VERSION 1.0

JULI 21, 2023

PEMROGRAMAN LANJUT

MODUL 4 – API (APPLICATION PROGRAMMING INTERFACE)

TIM PENYUSUN:
- WILDAN SUHARSO, S.KOM., M.KOM
- LARYNT SAWFA KENANGA
- AHYA NIKA SALSABILA

LAB. INFORMATIKA

UNIVERSITAS MUHAMMADIYAH MALANG

PEMROGRAMAN LANJUT

PERSIAPAN MATERI

Mahasiswa harus memahami konsep OOP yang telah dipelajari sebelumnya

TUJUAN

1. Mahasiswa mampu memahami konsep Java API dan mengenali kegunaan serta manfaatnya.
2. Mahasiswa mampu menggunakan kelas-kelas dari Java API untuk memperluas fungsionalitas program yang dibangun.
3. Mahasiswa mampu membuat program Java dengan API

TARGET MODUL

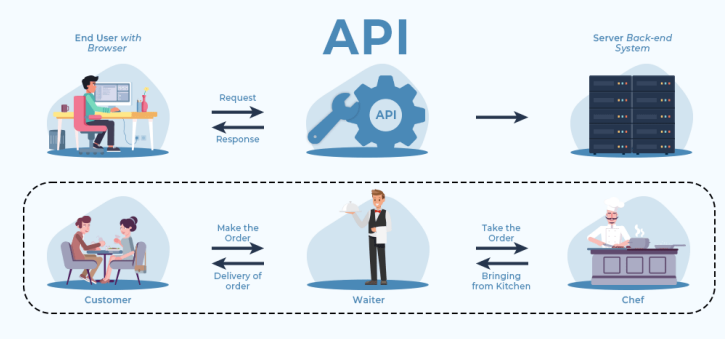
1. Mahasiswa mampu memahami API.
2. Mahasiswa mampu membuat program Java dengan API.

PERSIAPAN SOFTWARE/APLIKASI

1. Java Development Kit
2. Text Editor / IDE (**Preferably** Intellij IDEA, Visual Studio Code, Netbeans, etc).

MATERI POKOK

A. APA ITU JAVA API

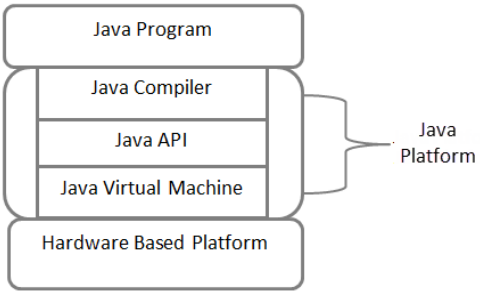


API (Application Programming Interface) adalah sekumpulan kode pemrograman yang membantu pengembang melakukan integrasi data antara dua aplikasi berbeda secara bersamaan. Dalam Java, API merupakan komponen perangkat lunak penting yang disertakan dalam JDK (Java Development Kit). API di Java mencakup kelas, antarmuka, dan antarmuka pengguna. Melalui API, pengembang dapat mengintegrasikan berbagai aplikasi dan situs web serta menyediakan informasi secara real-time.

Secara umum, API adalah antarmuka perangkat lunak yang memungkinkan dua aplikasi berkomunikasi tanpa harus melibatkan interaksi langsung dari pengguna. Hal ini mencakup sekelompok program komputer dan operasi yang memungkinkan pertukaran data dan komunikasi antara dua program perangkat lunak. API dapat dianggap sebagai kumpulan protokol komunikasi dan subrutin yang digunakan oleh berbagai program untuk berkomunikasi satu sama lain.

Dengan menggunakan API, pengembang dapat mempermudah proses pembuatan aplikasi dengan mengakses berbagai fungsi dan alat tanpa harus menulis kode yang rumit dari awal. API juga memfasilitasi para pengembang dengan cara efisien untuk mengembangkan perangkat lunak mereka. Selain itu, API memungkinkan pengembang untuk membuat antarmuka yang dapat diakses oleh pengguna untuk mengakses layanan tertentu.

B. MEMAHAMI API DI JAVA



Java Development Kit (JDK) terdiri dari tiga komponen utama, yaitu:

1. Java compiler
2. Java Virtual Machine (JVM)
3. Java Application Programming Interface (API)

Java API yang ditambahkan ke JDK menjelaskan fungsi dari setiap elemen. Dalam pemrograman Java, beberapa komponen telah dibuat sebelumnya dan secara luas digunakan. Oleh karena itu, seorang pengembang dapat menggunakan program yang sudah ditulis sebelumnya dengan menggunakan Java API. Setelah menyebutkan kelas dan paket API yang tersedia, pengembang dengan cepat dapat membuat kelas dan paket program yang diperlukan untuk dieksekusi.

