LAPORAN PRAKTIKUM ALGORITMA DAN PEMOGRAMAN

“STRING PADA JAVA”

DISUSUN OLEH:

MUHAMMAD FATHAN EDLIN

2511537001

DOSEN PENGAMPU:

Dr. WAHYUDI, S.T, M.T

ASISTEN PRAKTIKUM:

JOVANTRI IMMANUEL GELO

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji syukur kehadirat Allah SWT atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tugas dengan judul "String pada Java" dengan baik dan tepat waktu. Penulisan materi ini bertujuan untuk memberikan pemahaman yang lebih mendalam mengenai konsep dasar, fungsi, serta penerapan *string* dalam bahasa pemrograman Java.

Dalam pemrograman Java, *string* memiliki peranan yang sangat penting karena hampir semua program membutuhkan pengolahan data berupa teks. Materi ini diharapkan dapat membantu pembaca memahami cara kerja *string*, mulai dari deklarasi, inisialisasi, hingga penggunaan metode-metode bawaan yang sering digunakan dalam pengolahan teks.

Penyusunan materi ini tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis ingin menyampaikan terima kasih kepada guru pembimbing yang telah memberikan arahan, serta rekan-rekan yang turut membantu dalam penyusunan tugas ini. Diharapkan penjelasan yang terdapat di dalam tulisan ini dapat memberikan manfaat, baik bagi penulis sendiri maupun bagi pembaca yang ingin memperdalam pengetahuan tentang *string* dalam bahasa Java.

Penulis menyadari bahwa masih banyak kekurangan dalam penyusunan materi ini, baik dari segi penjelasan maupun penyajian. Oleh sebab itu, kritik dan saran yang membangun sangat diharapkan agar tulisan ini dapat menjadi lebih baik di kemudian hari.

Akhir kata, semoga tugas ini dapat memberikan tambahan wawasan dan menjadi referensi yang berguna bagi siapa pun yang ingin mempelajari pemrograman Java, khususnya mengenai penggunaan *string*.

Padang, 13 November 2025

Muhammad Fathan Edlin

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi yang pesat menjadikan pemrograman komputer sebagai salah satu keterampilan penting di era digital. Salah satu bahasa pemrograman yang banyak digunakan hingga saat ini adalah Java. Java dikenal sebagai bahasa pemrograman yang bersifat object-oriented, portable, dan banyak digunakan untuk membangun berbagai jenis aplikasi, mulai dari aplikasi desktop, web, hingga mobile.

Dalam proses pembuatan program, salah satu aspek yang sering dijumpai adalah pengolahan data berupa teks atau kumpulan karakter. Hampir setiap program membutuhkan manipulasi teks, seperti menampilkan pesan kepada pengguna, membaca input, memproses kata, atau menyimpan data berbentuk tulisan. Untuk menangani hal tersebut, Java menyediakan tipe data khusus bernama String.

String merupakan kumpulan karakter yang disusun secara berurutan dan digunakan untuk menyimpan teks. Misalnya, kata “Java”, “Pemrograman”, atau “Hello World” semuanya termasuk string. Dalam Java, string bukanlah tipe data primitif seperti int atau char, tetapi merupakan sebuah objek dari kelas String. Kelas ini memiliki banyak metode bawaan (built-in methods) yang sangat membantu dalam manipulasi data teks, seperti menggabungkan kata, memotong teks, mengganti karakter, mencari kata tertentu, serta mengubah huruf menjadi besar atau kecil.

Pemahaman terhadap string sangat penting bagi seorang programmer, karena hampir seluruh proses interaksi antara pengguna dan program melibatkan teks. Dengan memahami cara kerja string, programmer dapat menulis kode yang lebih efisien, mudah dibaca, serta mampu mengelola data teks secara lebih efektif.

Selain itu, Java juga menyediakan dua kelas lain, yaitu StringBuilder dan StringBuffer, yang digunakan untuk memanipulasi teks dengan cara yang lebih efisien terutama saat melakukan perubahan berulang pada string. Pemahaman terhadap ketiga komponen ini sangat penting agar programmer dapat memilih pendekatan yang paling sesuai dengan kebutuhan program yang dibuat.

Dengan demikian, pembahasan mengenai string pada Java menjadi dasar penting sebelum mempelajari konsep-konsep pemrograman yang lebih

kompleks. Melalui pemahaman yang baik tentang string, diharapkan **pembaca dapat memahami cara kerja teks di Java serta mampu mengimplementasikannya dalam program nyata.**

1.2 Tujuan

Tujuan dari penyusunan materi String pada Java ini adalah:

1. Menjelaskan pengeertian string serta peran pentiingnya dalam Bahasa java
2. Menunjukkan cara pembuatan dan penggunaan string dalam program
3. Menguraikan berbagai metode yang terdapat pada kelas string untuk manipulasi text
4. Menjelaskan perbedaan antara string, stringbulider, dan stringbuffer serta contoh penggunaannya
5. Memberikan contoh sederhana penerapan string dalam program java agar mudah dipahami oleh pembaca

1.3 Manfaat

Manfaat yang diharapkan dari penyusunan materi ini antara lain:

1. Bagi siswa atau pelajar, dapat menambah pengetahuan dan pemahaman tentang konsep dasar pengolahan teks dalam Java.
2. Bagi pengajar, dapat dijadikan sebagai bahan ajar atau referensi dalam pembelajaran pemrograman dasar Java.
3. Bagi pembaca umum, dapat memberikan gambaran jelas mengenai bagaimana Java mengelola teks secara efisien menggunakan string.
4. Menjadi dasar yang kuat untuk mempelajari konsep pemrograman lebih lanjut seperti data structure, file handling, atau regular expression yang juga melibatkan pengolahan teks.

