LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

BAB	:	COLLECTION STRING
NAMA	:	NAMA PRAKTIKAN
NIM	:	NIM PRAKTIKAN
ASISTEN	:	Tengku Muhammad Rafi Rahardiansyah Muhammad Bin Djafar Almasyhur
TGL PRAKTIKUM	:	31 Mei 2023

BAB 12
STRING

Tujuan

1. Memberikan pemahaman kepada mahasiswa tentang String, StringBuffer, StringBuilder, StringJoiner, StringTokenizer
2. Menerapkan metode-metode yang ada di String, StringBuffer, StringBuilder, StringJoiner, StringTokenizer

Ringkasan Materi

1. String

Java mendefinisikan class String dalam package java.lang.String, sehingga tidak perlu melakukan impor secara eksplisit. String merupakan tipe data untuk menyimpan nilai berupa teks. String merupakan class (tipe data reference).Ketika variabel bertipe String dibuat, secara otomatis compiler Java akan membuatkan sebuah object String. Kategori Konstruktor String yaitu:

- Membuat String Kosong
- Membuat String dari string lain
- String dari Byte Array
- String dari Array Karakter
- String dari Poin Kode
- String dari StringBuffer dan StringBuilder

Sintaksis:

String namaVariabel;
String namaVar = new String();

2. StringBuffer

Kelas StringBuffer dapat dipergunakan untuk membuat objek String yang dapat diubah datanya. Berbeda dengan kelas String, yang menyerupai konstanta String pada memori. Perubahan nilai String di kelas StringBuffer selalu tidak menciptakan objek String baru namun dilakukan dengan mengubah secara langsung pada lokasi memori saat diciptakan.

Kontruktor untuk StringBuffer yaitu:

- StringBuffer(): membuat buffer string kosong dengan kapasitas 16 karakter.
- StringBuffer(int capacity): membuat buffer string kosong dengan kapasitas karakter yang ditentukan. Hal ini berguna ketika Anda mengetahui kapasitas yang dibutuhkan oleh string buffer untuk menghemat waktu dalam meningkatkan kapasitas.
- StringBuffer(String str): membuat buffer string yang memiliki karakter yang sama dengan argumen string. Kapasitas awal buffer string adalah "panjang str + 16".
- StringBuffer(CharSequence seq): membuat buffer string dengan karakter yang sama seperti pada file CharSequence. Kapasitas awal buffer string adalah "panjang seq + 16".

Sintaksis:

```
StringBuffer namaVariabel = new StringBuffer();
```

3. StringBuilder

StringBuilder di Java adalah kelas yang digunakan untuk membuat karakter yang dapat diubah, atau dengan kata lain, urutan karakter yang dapat dimodifikasi. Seperti StringBuffer, kelas StringBuilder adalah alternatif untuk Java String Class, karena kelas String menyediakan urutan karakter yang tidak dapat diubah. Namun, ada satu perbedaan signifikan antara StringBuffer dan StringBuilder, yaitu StringBuilder tidak disinkronkan. Ini berarti bahwa StringBuilder di Java adalah pilihan yang lebih cocok saat bekerja dengan single thread, karena akan lebih cepat daripada StringBuffer.

Kelas StringBuilder memiliki 4 konstruktor, yaitu:

- StringBuilder(): membuat pembuat string kosong dengan kapasitas 16 karakter.
- StringBuilder(int capacity): membuat pembuat string kosong dengan kapasitas karakter yang ditentukan. Ini berguna ketika Anda mengetahui kapasitas yang dibutuhkan oleh pembuat string untuk menghemat waktu dalam meningkatkan kapasitas.
- StringBuilder(String str): membuat pembuat string baru yang memiliki karakter yang sama dengan argumen string. Kapasitas awal pembuat string adalah "panjang str + 16".
- StringBuilder(CharSequence seq): membuat pembuat string baru dengan karakter yang sama seperti di CharSequence. Kapasitas awal pembuat string adalah "panjang seq + 16".

Sintaksis:

```
StringBuilder namaVariabel = new StringBuilder();
```

4. StringJoiner

java menambahkan new final class StringJoiner dalam paket java.util. StringJoiner digunakan untuk membangun urutan karakter yang dipisahkan oleh pembatas (delimiter). StringJoiner juga dapat membuat string dengan menambahkan pembatas seperti koma(), tanda hubung(-) dll. Selain itu juga dapat meneruskan awalan dan akhiran ke urutan karakter.

Konstruktor Kelas StringJoiner,yaitu:

- StringJoiner(CharSequence delimiter): membuat instance dengan delimiter yang diberikan dan tanpa urutan karakter sufiks awalan. Jika pembatasnya nol, NullPointerException dilempar.
- StringJoiner (pembatas CharSequence, awalan CharSequence, akhiran CharSequence): membuat instance dengan urutan karakter pembatas, awalan, dan akhiran yang diberikan. Ini akan membuang NullPointerException jika salah satu argumennya nol.

