Nama : Afad Fath Musyarof Halim
NIM : 2211104030
Kelas : SE06-01

1. **Bad Smell**
    a. **Long Method**

        Method printInfo() terlalu panjang dan dapat dipecah menjadi method yang lebih kecil seperti printArtistInfo atau printAlbumInfo

    b. **Primitive Obsession**

        Variable genre menyimpan dengan tipe data int sehingga data yang tersimpan tetaplah dalam bentuk int dan tidak dengan nama genre tersebut, dapat di perbaiki dengan menggunakan enum sehingga nama genre dan urutan index nya dapat disimpan

    c. **Data Clump & Feature Envy**

        Variable untuk Album dan Artist dapat dipisah menjadi class lain sehingga method get dan set juga dapat di perjelas untuk bagian apa

    d. **Magic Number**

        printInfo menggunakan parameter detailLevel dengan tipe data int sehingga akan memunculkan angka namun angka tersebut kurang di jelaskan, dapat di perbaiki dengan menggunakan enum untuk membagi detailLevel dan informasi setiap level

    e. **Comment**

        Terdapat 2 penggunaan komentar untuk param genre padahal salah satunya di gunakan untuk param yang berbeda (genre dan detailLevel), serta karena genre dan detailLevel berupa enum maka komentar berisi penjelasan hanya memperbanyak kode

2. **Kode sumber**
    a. **Sebelum Refactoring**

```
package Assignment;


public class Song {

        private String id;
        private String title;
        private String releaseYear;
        private String musicFileURL;
        private int genre;

        private String albumName;
```

```
	private String albumCoverURL;

	private String artistName;
	private String artistAlias;
	private String artistImageURL;

	public Song(String id, String title, String releaseYear, String
musicFileURL) {
		this.id = id;
		this.title = title;
		this.releaseYear = releaseYear;
		this.musicFileURL = musicFileURL;
	}

	public void setAlbum(String albumName, String
albumCoverURL) {
		this.albumName = albumName;
		this.albumCoverURL = albumCoverURL;
	}

	public void setArtist(String artistName, String artistAlias,
String artistImageURL) {
		this.artistName = artistName;
		this.artistAlias = artistAlias;
		this.artistImageURL = artistImageURL;
	}

	/**
	 * Set the genre of this song
	 *
	 * 0 = undefined
	 * 1 = pop
	 * 2 = rock
	 * 3 = hip hop
	 * 4 = RnB
	 * 5 = jazz
	 * 6 = instrumentals
	 * 7 = clowncore
	 *
	 * @param genre
	 */
	public void setGenre(int genre) {
		this.genre = genre;
	}
```

```
	/**
	 * Print info of the song based on desired detail level
	 *
	 * 0 = song info only
	 * 1 = song info and artist info
	 * 2 = song info and album info
	 * 3 = song, artist, and album info
	 *
	 * @param genre
	 */
	public void printInfo(int detailLevel) {
		if (detailLevel == 0) {
			System.out.println("song title: " + title);
			System.out.println("release year: " +
releaseYear);
			if (genre > 0) {
				System.out.println("genre: " + genre);
			}
		}else if(detailLevel == 1) {
			System.out.println("song title: " + title);
			System.out.println("release year: " +
releaseYear);
			if (genre > 0) {
				System.out.println("genre: " + genre);
			}
			if (!artistName.equals("")) {
				System.out.println("artist name: " +
artistName);
			}
			if (!artistAlias.equals("")) {
				System.out.println("artist also known
as: " + artistAlias);
			}
		}else if (detailLevel == 2) {
			System.out.println("song title: " + title);
			System.out.println("release year: " +
releaseYear);
			if (genre > 0) {
				System.out.println("genre: " + genre);
			}
			if (!albumName.equals("")) {
				System.out.println("album title: " +
albumName);
```

```
                }
            }else if (detailLevel == 3) {
                System.out.println("song title: " + title);
                System.out.println("release year: " +
releaseYear);
                if (genre > 0) {
                    System.out.println("genre: " + genre);
                }
                if (!artistName.equals("")) {
                    System.out.println("artist name: " +
artistName);
                }
                if (!artistAlias.equals("")) {
                    System.out.println("artist also known
as: " + artistAlias);
                }
                if (!albumName.equals("")) {
                    System.out.println("album title: " +
albumName);
                }
            }
        }

}
```

b. Setelah Refactoring

```
package Assignment;

enum Genre {
  UNDEFINED, POP, ROCK, HIPHOP, RNB, JAZZ, INSTRUMENTALS,
CLOWNCORE;
}

enum DetailLevel {
  SONG_ONLY, SONG_AND_ARTIST, SONG_AND_ALBUM,
FULL_DETAIL;
}

class Album {
  private String name;
  private String coverURL;

  public Album(String name, String coverURL) {
```

```
        this.name = name;
        this.coverURL = coverURL;
    }

    public String getName() {
        return name;
    }
}

class Artist {
    private String name;
    private String alias;
    private String imageURL;

    public Artist(String name, String alias, String imageURL) {
        this.name = name;
        this.alias = alias;
        this.imageURL = imageURL;
    }

    public String getName() {
        return name;
    }

    public String getAlias() {
        return alias;
    }
}

public class Song {
    private String id;
    private String title;
    private String releaseYear;
    private String musicFileURL;
    private Genre genre = Genre.UNDEFINED;
    private Album album;
    private Artist artist;

    public Song(String id, String title, String releaseYear, String
musicFileURL) {
        this.id = id;
        this.title = title;
        this.releaseYear = releaseYear;
        this.musicFileURL = musicFileURL;
```

```
    }

    public void setAlbum(Album album) {
      this.album = album;
    }

    public void setArtist(Artist artist) {
      this.artist = artist;
    }

    public void setGenre(Genre genre) {
      this.genre = genre;
    }

    public void printInfo(DetailLevel detailLevel) {
      printBasicInfo();
      if (detailLevel == DetailLevel.SONG_AND_ARTIST
        || detailLevel == DetailLevel.FULL_DETAIL) {
        printArtistInfo();
      }
      if (detailLevel == DetailLevel.SONG_AND_ALBUM
        || detailLevel == DetailLevel.FULL_DETAIL) {
        printAlbumInfo();
      }
    }

    private void printBasicInfo() {
      System.out.println("Song Title: " + title);
      System.out.println("Release Year: " + releaseYear);
      if (genre != Genre.UNDEFINED) {
        System.out.println("Genre: " + genre);
      }
    }

    private void printArtistInfo() {
      if (artist != null) {
        System.out.println("Artist Name: " + artist.getName());
        if (!artist.getAlias().isEmpty()) {
          System.out.println("Also known as: " + artist.getAlias());
        }
      }
    }

    private void printAlbumInfo() {
```

```
            if (album != null) {
                System.out.println("Album Title: " + album.getName());
            }
        }
    }
```