BAB II
PEMBAHASAN

2.1 Pengertian String dalam java

Dalam bahasa pemrograman Java, String merupakan salah satu tipe data yang paling sering digunakan untuk menyimpan dan mengelola teks. String digunakan untuk menampung kumpulan karakter seperti huruf, angka, simbol, atau spasi yang dikelilingi oleh tanda kutip ganda (").

Contohnya:

```
String nama = "Fathan";
```

Pada contoh di atas, variabel nama menyimpan nilai berupa teks “Fathan”. Berbeda dengan bahasa pemrograman lain, di Java String bukan tipe data primitif, melainkan sebuah objek dari kelas String yang terdapat di paket java.lang. Hal ini berarti setiap String memiliki berbagai metode (method) bawaan yang bisa digunakan untuk memanipulasi atau mengolah data teks.

2.2 Cara membuat String

Ada dua cara umum untuk membuat String di Java, yaitu menggunakan literal String dan keyword new.

1. Menggunakan Literal String
   Cara ini paling sederhana, cukup menuliskan teks di dalam tanda kutip ganda.
   Contoh:
   ```
   String kata = "Belajar Java";
   ```
2. Menggunakan Keyword new
   Cara ini digunakan untuk membuat objek String baru secara eksplisit.
   Contoh:
   ```
   String kalimat = new String("Belajar Java");
   ```

2.3 Metode–Metode pada String

elas String di Java menyediakan banyak metode untuk memudahkan pengolahan teks. Berikut beberapa metode yang sering digunakan:

1. length() – Menghitung jumlah karakter dalam String.
   ```
   String teks = "Java";
   System.out.println(teks.length());
   ```
2. toUpperCase() dan toLowerCase() – Mengubah huruf menjadi kapital atau kecil.
   ```
   String kata = "Belajar";
   System.out.println(kata.toUpperCase());
   System.out.println(kata.toLowerCase());
   ```
3. charAt(int index) – Mengambil karakter tertentu berdasarkan posisi indeks.
   ```
   String teks = "Program";
   ```

```
System.out.println(teks.charAt(2));
```

4. substring(int beginIndex, int endIndex) – Mengambil sebagian teks dari String.
```
String kalimat = "Pemrograman Java";
System.out.println(kalimat.substring(0, 11));
```
5. equals() dan equalsIgnoreCase() – Membandingkan dua String.
```
String s1 = "Java";
String s2 = "java";
System.out.println(s1.equals(s2));
System.out.println(s1.equalsIgnoreCase(s2));
```
6. replace() – Mengganti karakter atau kata dalam String.
```
String kalimat = "Saya belajar Python";
System.out.println(kalimat.replace("Python", "Java"));
```
7. contains() – Mengecek apakah suatu teks terdapat di dalam String.
```
String kalimat = "Pemrograman Java";
System.out.println(kalimat.contains("Java"));
```

2.4 Penggabungan String (Concatenation)

Java menyediakan beberapa cara untuk menggabungkan dua atau lebih String menjadi satu. Cara paling umum adalah menggunakan tanda “+” atau metode concat().

Contoh 1 – Menggunakan operator “+”

```
String nama = "Fathan";
String pesan = "Halo " + nama + ", selamat belajar Java!";
System.out.println(pesan);
```

Contoh 2 – Menggunakan metode concat()

```
String teks1 = "Belajar ";
String teks2 = "Java";
System.out.println(teks1.concat(teks2));
```

Selain itu, pada aplikasi yang memerlukan penggabungan String dalam jumlah besar, disarankan menggunakan kelas StringBuilder atau StringBuffer, karena lebih efisien dalam penggunaan memori.

2.5 Perbedaan String, StringBuilder, dan StringBuffer

Kelas	Bersifat	Kelebihan	Kekurangan
String	Immutable (tidak bisa diubah)	Aman digunakan, mudah dipahami	Tidak efisien jika sering diubah
StringBuilder	Mutable (bisa diubah)	Proses lebih cepat	Tidak aman untuk thread
StringBuffer	Mutable dan thread-safe	Aman untuk multi-thread	Sedikit lebih lambat dari StringBuilder

Contoh penggunaan StringBuilder:

```
StringBuilder sb = new StringBuilder("Belajar ");

sb.append("Java");

System.out.println(sb);
```

2.6 Contoh Program Lengkap

Berikut contoh program sederhana yang menunjukkan penggunaan berbagai metode String:

```
public class StringExample {
    public static void main(String[] args) {
        String nama = "Fathan";
        String pesan = "Selamat datang di dunia Java, " + nama + "!";

        System.out.println(pesan);
        System.out.println("Panjang teks: " + pesan.length());
        System.out.println("Huruf besar: " + pesan.toUpperCase());
        System.out.println("Mengandung kata 'Java': " +
pesan.contains("Java"));
        System.out.println("Ganti 'Java' dengan 'Pemrograman': " +
pesan.replace("Java", "Pemrograman"));
    }
}
```

Hasil Output:

```
Selamat datang di dunia Java, Fathan!
Panjang teks: 36
Huruf besar: SELAMAT DATANG DI DUNIA JAVA, FATHAN!
Mengandung kata 'Java': true
Ganti 'Java' dengan 'Pemrograman': Selamat datang di dunia
Pemrograman, Fathan!
```

2.7 Kesimpulan Sementara

String merupakan bagian penting dalam Java karena hampir semua program membutuhkan pengolahan teks. Dengan memahami cara kerja, metode, serta perbedaan antara String, StringBuilder, dan StringBuffer, programmer dapat menulis kode yang lebih efisien, mudah dibaca, dan hemat memori.

BAB III

KESIMPULAN DAN SARAN

3.1 Kesimpulan

Dari pembahasan mengenai *String pada Java*, dapat disimpulkan bahwa *String* merupakan salah satu tipe data yang sangat penting dalam pemrograman Java. *String* digunakan untuk menyimpan dan memanipulasi data berupa teks. Dalam Java, *String* bukan termasuk tipe data primitif, melainkan merupakan sebuah **kelas** dari paket java.lang. Oleh karena itu, setiap objek *String* memiliki berbagai metode yang memudahkan programmer untuk mengelola data teks, seperti penggabungan, pemisahan, penggantian, pencarian, dan perbandingan.

Java memperlakukan *String* sebagai objek yang bersifat **immutable**, artinya nilai dari suatu *String* tidak dapat diubah setelah dibuat. Jika ada perubahan, maka Java sebenarnya membuat objek *String* baru. Untuk manipulasi teks yang membutuhkan perubahan berulang, Java menyediakan kelas **StringBuilder** dan **StringBuffer**, yang lebih efisien dalam penggunaan memori dan kecepatan pemrosesan.

Selain itu, penggunaan *String* juga sering dijumpai dalam berbagai aspek pemrograman seperti input/output, komunikasi antar objek, serta tampilan antarmuka pengguna. Kemampuan untuk memahami dan mengelola *String* dengan baik akan sangat membantu programmer dalam menulis kode yang lebih efisien, mudah dibaca, dan mudah dikelola.

Secara keseluruhan, penguasaan terhadap konsep dan penggunaan *String* dalam Java merupakan dasar penting yang harus dipahami oleh setiap programmer sebelum melangkah ke tahap pemrograman yang lebih kompleks.

3.2 Saran

Dalam mempelajari *String pada Java*, disarankan agar mahasiswa atau pelajar:

1. **Sering berlatih membuat program sederhana** yang melibatkan pengolahan teks, seperti menggabungkan, membalik, atau memeriksa isi *String*.
2. **Mempelajari metode-metode penting dalam kelas String**, seperti length(), substring(), equals(), compareTo(), toUpperCase(), dan toLowerCase().
3. **Memahami perbedaan antara String, StringBuilder, dan StringBuffer** agar dapat memilih tipe yang paling sesuai dengan kebutuhan program.
4. **Menerapkan konsep *immutable object*** dengan baik agar terhindar dari penggunaan memori yang tidak efisien.

5. **Membaca dokumentasi resmi Java** secara rutin untuk mengetahui metode-metode baru dan cara penggunaannya.

Dengan memahami dan mempraktikkan konsep *String* secara mendalam, diharapkan seseorang dapat menulis program Java yang tidak hanya benar secara sintaks, tetapi juga efisien, terstruktur, dan mudah dikembangkan di masa depan.

DAFTAR PUSTAKA

[1] Cakwala.com, “Pengertian String Dalam Java,”.[Daring]. Tersedia pada: https://www.cakrawala.ac.id/berita/apa-itu-string [Diakses: 13-Nov-2025]

[2] Codingstudio.id, “Cara membuat String,”.[Daring]. Tersedia pada: https://codingstudio.id/blog/string-adalah/ [Diakses: 13-Nov-2025]

[3] Codingstudio.id, “Metode-metode Pada String,”.[Daring]. Tersedia pada: https://codingstudio.id/blog/string-adalah/ [Diakses: 13-Nov-2025]

[4] Codegym.cc, “Penggabungan String,”.[Daring]. Tersedia pada: https://codegym.cc/id/groups/posts/id.868.penggabungan-string-di-java [Diakses: 13-Nov-2025]

[5] Codegym.cc, “Perbedaan String, Stringbuilder, dan StringBuffer,”.[Daring]. Tersedia pada: https://codegym.cc/id/groups/posts/id.868.penggabungan-string-di-java [Diakses: 13-Nov-2025]