Sintaksis:

```
StringJoiner namaVariabel = new StringJoiner(<delimiter>);
```

5. StringTokenizer

Kelas java.util.StringTokenizer digunakan untuk memecah String menjadi token. StringTokenizer tidak menyediakan fasilitas untuk membedakan angka, string yang dikutip, pengidentifikasi dll seperti kelas StreamTokenizer. Di kelas StringTokenizer, pembatas dapat diberikan pada saat pembuatan atau satu per satu ke token.

Kelas StringTokenizer memiliki tiga konstruktor,yaitu:

- StringTokenizer(String str): Ini membuat instance tokenizer string dengan string yang diberikan dan karakter pembatas default. Karakter pembatas default adalah karakter spasi ( ), karakter tab (\t), karakter baris baru (\n), karakter carriage-return (\r), dan karakter form-feed (\f). Karakter pembatas bukan bagian dari string token.
- StringTokenizer(String str, String delimiter): Membuat instance StringTokenizer dengan string dan pembatas yang ditentukan. String pembatas bukan bagian dari token yang dihasilkan.
- StringTokenizer(String str, String delimiter, boolean returnDelims): Mirip dengan konstruktor di atas kecuali bahwa string pembatas juga akan menjadi bagian dari string token.

Sintaksis:
StringTokenizer namaVariabel=new StringTokenizer(<string>)

Pelaksaan Percobaan

1. String

JavaString.java

```
1.  public class JavaString {
2.      public static void main(String[] args) {
3.          String s1 = "Hello";
4.          String s2 = "Hello";
5.          String s3 = new String("Hi");
6.          String s4 = "Hi";
7.          System.out.println((s1 == s2));
8.          System.out.println((s3 == s4));
9.          s3 = s3.intern();
10.         System.out.println((s3 == s4));
11.         System.out.println(s1.concat(s4));
12.         System.out.println(s1.repeat(3));
13.         System.out.println(s1.replace('e', 'o'));
14.         System.out.println(s1.substring(3));
15.         System.out.println(s1.toLowerCase());
16.         System.out.println(s1.toUpperCase());
17.         System.out.println(s1.charAt(1));
18.         System.out.println(s1.indexOf('l'));
19.         System.out.println(s1.equals(s2));
20.         System.out.println(s1.compareTo(s2));
21.         System.out.println(s1.startsWith("He"));
22.         System.out.println(s1.endsWith("lo"));
23.     }
24. }
```

2. StringBuffer

JavaStringBuffer.java

```
1.  public class JavaStringBuffer {
2.      public static void main(String[] args) {
3.          StringBuffer sb = new StringBuffer();
4.          System.out.println(sb.capacity());
5.          System.out.println(sb.toString());
6.          StringBuffer sb1 = new StringBuffer(1000);
7.          System.out.println(sb1.capacity());
8.          System.out.println(sb1.toString());
9.          StringBuffer sb2 = new
    StringBuffer("Hello");
10.         System.out.println(sb2.capacity());
11.         System.out.println(sb2.toString());
12.         CharSequence seq = new
    StringBuilder("Hello");
13.         StringBuffer sb3 = new StringBuffer(seq);
14.         System.out.println(sb3.capacity());
15.         System.out.println(sb3.toString());
16.         sb3.append(" Java");
17.         System.out.println(sb3.toString());
18.         sb3.insert(10, " Programming");
19.         System.out.println(sb3.toString());
20.         System.out.println(sb3.compareTo(sb2));
21.         System.out.println(sb3.equals(sb2));
22.         System.out.println(sb3.charAt(2));
23.         System.out.println(sb3.substring(3));
24.         System.out.println(sb3.length());
25.         System.out.println(sb3.reverse());
26.     }
27. }
```

3. StringBuilder

JavaStringBuilder.java

```
1.  public class JavaStringBuilder {
2.      public static void main(String[] args) {
3.          StringBuilder sb = new StringBuilder();
4.          System.out.println(sb.capacity());
5.          System.out.println(sb.toString());
6.          StringBuilder sb1 = new
    StringBuilder(1000);
7.          System.out.println(sb1.capacity());
8.          System.out.println(sb1.toString());
9.          StringBuilder sb2 = new
    StringBuilder("Java");
10.         System.out.println(sb2.capacity());
11.         System.out.println(sb2.toString());
12.         CharSequence seq = new
    StringBuilder("String");
13.         StringBuilder sb3 = new
    StringBuilder(seq);
14.         System.out.println(sb3.capacity());
15.         System.out.println(sb3.toString());
16.         sb3.append(" Java");
17.         System.out.println(sb3.toString());
18.         sb3.insert(11, " Builder");
19.         System.out.println(sb3.toString());
20.         System.out.println(sb3.compareTo(sb2));
```

```
        System.out.println(sb3.equals(sb2));
        System.out.println(sb3.charAt(2));
        System.out.println(sb3.substring(3));
        System.out.println(sb3.length());
        System.out.println(sb3.reverse());
    }
}
```

4. StringJoiner

JavaStringJoiner.java

```
import java.util.StringJoiner;
public class JavaStringJoiner {
    public static void main(String[] args) {
        StringJoiner sj = new StringJoiner(",", "{", "}");
        sj.add("USA");
        sj.add("India");
        sj.add("UK");
        System.out.println(sj);
        StringJoiner sj1 = new StringJoiner("|", "#", "#");
        sj1.add("France");
        sj1.add("Germany");
        System.out.println(sj1);
        sj.merge(sj1);
        System.out.println(sj);
    }
}
```

5. StringTokenizer

JavaStringTokenizer.java

```
1.  import java.util.Iterator;
2.  import java.util.StringTokenizer;
3.  public class JavaStringTokenizer {
4.      public static void main(String[] args) {
5.          String csv = "Java,Python,Android";
6.          StringTokenizer st = new StringTokenizer(csv, ",");
7.          printTokens(st);
8.          String line = "Welcome to
    JavaString.net\nLearn\tJava\tProgramming";
9.          StringTokenizer st1 = new StringTokenizer(line);
10.         printTokens(st1);
11.         StringTokenizer st2 = new StringTokenizer("Hi Hello
    Yes");
12.         System.out.println(st2.countTokens());
13.         st2.nextToken();
14.         System.out.println(st2.countTokens());
15.         StringTokenizer st3 = new
    StringTokenizer("Hi|Hello|Yes", "|", true);
16.         System.out.println(st3.countTokens());
17.         printTokens(st3);
18.         StringTokenizer st4 = new StringTokenizer("Hello Java
    World");
19.         Iterator<Object> iterator = st4.asIterator();
20.         while (iterator.hasNext()) {
21.             System.out.println(iterator.next());
22.         }
23.     }
24.     private static void printTokens(StringTokenizer st) {
25.         while (st.hasMoreTokens()) {
26.             System.out.println(st.nextToken());
27.         }
28.    }
29. }
```

Data dan Analisis Hasil Percobaan

1. String
   a. Jalankan program JavaString.java kemudian tuliskan outputnya
   b. Jelaskan hasil output line 7,8 dan 10, Mengapa bernilai true atau false
   c. Jelaskan beda dari method equal dan compareTo pada String

Source code

```
1  Tulis source code di sini pake courier new 12
```

Output

Penjelasan

2. StringBuffer
   a. Jalankan program JavaStringBuffer.java kemudian tuliskan outputnya
   b. Mengapa baris 5 dan 8 hanya ditampilkan output kosong?
   c. Mengapa pada baris 10 kalau dilihat kapasitasnya akan mengeluarkan angka 21padahal string “Hello” hanya terdiri dari 5 karakter?
   d. Apakah bedanya method append dan insert pada klas StringBuffer?

Source code

```
1  Tulis source code di sini pake courier new 12
```

Output

Penjelasan

3. StringBuilder

   a. Jalankan program JavaStringBuilder.java kemudian tuliskan outputnya
   b. Apabila pada baris 17 ditampilkan juga capacity dari sb3, berapakah output yang dihasilkan? Jelaskan alasannya
   c. Apabila pada baris 19 ditampilkan juga capacity dari sb3, berapakah output yang dihasilkan? Jelaskan alasannya

Source code

```
1  Tulis source code di sini pake courier new 12
```

Output

Penjelasan

4. StringJoiner
   a. Jalankan program JavaStringJoiner.java kemudian tuliskan outputnya
   b. Apabila pada baris 13 diganti menjadi `sj1.merge(sj);`
   c. Apakah output yang dihasilkan?jelaskan alasannya

Source code

```
1   Tulis source code di sini pake courier new 12
```

Output

Penjelasan

5. StringTokenizer
   a. Jalankan program JavaStringTokenizer.java kemudian tuliskan outputnya
   b. Mengapa pada baris ke 9 bisa dipisah token berdasarkan lebih dari satu delimiter?

Source code

```
1   Tulis source code di sini pake courier new 12
```

Output

Penjelasan

Tugas Praktikum

1. Buatlah program yang dapat memecah String menjadi suatu token dan menghitung frekuensi masing-masing token untuk paragraf berikut ini!

   Saya belajar bahasa Java. Bahasa Java mempunyai kelas berupa String. Belajar String Java itu mudah. String Java juga sering diimplementasikan.

Token	**Frekuensi**
Saya	1
belajar	2
bahasa	2
Java	4
….	….
dst	dst