Java API adalah elemen penting dari JDK dan mengidentifikasi fitur dari setiap elemen. Ketika kita memprogram sesuatu dalam bahasa Java, komponen ini (Java API) sudah diproduksi dan selesai. Melalui Java, seorang pengembang API dapat menggunakan program yang sudah ditulis sebelumnya. Pengembang pertama-tama mendeklarasikan kelas dan paket, kemudian pengembang tersebut dapat menggunakan program aplikasi kelas dan paket untuk dieksekusi.

C. SIAPA YANG MENGGUNAKAN JAVA API

Terdapat tiga jenis pengembang yang menggunakan API Java berdasarkan pekerjaan atau proyek mereka:

1. Internal developers
2. Partner developers
3. Open developers

D. MENGAPA KITA BUTUH API DI JAVA

Java API sangat penting dalam pengembangan perangkat lunak karena memberikan berbagai manfaat dan kemudahan bagi para pengembang. Berikut adalah alasan mengapa Java API dibutuhkan:

1. Efisien

   API yang dibuat oleh aplikasi dari pihak ketiga dapat sangat membantu mengurangi pekerjaan di perusahaan. Hal ini membuat proses pembuatan aplikasi menjadi lebih cepat. Perusahaan bisa meminta bantuan dari pihak ketiga untuk beberapa bagian pekerjaan dengan biaya yang lebih rendah dibandingkan membangun aplikasi sendiri.

2. API membuat segalanya lebih sederhana

   API memudahkan logika kompleks dengan memecahnya menjadi bagian-bagian yang lebih kecil. Mereka juga menyediakan cara yang mudah untuk mengakses data sesuai dengan kebutuhan pengguna. API dapat memberikan data yang kita butuhkan tanpa harus melakukan penelitian atau manipulasi tambahan, sehingga mempercepat proses pembuatan aplikasi.

E. KEUNTUNGAN DAN KERUGIAN PENGGUNAAN API

Keuntungan API	Kerugian API
Efisiensi: API menghasilkan hasil yang efisien, lebih cepat, dan lebih andal daripada hasil yang dihasilkan secara manual.	**Biaya:** Pengembangan dan implementasi API kadang-kadang mahal dan memerlukan dukungan dan pemeliharaan tinggi dari para pengembang.
Pengiriman Layanan yang Fleksibel: API menyediakan pengiriman layanan yang cepat dan fleksibel sesuai dengan kebutuhan para pengembang.	**Masalah Keamanan:** Menggunakan API menambahkan lapisan permukaan lain yang rentan terhadap serangan, sehingga masalah risiko keamanan umum terjadi pada API.

Integrasi: Fitur terbaik dari API adalah memungkinkan perpindahan data antara berbagai situs dan dengan demikian meningkatkan pengalaman pengguna yang terintegrasi.	
Otomatisasi: Karena API menggunakan komputer robotik daripada manusia, hasil yang dihasilkan lebih baik dan lebih otomatis.	
Fungsionalitas Baru: Saat menggunakan API, para pengembang menemukan alat dan fungsionalitas baru untuk pertukaran API.	

F. BAGIAN UTAMA JAVA API

Java API memiliki 3 bagian utama, yaitu :

1. Java Macro Edition (ME): Merancang aplikasi yang dijalankan pada perangkat kecil seperti telepon genggam.
2. Java Standard Edition (SE): API yang standar dengan tujuan merancang aplikasi dekstop menggunakan bahasa dasar yang mendukung keamanan, grafis, serta konektivitas data dan jaringan.
3. Java Enterprise Edition (EE): API untuk merancang aplikasi server yang mendukung basis data.

G. FITUR JAVA API

1. Java Server Pages: Java Server Pages merupakan teknologi web berbasis Java yang berjalan pada platform Java.
2. Java Database Connectivity: Java Database Connectivity dirancang untuk mengakses database universal berdasarkan SQL.
3. Java Card: Java Card adalah open standar dari Sun Microsystems untuk platform pengembalian kartu pintar.
4. Applet: Applet adalah program yang diakses melalui halaman web. Program ini di download lalu dapat dijalankan pada jendela menjelajah web.
5. Java Networking: Java Networking adalah konsep menghubungkan dua atau lebih perangkat komputasi bersama.

