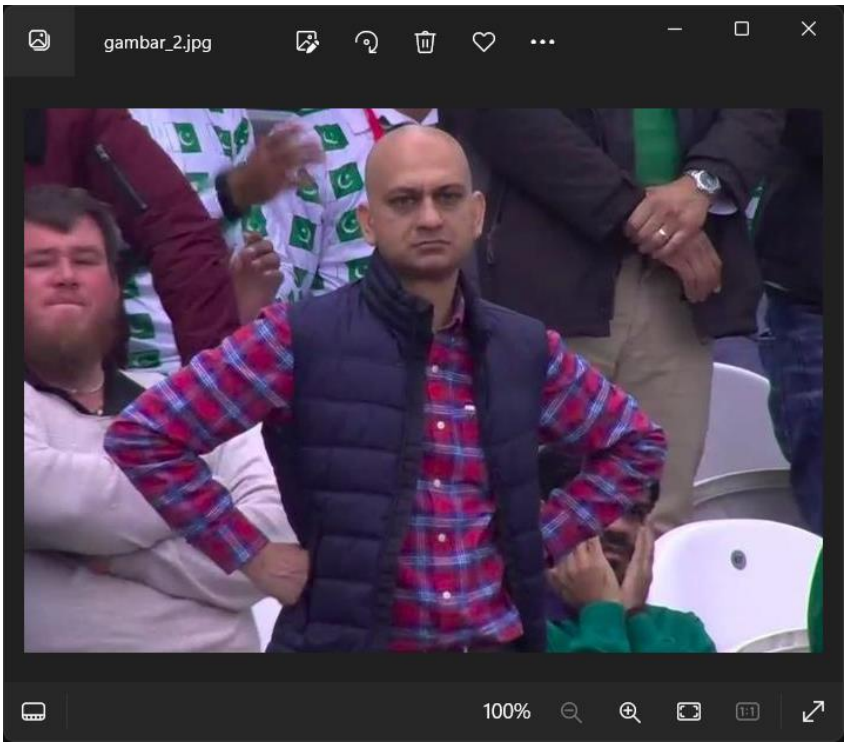
```
    System.out.print("Masukkan path gambar: ");
    String imagePath = scanner.nextLine(); // Membaca path gambar dari pengguna

    try {
        colorAnalyzer.readAndProcessImage(imagePath); // Memproses gambar untuk
mendapatkan frekuensi warna
        colorAnalyzer.displayResults(); // Tampilkan hasil analisis warna
    } catch (IOException e) {
        System.err.println("Gagal memuat gambar. Pastikan path yang dimasukkan
benar."); // Tangani kesalahan IO
        e.printStackTrace(); // Cetak tumpukan kesalahan
    } finally {
        scanner.close(); // Tutup Scanner untuk membersihkan penggunaan memory
untuk scanner
    }
}
```

- Deskripsi: Metode main sebagai titik masuk program, meminta input path gambar, memproses gambar, dan menampilkan hasil analisis warna.
- Penjelasan Detail:
    - ImageColorAnalyzer colorAnalyzer = new ImageColorAnalyzer();: Membuat objek ImageColorAnalyzer.
    - Scanner scanner = new Scanner(System.in);: Membuat objek Scanner untuk menerima input dari pengguna.
    - String imagePath = scanner.nextLine();: Meminta pengguna untuk memasukkan path gambar.
    - colorAnalyzer.readAndProcessImage(imagePath);: Memproses gambar yang diberikan.
    - colorAnalyzer.displayResults();: Menampilkan hasil analisis warna.
    - catch (IOException e) { ... }: Menangkap dan menangani kesalahan IO yang mungkin terjadi saat membaca gambar.
    - finally { scanner.close(); }: Menutup objek Scanner setelah selesai digunakan.
- Implementasi: Menjalankan program untuk menganalisis warna dominan pada gambar yang diberikan.

3. Full Source Code

https://github.com/hisyam99/mini-project-strukdat-2024/blob/main/MINI_PROJECT_2/src/main/java/org/project/v2/ImageColorAnalyzer.java

4. Output Program



Diberikan file gambar diatas dengan nama “gambar_2.jpg”, kita ingin menentukan warna apa saja yang dominan, serta jumlah frekuensi kemunculan warna tersebut dengan menggunakan program yang sudah kita buat tadi, maka outputnya sebagai berikut: