

LAPORAN PRAKTIKUM
ALGORITMA PEMOGRAMAN
STRING PADA PEMOGRAMAN JAVA

Disusun Oleh :

Aliifah Felda Mufarrihati Salwaa

2511531011

Dosen Pengampu :

Dr. Wahyudi, S.T., M.T.

Asisten Praktikum :

Aufan Taufiqurrahman



DEPARTEMEN INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS
TAHUN 2025

KATA PENGANTAR

Laporan praktikum ini disusun sebagai salah satu bentuk pertanggungjawaban kegiatan praktikum algoritma pemograman yang membahas tentang string pada pemograman Java. Melalui laporan ini penulis dapat memahami materi praktikum secara mendalam dan dapat lebih teliti, teratur, serta memiliki kemampuan yang baik dalam penulisan kode pemograman sesuai kaidah akademik. Sehingga laporan ini dapat menjadi sarana belajar, dokumentasi kegiatan, dan referensi praktikum berikutnya.

Penulis menyadari bahwa laporan ini masih memiliki banyak kekurangan, baik dari isi maupun penyajiannya. Oleh karena itu, saran dan kritik sangat penulis harapkan guna untuk menyempurnakan laporan berikutnya.

Padang, 12 November 2025

Aliifah Felda Mufarrihati Salwaa

2511531011

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI.....	ii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang Praktikum	1
1.2 Tujuan Praktikum.....	1
1.3 Manfaat Praktikum.....	1
BAB II PEMBAHASAN	2
2.1 String pada Pemograman Java	2
2.2 Praktikum “BilanganPrima”.....	3
2.3 Praktikum “Mahasiswa”.....	5
2.4 Praktikum “PanggilMahasiswa”	7
2.5 Praktikum “PanggilMahasiswa2”	8
2.6 Praktikum “String1”.....	10
2.7 Praktikum “String2”	12
BAB III KESIMPULAN	14
3.1 Kesimpulan.....	14
3.2 Saran.....	14
DAFTAR PUSTAKA	15

BAB I

PENDAHULUAN

1.1 Latar Belakang Praktikum

String pada pemograman bahasa Java merupakan salah satu hal yang paling sering kita gunakan bahkan sejak awal memulai belajar Java. String bukanlah tipe data primitif, tetapi merupakan sebuah objek yang mewakili sebuah kata atau kalimat. Penggunaan string dapat bermacam-macam seperti penggunaan langsung sebagai literal (menggunakan tanda kutip), sebagai kontrol dalam pernyataan switch, dan juga dapat disimpan dalam array.

Sebagai pemula dalam menggunakan bahasa pemograman Java, string terkadang belum dipahami sepenuhnya. Mahasiswa yang masih awam memahami string sebagai tipe data yang dapat dipanggil kembali dalam proses pembuatan kode. Tetapi nyatanya, penggunaan string dapat lebih dari itu. Oleh karena itu, untuk memahami lebih dalam tentang penggunaan dan fungsi dari string, praktikum pekan ini dilaksanakan.

1.2 Tujuan Praktikum

Tujuan dari pelaksanaan praktikum ini adalah sebagai berikut:

1. Mengetahui tata cara penggunaan string dalam pemograman Java.
2. Mengetahui fungsi string dalam pemograman Java
3. Mengetahui hasil dari string dalam pemograman Java.

1.3 Manfaat Praktikum

Manfaat dari pelaksanaan praktikum ini adalah sebagai berikut:

1. Memahami bagaimana sebuah kode pada bahasa Java dapat dijalankan dengan menggunakan string.
2. Mampu mengimplementasikan fungsi dari string dalam pemograman Java.
3. Meningkatkan kemampuan dalam penulisan kode program Java dengan memahami penggunaan dan fungsi string pada.