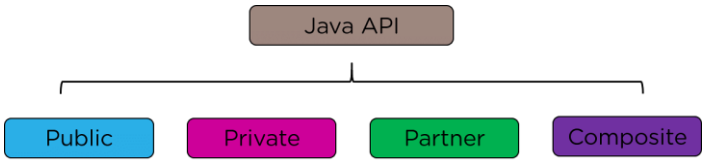
H. CARA KERJA JAVA API

1. Aplikasi klien memulai permintaan API yang disebut API call untuk mendapatkan informasi. Permintaan ini, yang mencakup kata kerja permintaan, header, dan kadang-kadang badan permintaan dari aplikasi ke server web, dihandle oleh URI (Uniform Resource Identifier) API.
2. Setelah menerima permintaan yang sah, API menghubungi program eksternal atau server web.
3. API menerima respons dari server yang berisi data yang diminta.
4. Data dikirimkan dari API ke aplikasi awal yang memintanya.

I. DASAR-DASAR API

1. Membangun API untuk merespons OK kemudian menyediakan string atau konten web.
2. Merespons dan mengonsumsi JSON/XML.
3. Menangani pengiriman formulir (form submissions).
4. Mengonsumsi data (dalam bentuk form, JSON, atau XML), memvalidasi data, dan menyimpannya dalam database.
5. Menghubungkan ke API lain.
6. Menyimpan data ke berbagai penyimpanan data SQL/NoSQL.
7. Mengubah data dalam database.
8. Menghapus data dalam database.
9. Menjamin keamanan API Anda.

J. TIPE JAVA API



1. Public

   API Public (atau terbuka) adalah API Java yang disertakan dalam JDK. Mereka tidak memiliki batasan ketat tentang bagaimana para pengembang menggunakannya.

2. Private

   API Private (atau internal) dikembangkan oleh organisasi tertentu dan hanya dapat diakses oleh karyawan yang bekerja untuk organisasi tersebut.

3. Partner

   API Partner dianggap sebagai API pihak ketiga dan dikembangkan oleh organisasi untuk operasi bisnis strategis.

4. Composite

API Composite adalah mikro layanan, dan para pengembang membangunnya dengan menggabungkan beberapa API layanan.

K. CONTOH JAVA API

Berikut adalah beberapa contoh Java API yang umum digunakan:

1. Java API untuk Manajemen Koleksi (Java Collections Framework):
   - java.util.List: Mengimplementasikan sebuah daftar berurut dari elemen.

```java
import java.util.ArrayList;
import java.util.List;

public class ListExample {
    public static void main(String[]
args) {
        List<String> list = new
ArrayList<>();
        list.add("Apple");
        list.add("Banana");
        list.add("Orange");

        System.out.println(list);  //
Output: [Apple, Banana, Orange]
    }
}
```

   - java.util.Set: Mengimplementasikan kumpulan elemen unik.

```java
import java.util.HashSet;
import java.util.Set;

public class SetExample {
    public static void main(String[] args) {
        Set<String> set = new HashSet<>();
        set.add("Apple");
        set.add("Banana");
        set.add("Orange");

        System.out.println(set);  // Output: [Apple, Banana, Orange]
    }
}
```

   - java.util.Map: Mengimplementasikan pasangan kunci-nilai.

```java
import java.util.HashMap;
import java.util.Map;

public class MapExample {
    public static void main(String[] args) {
        Map<Integer, String> map = new HashMap<>();
        map.put(1, "Apple");
        map.put(2, "Banana");
        map.put(3, "Orange");

        System.out.println(map);  // Output: {1=Apple, 2=Banana, 3=Orange}
    }
}
```

2. Java API untuk Manajemen I/O (Input/Output):

- java.io.InputStream: Membaca byte dari input stream.

```java
import java.io.FileInputStream;
import java.io.InputStream;

public class InputStreamExample {
    public static void main(String[] args) {
        try {
            InputStream inputStream = new FileInputStream("example.txt");
            int data = inputStream.read();

            while (data != -1) {
                System.out.print((char) data);
                data = inputStream.read();
            }

            inputStream.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

- java.io.OutputStream: Menulis byte ke output stream.

```java
import java.io.FileOutputStream;
import java.io.OutputStream;

public class OutputStreamExample {
    public static void main(String[] args) {
        try {
            OutputStream outputStream = new FileOutputStream("example.txt");
            String data = "Hello, World!";
            byte[] bytes = data.getBytes();

            outputStream.write(bytes);

            outputStream.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

- java.io.File: Merepresentasikan file atau direktori dalam sistem file.

```java
import java.io.File;

public class FileExample {
    public static void main(String[] args) {
        File file = new File("example.txt");

        System.out.println("File Name: " + file.getName());
        System.out.println("File Path: " + file.getAbsolutePath());
        System.out.println("File Size: " + file.length() + " bytes");
    }
}
```

3. Java API untuk Manajemen Jaringan:
   - java.net.URL: Merepresentasikan alamat URL dan dapat digunakan untuk membuka koneksi ke sumber daya jaringan.

```java
import java.net.URL;
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class URLExample {
    public static void main(String[] args) {
        try {
            URL url = new URL("https://www.example.com");
            BufferedReader reader = new BufferedReader(new InputStreamReader(url.openStream()));
            String line;

            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }

            reader.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

- java.net.HttpURLConnection: Mengizinkan koneksi HTTP ke server dan dapat digunakan untuk mengirim permintaan HTTP dan menerima respons.

```java
import java.net.HttpURLConnection;
import java.net.URL;

public class HttpURLConnectionExample {
    public static void main(String[] args) {
        try {
            URL url = new URL("https://www.example.com");
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setRequestMethod("GET");

            int responseCode = connection.getResponseCode();
            System.out.println("Response Code: " + responseCode);

            connection.disconnect();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

4. Java API untuk Manajemen Konkurensi:
   - java.util.concurrent.Executor: Mengeksekusi tugas pada thread terpisah.

```java
import java.util.concurrent.Executor;
import java.util.concurrent.Executors;

public class ExecutorExample {
    public static void main(String[] args) {
        Executor executor = Executors.newSingleThreadExecutor();

        executor.execute(() -> {
            System.out.println("Task executed on a separate thread.");
        });
    }
}
```

- java.util.concurrent.Lock: Mengontrol akses ke bagian kode yang dapat dijalankan secara bersamaan (thread-safe).

```java
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;

public class LockExample {
    private Lock lock = new ReentrantLock();

    public void performTask() {
        lock.lock();
        try {
            // Bagian kode yang membutuhkan akses eksklusif
            // ...
        } finally {
            lock.unlock();
        }
    }

    public static void main(String[] args) {
        LockExample example = new LockExample();

        Thread thread1 = new Thread(() -> {
            example.performTask();
        });
        Thread thread2 = new Thread(() -> {
            example.performTask();
        });

        thread1.start();
        thread2.start();
    }
}
```

5. Java API untuk GUI (Graphical User Interface):

```java
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.JPanel;

public class GUIExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Contoh GUI");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();
        JButton button = new JButton("Klik saya!");

        panel.add(button);
        frame.add(panel);

        frame.pack();
        frame.setVisible(true);
    }
}
```

- javax.swing.JFrame: Merepresentasikan jendela aplikasi GUI.
- javax.swing.JButton: Membuat tombol pada GUI.
- javax.swing.JPanel: Mengelompokkan komponen GUI.

6. Java API untuk Manajemen Basis Data:

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class DatabaseExample {
    public static void main(String[] args) {
        try {
            Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/mydatabase", "username", "password");

            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery("SELECT * FROM employees");

            while (resultSet.next()) {
                String name = resultSet.getString("name");
                int age = resultSet.getInt("age");
                System.out.println("Name: " + name + ", Age: " + age);
            }

            resultSet.close();
            statement.close();
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

- java.sql.Connection: Membuka koneksi ke basis data.
- java.sql.Statement: Mengeksekusi pernyataan SQL pada basis data.
- java.sql.ResultSet: Merepresentasikan hasil dari kueri SQL.

7. Java API untuk Pengolahan Gambar:

```java
import java.awt.image.BufferedImage;
import javax.imageio.ImageIO;
import java.io.File;
import java.io.IOException;

public class ImageProcessingExample {
    public static void main(String[] args) {
        try {
            BufferedImage image = ImageIO.read(new File("input.jpg"));

            int width = image.getWidth();
            int height = image.getHeight();

            System.out.println("Width: " + width + ", Height: " + height);

            ImageIO.write(image, "png", new File("output.png"));
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

- java.awt.image.BufferedImage: Merepresentasikan gambar di dalam memori.
- javax.imageio.ImageIO: Membaca dan menulis file gambar.

Contoh di atas hanya mencakup beberapa contoh Java API yang populer. Ada banyak lagi API yang tersedia dalam ekosistem Java untuk berbagai kebutuhan pengembangan perangkat lunak. Contoh lain ada di **link ini**.

MATERI PRATIKUM

1. Implementasi Java API dalam Program ArrayList

Dalam contoh ini, kita mengimpor library java.util.ArrayList. Selanjutnya, kita membuat objek list menggunakan kelas ArrayList, dan menggunakan API yang disediakan oleh kelas tersebut untuk menambahkan dan menghapus elemen dalam list. API yang digunakan dalam contoh ini adalah method-method yang disediakan oleh kelas ArrayList, seperti add dan remove.

```java
import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {
        // Membuat objek list dengan library ArrayList
        ArrayList<Integer> numbers = new ArrayList<Integer>();

        // Menambahkan elemen ke dalam list
        numbers.add(1);
        numbers.add(2);
        numbers.add(3);

        // Menampilkan elemen-elemen dalam list
        System.out.println("List sebelum dihapus: " + numbers);

        // Menghapus elemen dari dalam list
        numbers.remove(1);

        // Menampilkan elemen-elemen dalam list setelah dihapus
        System.out.println("List setelah dihapus: " + numbers);
    }
}
```

2. Implementasi Java API dalam Program File Reader

   Dalam contoh ini, kita mengimpor library java.io. API yang digunakan dalam contoh ini adalah method-method yang disediakan oleh kelas File dan BufferedReader, seperti File(String pathname), BufferedReader(Reader in), dan readLine().

```java
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;

public class Main {
    public static void main(String[] args) {
        // Membuka file
        File file = new File("data.txt");

        try {
            // Membaca isi file menggunakan BufferedReader
            BufferedReader reader = new BufferedReader(new FileReader(file));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("Gagal membaca file: " + e.getMessage());
        }
    }
}
```

3. Implementasi Java API dalam Program Local Date dan Local Time

Dalam contoh ini, kita mengimpor library java.time. API yang digunakan dalam contoh ini adalah method-method yang disediakan oleh kelas-kelas LocalDate, LocalTime, dan DateTimeFormatter, seperti now(), format(), dan ofPattern().

```java
import java.time.LocalDate;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class Main {
    public static void main(String[] args) {
        // Membuat objek tanggal dan waktu saat ini
        LocalDate tanggal = LocalDate.now();
        LocalTime waktu = LocalTime.now();

        // Memformat tanggal dan waktu dalam bentuk string
        String tanggalFormatted = tanggal.format(DateTimeFormatter.ofPattern("dd/MM/yyyy"));
        String waktuFormatted = waktu.format(DateTimeFormatter.ofPattern("HH:mm:ss"));

        // Menampilkan hasil
        System.out.println("Tanggal hari ini: " + tanggalFormatted);
        System.out.println("Waktu saat ini: " + waktuFormatted);
    }
}
```

REFERENSI

https://josikie.com/apa-itu-java-api/

https://www.simplilearn.com/tutorials/java-tutorial/java-api#what_are_java_apis

https://www.educba.com/what-is-api-in-java/

https://www.geeksforgeeks.org/what-is-an-api/

https://www.javatpoint.com/what-is-an-api

https://www.mas-software.com/blog/api-adalah-jenis-arsitektur

https://www.javatpoint.com/what-is-an-api

https://www.javatpoint.com/api-application-programming-interface

https://www.freecodecamp.org/news/what-is-an-api-and-how-to-test-it/

https://docs.oracle.com/en/java/javase/11/docs/api/index.html

https://docs.oracle.com/javase/8/docs/api/

https://rapidapi.com/blog/how-to-use-an-api-with-java/

LATIHAN PRATIKUM

LATIHAN 1

Buatlah program kalkulator yang berisikan operasi perhitungan akar kuadrat, pangkat, logaritma natural (basis e), dan factorial. Implementasikan materi API dalam pembuatan program ini ! **(Clue: Math.sqrt, Math.pow, Math.log)**. Contoh output bisa dilihat di link ini.