BAB II PEMBAHASAN

2.1. String pada Pemograman Java

String seringkali salah diartikan sebagai sebuah tipe data primitif oleh individu yang baru saja belajar pemograman bahasa Java. Meskipun sering digunakan seperti tipe dasar, string adalah sebuah objek yang didefinisikan di dalam *library* standar Java. Hal ini memungkinkan string memiliki metode bawaan untuk memanipulasi sebuah teks.

String mewakili kumpulan huruf dan biasanya kita membuat dengan tanda kutip ganda seperti: “Hello World” “String in java is so powerfull” dan sebagainya yang berbentuk sebuah kata atau kalimat. String pada pemograman Java juga bersifat *immutable* atau tidak dapat diubah setelah string tersebut dibuat. Operasi seperti penggabungan (+) sejatinya membuat objek string baru, bukan mengubah objek aslinya.

Sebagai objek, string memiliki beberapa metode bawaan, diantaranya:

- `length()`; untuk mendapatkan jumlah karakter dalam string.
- `charAt(int index)`; untuk mendapatkan karakter pada indeks tertentu.
- `equals(String)`; untuk membandingkan isi dua string.
- `compareTo()`; untuk membandingkan dua string secara leksiografis (urutan).
- `indexOf(String)`; untuk mencari indeks pertama kemunculan substring.
- `lastIndexOf(String)`; untuk mencari indeks terakhir kemunculan substring.
- `Substring()`; untuk megambil bagian dari string.
- `Concat()`; untuk menggabungkan dua string.

2.2. Praktikum “BilanganPrima”

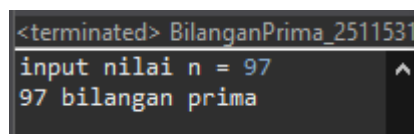
```
1. package pekan7_2511531011;
2.
3. import java.util.Scanner;
4.
5. public class BilanganPrima_2511531011 {
6.
7.     public static boolean isPrime(int n) {
8.         int factors=0;
9.         for (int i=1; i<=n; i++) {
10.             if (n%i==0) {
11.                 factors++;
12.             }
13.         }
14.         return (factors==2);
15.     }
16.
17.     public static void main(String[] args) {
18.         Scanner input = new Scanner(System.in);
19.         System.out.print("input nilai n = ");
20.         int a = input.nextInt();
21.         if (isPrime(a)) {
22.             System.out.println(a+" bilangan prima");
23.         } else {
24.             System.out.println(a+" bukan bilangan prima");
25.         }
26.         input.close();
27.
28.     }
29.
30. }
```

Kode Program 2.1: Praktikum “BilanganPrima”

Pada kode program ini kita menggunakan dua metode pemrograman Java untuk mengetahui apakah data yang di *input* oleh user merupakan sebuah bilangan prima. Berikut langkah-langkah dan penjelasan lengkap mengenai kode program diatas:

1. Pada baris kode 3 kita melakukan inisialisasi *scanner* agar program dapat mengambil *input* dari user.
2. Pada baris kode 7 kita melakukan inisialisasi metode pertama yakni metode boolean *isPrime* dengan parameter *int n*.
3. Pada baris kode 8 kita mendeklarasikan variabel *factors=0*; variabel ini akan digunakan untuk menghitung jumlah faktor dari variabel *n*.
4. Pada baris 9-15 kita menggunakan perulangan *for* dimana:
 - a. Inisialisasi nilai *i=1*
 - b. Kondisi dimana *i<=n*

- c. Iterasi dengan operator *increment* (dimana nilai *i* akan terus bertambah selama perulangan berlangsung).
 - d. Di dalam perulangan, kita menggunakan *statement if* dimana jika *n* modulus 1=0, maka variabel *factors* akan bertambah.
 - e. Kemudian program akan melakukan pengecekan dengan *factors==2*; dimana jika nilai akhir dari variabel *factors* adalah 2, maka metode akan mengembalikan nilai *true* dan jika tidak akan bernilai *false*. *factors==2* kita gunakan karena syarat dari suatu bilangan prima itu adalah ketika bilangan tersebut habis dibagi dengan 1 dan dirinya sendiri. Dimana ini berarti bilangan tersebut hanya memiliki 2 faktor (1 dan dirinya sendiri).
5. Pada baris kode 17 kita mendeklarasikan metode baru yakni metode *main* yang akan menjadi titik awal eksekusi program.
6. Pada baris kode 19-20 kita akan meminta *input* dari user dan memasukkannya ke dalam variabel *int a*.
7. Pada baris kode 21-25 kita menggunakan *statement if-else* dengan memanggil metode *isPrime(a)* untuk melakukan pengecekan pada variabel *a*, dimana:
- a. Jika *isPrime(a)* mengembalikan nilai *true*, maka program akan mencetak bahwasanya *a* adalah bilangan prima.
 - b. Jika *isPrime(a)* mengembalikan nilai *false*, maka program akan mencetak *a* bukan bilangan prima.

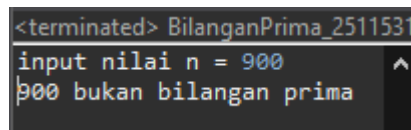


```
<terminated> BilanganPrima_2511531
input nilai n = 97
97 bilangan prima
```

Gambar 2.1: *output* 1 kode program “BilanganPrima”

Gambar 2.1 merupakan salah satu *output* yang dihasilkan oleh program BilanganPrima. Awalnya program akan meminta user untuk menginputkan nilai dari variabel *n* (berupa angka). Pada kasus ini user memasukkan input 97. Kemudian program akan menjalankan seluruh rangkaian seperti langkah-langkah yang telah dijelaskan diatas. Terakhir, program akan mencetak hasil proses yang

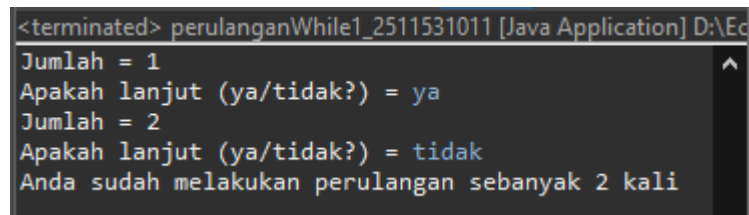
dimana pada kasus ini 97 adalah bilangan prima. Berikut contoh lain dari *output* kode program BilanganPrima (jika *isPrime(a)* bernilai *false*):



```
<terminated> BilanganPrima_2511531
input nilai n = 900
900 bukan bilangan prima
```

Gambar 2.2: *output* 2 kode program “BilanganPrima”

Program akan terus mengeluarkan *output* yang sama hingga user memasukkan kata “tidak”. Hal ini terjadi karena kita menggunakan string Java *equalsIgnoreCase()* dimana metode ini akan membandingkan antara dua buah string dengan mengabaikan huruf besar dan huruf kecil. Dalam kasus ini, program akan membandingkan antara *input* yang diberikan user (variabel *jawab*) dengan kata “tidak”. Jika ditemukan kesamaan, maka program akan menjalankan perintah selanjutnya yakni menjadikan variabel *running* bernilai *false* dan melakukan *print out*. Berikut contoh *output* jika user melakukan *input* “tidak”:



```
<terminated> perulanganWhile1_2511531011 [Java Application] D:\Ec
Jumlah = 1
Apakah lanjut (ya/tidak?) = ya
Jumlah = 2
Apakah lanjut (ya/tidak?) = tidak
Anda sudah melakukan perulangan sebanyak 2 kali
```

Gambar 2.3: *output* 3 dari kode program “perulanganWhile1”