```
=== Calculator ===
1. Akar Kuadrat
2. Pangkat
3. Logaritma
4. Faktorial
5. Keluar
Pilih operasi (1/2/3/4/5):
```

LATIHAN 2

Buatlah program text analyzer yang berisikan analisis teks berupa hitung jumlah karakter, hitung jumlah kata, dan cari kata dalam teks. Implementasikan materi API dalam pembuatan program ini ! **(Clue: isEmpty( ), length( ), split( ), equalsIgnoreCase( ))**. Contoh output bisa dilihat di link ini.

```
=== MENU TEKS ANALYZER ===
1. Masukkan Teks
2. Hitung Jumlah Karakter
3. Hitung Jumlah Kata
4. Cari Kata dalam Teks
5. Keluar
Pilihan Anda:
```

TUGAS

TUGAS 1

Anda adalah seorang peneliti dalam bidang psikologi yang tertarik untuk melakukan eksperimen mengenai persepsi warna pada manusia. Anda ingin mengumpulkan data mengenai bagaimana manusia merespons kombinasi warna tertentu dan mengidentifikasi preferensi warna mereka. Data yang diperlukan adalah data acak bertipe data integer dan string. Oleh karena itu, buatlah program Random Data Generator dengan mengimplementasikan materi API untuk menghasilkan data acak integer dan string. **(Clue: java.util.Random, StringBuilder, append( ), charAt( ), toString())**. Contoh output bisa dilihat di link ini.

```
Menu:
1. Random Angka
2. Random String
3. Keluar
Pilih opsi (1/2/3):
```

Petunjuk:

- Random Angka: Menghasilkan angka acak di antara batas bawah dan batas atas yang dimasukkan oleh pengguna.
- Random String: Menghasilkan sebuah string acak dengan panjang yang dimasukkan oleh pengguna, terdiri dari karakter huruf (kapital dan kecil) dan angka.

TUGAS 2

Perhatikan data berikut !

Data : 38, -15, 72, 0, -42, 19, -63, 50, 27, -84, 61, 5

Buatlah program dengan mengimplementasikan materi API untuk **mengurutkan** data di atas, kemudian **cari index elemen** menggunakan inputan dari user ! Hitunglah elemen rasio yang memiliki nilai positif, negatif, dan nol. Kemudian **cetak nilai rasio** dari setiap pecahan pada baris baru. **(Clue: import java.util.Arrays, sort( ), binarySearch( )).**

```
Data yang sudah diurutkan:
-84 -63 -42 -15 0 5 19 27 38 50 61 72
Masukkan nilai yang ingin dicari indexnya: 5
Nilai ditemukan di index: 5
Jumlah elemen dengan nilai positif: 7
Jumlah elemen dengan nilai negatif: 4
Jumlah elemen dengan nilai nol: 1
```

TUGAS 3

Jelaskan kepada asisten perbedaan antara API dan Library dengan bahasa kamu sendiri !

KRITERIA & DETAIL PENILAIAN

Kriteria Penilaian	Nilai	
Latihan 1		15
Telah mengimplementasikan API untuk perhitungan akar kuadrat (bisa dijalankan dan menghasilkan hasil yang benar)	3	
Telah mengimplementasikan API untuk perhitungan pangkat (bisa dijalankan dan menghasilkan hasil yang benar)	3	
Telah mengimplementasikan API untuk perhitungan logaritma natural (bisa dijalankan dan menghasilkan hasil yang benar)	3	
Perhitungan faktorial bisa dijalankan dan menghasilkan hasil yang benar	4	
Program tidak error	2	
Latihan 2		15
Telah mengimplementasikan API dalam program	3	
Program bisa digunakan untuk menghitung jumlah karakter dari teks yang diinputkan user	3	
Program bisa digunakan untuk menghitung jumlah kata dari teks yang diinputkan user	3	
Program bisa digunakan untuk mencari kata dalam teks yang diinputkan user	4	
Program tidak error	2	
Tugas 1		20
Telah mengimplementasikan API dalam program	6	
Program bisa digunakan untuk mendapatkan data berupa angka random di antara batas bawah dan batas atas yang dimasukkan oleh pengguna	6	
Program bisa digunakan untuk mendapatkan data berupa string random dengan panjang yang dimasukkan oleh pengguna	6	
Program tidak error	2	
Tugas 2		15
Telah mengimplementasikan API untuk mengurutkan data	4	
Telah mengimplementasikan API untuk melakukan searching index data	4	
Program bisa menghitung elemen rasio data yang memiliki nilai positif, negatif, dan nol	4	
Program Tidak Error	3	

Tugas 3		10
Pemahaman		25
Total		**100**