2.3 Praktikum “Mahasiswa”

```
1. package pekan7_2511531011;
2.
3. public class Mahasiswa_2511531011 {
4.     //variabel global
5.     private int nim;
6.     private String nama, nim2;
7.     //membuat mutator(setter)
8.     public void setNim (int nim) {
9.         this.nim=nim;
10.    }
11.    public void setNim2 (String nim2) {
12.        this.nim2=nim2;
13.    }
14.    public void setNama (String nama) {
15.        this.nama=nama;
```



```

16.     }
17.     //membuat accessor(getter)
18.     public int getNim () {
19.         return nim;
20.     }
21.     public String getNim2 () {
22.         return nim2;
23.     }
24.     public String getNama () {
25.         return nama;
26.     }
27.     //metode lain
28.     public void Cetak () {
29.         System.out.println("Nim    : "+nim);
30.         System.out.println("Nama   : "+nama);
31.     }
32.     public void Cetak2 () {
33.         System.out.println("Nim    : "+nim2);
34.         System.out.println("Nama   : "+nama);
35.     }
36. }

```

Kode Program 2.2: praktikum “Mahasiswa”

Pada kode pemograman ini kita melakukan deklarasi berbagai metode. Hal ini dilakukan agar metode yang ada disini dapat dipanggil kembali pada class yang berbeda sehingga variabel yang ada pada program ini dapat digunakan berulang kali. Berikut langkah-langkah dan penjelasan lengkap dari program yang ada diatas:

1. Pada baris kode 5-6 kita melakukan inisialisasi variabel private int nim dan String nama, nim2. Variabel private ini hanya bisa digunakan di dalam kelas ini saja.
2. Pada baris kode 7-9 kita akan membuat mutator (*setter*), yakni metode untuk mengatur nilai dari variabel privat. Kita menggunakan metode setNim untuk mengisi nilai variabel nim. Kita juga menggunakan this.nim untuk membedakan antara nim paramater dan nim kelas.
3. Pada baris kode 11-12 kita menggunakan metode setNim2 untuk mengisi nilai variabel nim2. Kita juga menggunakan this.nim2 untuk membedakan antara nim2 parameter dan nim2 variabel.
4. Pada baris kode 14-15 kita menggunakan metode setNama untuk mengisi variabel nama. Kita juga menggunakan this.nama untuk membedakan antara nama parameter dengan nama variabel.

5. Pada baris kode 17 kita akan masuk dalam pembuatan *getter* (*accessor*), yakni metode untuk mengembalikan nilai dari variabel privat.
6. Pada baris kode 18-19 kita akan mengembalikan nilai dari variabel *nim* dengan menggunakan *getNim* dan tipe kembalian adalah *integer*.
7. Pada baris kode 21-25 kita akan mengembalikan nilai dari variabel *nim2* dan *nama* dengan menggunakan *getNim2* dan *getNama*. Tipe kembalian dari dua variabel ini adalah *string*.
8. Pada baris 27-35 kita akan menggunakan metode lain selain *setter* dan *getter*. Kita melakukan metode cetak untuk menampilkan *nim* dan *nama*. Selain itu kita juga menggunakan metode cetak2 untuk menampilkan *nim2* yang berupa *string* dan *nama*.

Kode program kali ini tidak menghasilkan *output* apapun. Hal ini dikarenakan kita tidak menggunakan metode *main* yang menjadi titik masuk eksekusi. Kode program ini juga tidak melakukan pemanggilan *output*. Walaupun ada metode *Cetak()* dan *Cetak2()* tetapi metode ini tidak sekalipun dipanggil. Sebuah metode akan dapat berjalan ketika ia dipanggil.

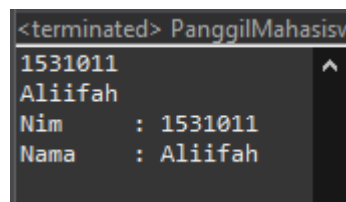
2.4 Praktikum “PanggilMahasiswa”

```
1. package pekan7_2511531011;
2.
3. public class PanggilMahasiswa_2511531011 {
4.
5.     public static void main(String[] args) {
6.         Mahasiswa_2511531011 a=new Mahasiswa_2511531011();
7.         a.setNim (1531011);
8.         a.setNama("Aliifah");
9.         System.out.println(a.getNim());
10.        System.out.println(a.getNama());
11.        a.Cetak();
12.
13.    }
14.
15. }
```

Kode Program 2.3: Praktikum “PanggilMahasiswa”

Pada kode program ini, kita akan melakukan pemanggilan terhadap metode-metode yang telah kita deklarasikan pada praktikum “Mahasiswa”. Berikut untuk langkah-langkah dan penjelasan lengkap tentang kode program diatas:

1. Pada baris kode 6 kita melakukan deklarasi bahwa variabel *a* yang akan menampung referensi ke objek dari kelas “Mahasiswa_2511531011” dimana kita telah mendeklarasikan seluruh metode disana.
2. Pada baris kode 7-8 kita memanggil mutator *setNim* dan *setNama*. Metode ini mengubah nilai dari variabel *nim* dan *nama* lalu menyimpannya dalam objek *a*.
3. Pada baris kode 9-10 kita memanggil *accessor* *getNim* dan *getNama*. Metode ini mengembalikan nilai dari variabel *nim* dan *nama*, lalu kemudian mencetak nilai dari *a.getNim* dan *a.getNama*.
4. Pada baris kode 11 kita akan memanggil metode lain yakni metode *cetak*.



```
<terminated> PanggilMahasiswa
1531011
Aliifah
Nim    : 1531011
Nama   : Aliifah
```

Gambar 2.4: *output* kode program “PanggilMahasiswa”

Gambar 2.4 merupakan *output* yang dihasilkan oleh program “PanggilMahasiswa” dimana program akan mengeluarkan *output* dari perintah *system print* dan *a.Cetak()*. *Nim* dan *nama* yang dihasilkan juga merupakan perubahan dari variabel pada kelas “Mahasiswa” dan merupakan karakter apapun yang kita inisialisasikan pada *a.setNim* dan *a.setNama*.

2.5 Praktikum “PanggilMahasiswa2”

```
1. package pekan7_2511531011;
2.
3. import java.util.Scanner;
4.
5. public class PanggilMahasiswa2_2511531011 {
6.
7.     public static void main(String[] args) {
8.         Scanner input = new Scanner(System.in);
```

```

9.      System.out.print("Nim      : ");
10.     String x=input.nextLine();
11.     System.out.print("Nama      : ");
12.     String y=input.nextLine();
13.     Mahasiswa_2511531011 a= new Mahasiswa_2511531011();
14.     a.setNim2(x);
15.     a.setNama(y);
16.     if (x.startsWith("25")) {
17.         System.out.println(y+" anda angkatan 2025");
18.     }
19.     if (x.contains("1153")) {
20.         System.out.println("Anda Mahasiswa Informatika");
21.     }
22.     a.Cetak2();
23.     input.close();
24.
25. }
26.
27. }

```

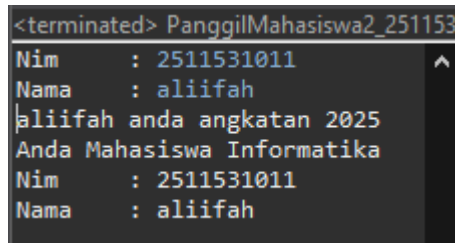
Kode Program 2.4: Praktikum “PanggilMahasiswa2”

Pada kode program kali ini kita akan melakukan pemanggilan kedua dari kelas “Mahasiswa”.

1. Pada baris kode 3 kita inialisasi `java.util.Scanner` untuk melakukan *input*.
2. Pada baris kode 9-12 kita melakukan inialisasi variabel string `x` dan memasukkan input nim ke dalam variabel tersebut. Kita juga melakukan inialisasi variabel `y` dan memasukkan input nama ke dalam variabel tersebut.
3. Pada baris kode 13 kita melakukan deklarasi bahwa variabel `a` yang akan menampung referensi ke objek dari kelas “`Mahasiswa_2511531011`” dimana kita telah mendeklarasikan seluruh metode disana.
4. Pada baris 14-15 kita membuat `a.setNim2(x)` dan `a.setNama(y)` dimana ini akan mengatur nilai `nim2` dan nilai nama dari objek `a`.
5. Pada baris kode 16-18 kita akan memanggil metode `startsWith(25)` dari kelas `string` dimana metode ini akan memeriksa apakah string `x` dimulai dengan 25. Jika ya, maka program akan mengeluarkan *output* berupa variabel `y` dan kalimat bahwasanya user adalah angkatan 2025.
6. Pada baris kode 19-21 kita akan memanggil metode `contains(“1153”) dari kelas string dimana metode ini akan memeriksa apakah string y`

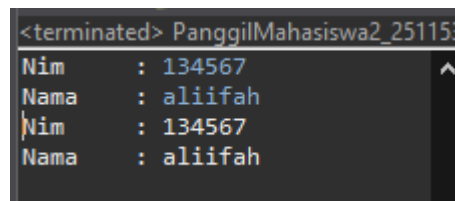
mengandung karakter “1153”. Jika ya, maka program akan mengeluarkan *output* berupa anda mahasiswa informatika.

7. Pada baris kode 22 kita memanggil metode cetak2 yang sudah tersedia dari kumpulan metode pada kelas “Mahasiswa_2511531011”.



```
<terminated> PanggilMahasiswa2_251153
Nim      : 2511531011
Nama     : aliifah
aliifah anda angkatan 2025
Anda Mahasiswa Informatika
Nim      : 2511531011
Nama     : aliifah
```

Gambar 2.5: salah satu *output* kode program “PanggilMahasiswa2”

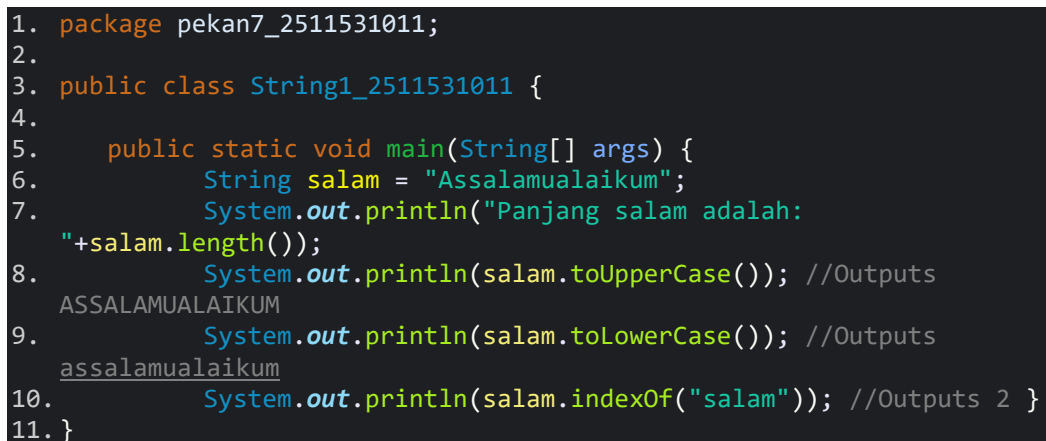


```
<terminated> PanggilMahasiswa2_251153
Nim      : 134567
Nama     : aliifah
Nim      : 134567
Nama     : aliifah
```

Gambar 2.6: salah satu *output* kode program “PanggilMahasiswa2”

Dari kedua *output* ini kita dapat membandingkan bahwasanya *statement if* (program pemeriksa) akan mengeluarkan *output* ketika variabel x dimulai dengan 25 dan mengandung 1153. Jika tidak ada, maka *statement if* akan mengembalikan nilai *false* dan tidak akan mencetak angkatan 2025 dan mahasiswa informatika.

2.6 Praktikum “String1”

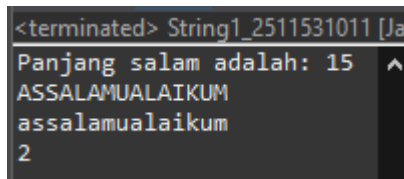


```
1. package pekan7_2511531011;
2.
3. public class String1_2511531011 {
4.
5.     public static void main(String[] args) {
6.         String salam = "Assalamualaikum";
7.         System.out.println("Panjang salam adalah:
8.         "+salam.length());
9.         System.out.println(salam.toUpperCase()); //Outputs
10.        ASSALAMUALAIKUM
11.        System.out.println(salam.toLowerCase()); //Outputs
12.        assalamualaikum
13.        System.out.println(salam.indexOf("salam")); //Outputs 2 }
14. }
```

Kode Program 2.5: kode program praktikum “String1”

Pada kode pemograman ini, kita akan memanggil dan menguji beberapa metode dari objek string. Berikut merupakan langkah-langkah dan penjelasan lengkap mengenai program diatas:

1. Pada baris kode 6 kita melakukan deklarasi dari variabel String `salam="Assalamualaikum"`.
2. Pada baris kode 7 kita akan menentukan panjang dari karakter salam dengan menggunakan metode `salam.length()`. Dimana nanti program akan mencetak nilai dari panjang karakter salam.
3. Pada baris kode 8 kita akan melakukan perubahan terhadap variabel salam dimana kita akan mengubah nilai dari variabel salam menjadi *uppercase* dengan menggunakan metode string `salam.toUpperCase()`.
4. Pada baris kode 9 kita akan melakukan perubahan terhadap variabel salam dimana kita akan mengubah nilai dari variabel salam menjadi *lowercase* dengan menggunakan metode string `salam.toLowerCase()`.
5. Pada baris kode 10 kita akan memeriksa index dari karakter pertama yang kita ingin uji. Dalam kasus ini kita ingin mendapatkan indeks karakter pertama kata "salam" dari variabel salam yang kita punya. Disini karakter "s" berada pada indeks kedua dalam variabel `salam=Assalamualaikum`.



```
<terminated> String1_2511531011 [Java]
Panjang salam adalah: 15
ASSALAMUALAIKUM
assalamualaikum
2
```

Gambar 2.7: *output* kode program "String1"

Output ini menjelaskan bahwasanya panjang karakter dari variabel salam adalah 15. Disini kita juga mendapatkan hasil berupa *uppercase* dan *lowercase* sesuai dengan metode string yang telah kita tuliskan. Kita juga mendapatkan hasil dari indeks "salam" adalah 2.

2.7 Praktikum “String2”

```
package pekan7_2511531011;

import java.util.Scanner;

public class String2_2511531011 {

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Nama Depan: ");
        String firstName = input.nextLine();
        System.out.print("Nama Belakang: ");
        String lastName = input.nextLine();
        String txt1 = "Dosen\"Intelektual\" kampus";
        System.out.println("Nama Lengkap: "+firstName+"
"+lastName);
        System.out.println("Nama Lengkap:
"+firstName.concat(lastName));
        System.out.println(txt1);
        int x = 10;
        int y = 20;
        int z = x+y;
        System.out.println("X + y = "+z);
        String a = "10";
        String b = "20";
        String c = a+b;
        System.out.println("String a + String b = "+c);
        String v = a + y;
        System.out.println("String a + integer y = "+v);

        input.close();

    }

}
```

Kode Program 2.6: Praktikum “String2”

Pada kode program ini kita akan melakukan beberapa proses dan operasi dengan menggunakan objek string. Kita akan mempelajari berbagai cara untuk menampilkan objek string. Berikut langkah-langkah dan penjelasan lengkap dari program diatas:

1. Pada baris kode 3 kita melakukan import java.util.Scanner untuk membaca *input* dari user.
2. Pada baris kode 9-12 kita meminta *input* dari user berupa nama depan dan nama belakang. Nama depan akan disimpan dalam variabel *firstName* dan nama belakang akan disimpan dalam variabel *lastName*.
3. Pada baris kode 13 kita melakukan insialisasi variabel *txt1*.

4. Pada baris kode 14 kita akan membuat program mengeluarkan *output* berupa nama depan dan nama belakang. Disini kita menggunakan cara seperti biasa yakni dengan menggunakan operator (+); `firstName+lastName`.
5. Pada baris kode 15 kita juga ingin mencetak nama depan dan nama belakang. Disini kita menggunakan `firstName.concat(lastName)` dimana metode `concat` ini menggabungkan string `firstName` dengan `lastName` dan mengembalikan string baru.
6. Pada baris kode 16 kita akan melakukan cetak `txt1`.
7. Pada baris kode 17-19 kita melakukan inisialisasi variabel integer `x=10`, `y=20`, dan `z=x+y`
8. Pada baris kode 20 program akan menampilkan hasil dari penjumlahan integer `x+y=z`.
9. Pada baris kode 21-23 kita melakukan inisialisasi variabel string `a=10`, `b=20`, `c=a+b`.
10. Pada baris kode 24 program akan menampilkan hasil dari penjumlahan string `a+b=c`.
11. Pada baris kode 25-16 kita melakukan inisialisasi variabel string `v=a+y` dan program akan menampilkan hasil dari penjumlahan string `a` dengan integer `y=v`.

```
Nama Depan: aliifah
Nama Belakang: felda
Nama Lengkap: aliifah felda
Nama Lengkap: aliifahfelda
Dosen"Intelektual" kampus
X + y = 30
String a + String b = 1020
String a + integer y = 1020
```

Gambar 2.8: *output* kode program “String2”

Pada *output* yang dihasilkan, kita dapat menyimpulkan bahwasanya kita dapat menggabungkan dua string dengan metode `concat`. Kita juga mengetahui bahwasanya penjumlahan integer berbeda dengan penjumlahan string. Jika penjumlahan integer menghasilkan angka baru, perubahan string hanya menggabungkan nilai dari kedua string tersebut.

BAB III

KESIMPULAN

3.1. Kesimpulan

Setelah melakukan praktikum tentang objek string pada pemograman bahasa Java, kita mengetahui bahwasanya string bukanlah tipe data, melainkan sebuah objek. Karena itu, string dapat memiliki beberapa metode. Memaksimalkan penggunaan string akan sangat membantu kita jika hendak membuat sebuah kode program kedepannya.

Penggunaan string dan metode-metodenya terkadang akan membuat program kita lebih efektif dan efisien. Contoh seperti pemanggilan, pembuatan *uppercase*, *lowercase*, penghitungan kata, dan penjumlahan string akan sangat membantu kita dalam memperpendek sebuah kode program sehingga kita tidak akan membutuhkan waktu yang lama untuk membuat sebuah program.

3.2. Saran

Agar lebih mahir dan lebih memahami tentang string pada pemograman bahasa Java dan metode-metodenya, hendaknya mahasiswa banyak bereksperimen sendiri dalam menulis kode program. Dengan memahami string dan metodenya secara mendalam, kita akan lebih mudah dan lebih menghemat waktu dalam menulis sebuah kode pemograman. String dan metode-metodenya juga sering dipakai dan menjadi dasar dalam beberapa program.

DAFTAR PUSTAKA

- [1] H. Schildt, *Java: A Beginner's Guide*, 6th ed. New York, NY: McGraw-Hill Education, 2014
- [2] "Java Encapsulation," *W3Schools* [Daring]. Tersedia pada: https://www.w3schools.com/java/java_encapsulation.asp [Diakses: 13-Nov-2025]