Praktikum Minggu 4

1. Konsep char dan string di Java.

Di dalam bahasa pemrograman Java, karakter (character) adalah tipe data yang merepresentasikan satu huruf, angka, simbol, atau karakter khusus lainnya. Karakter dapat direpresentasikan menggunakan tanda kutip tunggal (') dan setiap karakter memiliki nilai numerik dalam bentuk kode ASCII.

Contoh penggunaan karakter dalam Java:

```
char huruf = 'a';

char angka = '1';

char simbol = '#';
```

String, di sisi lain, adalah tipe data yang merepresentasikan kumpulan karakter. Dalam Java, String direpresentasikan menggunakan tanda kutip ganda ("). Untuk menggabungkan beberapa String dapat menggunakan operator "+" atau dengan method concat().

Contoh penggunaan String dalam Java:

```
String nama = "John Doe";

String alamat = "Jl. Jend. Sudirman";

String kota = "Jakarta";

String alamatLengkap = alamat + ", " + kota;

String sapaan = "Halo, " + nama + "!";

String sapaanLengkap = sapaan.concat(" Alamat kamu di " + alamatLengkap);
```

Perlu diingat bahwa karena String adalah objek, Anda harus menggunakan method equals() untuk membandingkan dua String, bukan operator "==".

2. Identifikasi permasalahan string

Beberapa permasalahan yang umum terjadi dalam manipulasi data string adalah sebagai berikut:

1. Panjang String: Panjang string yang sangat panjang dapat memakan banyak memori dan memperlambat kinerja program. Sebagai contoh, jika Anda perlu memproses sebuah file besar yang berisi banyak data string, Anda mungkin perlu

mempertimbangkan untuk memproses data tersebut dalam batch atau menggunakan teknik kompresi data.

2. Kesalahan penulisan: Kesalahan penulisan dapat menyebabkan program menghasilkan output yang salah atau bahkan error. Sebagai contoh, jika Anda ingin mencetak string "Hello World", tetapi salah mengetik dan menulis "Helo World", maka output yang dihasilkan akan berbeda.

3. Membandingkan string: Membandingkan string memerlukan perhatian ekstra karena perbandingan menggunakan operator == dapat menghasilkan hasil yang tidak diinginkan. Sebagai gantinya, Anda harus menggunakan metode equals() untuk membandingkan dua string.

4. Penggunaan memori yang tidak efisien: Jika Anda perlu menggabungkan beberapa string, mengulang-ulang string, atau melakukan manipulasi string lainnya, mungkin akan ada banyak objek string yang dibuat dalam memori. Ini dapat memakan banyak memori dan memperlambat program Anda. Untuk mengatasi masalah ini, Anda dapat menggunakan StringBuilder atau StringBuffer untuk memanipulasi string secara efisien.

5. Karakter khusus: Beberapa karakter khusus, seperti karakter baris baru (\n), tab (\t), atau karakter kutip ganda (") perlu diperlakukan secara khusus dalam string. Jika Anda lupa menangani karakter khusus ini dengan benar, dapat menyebabkan masalah dalam program Anda. Sebagai contoh, jika Anda ingin mencetak string "Hello" diikuti dengan baris baru dan string "World", Anda perlu menulis "Hello\nWorld".

3. Character class method.

Dalam Java, class Character menyediakan berbagai method untuk memanipulasi karakter. Berikut ini adalah beberapa method yang disediakan oleh class Character:

1. isLetter(char ch): Method ini mengembalikan nilai true jika karakter yang diberikan adalah huruf, baik huruf besar maupun huruf kecil. Jika karakter bukan huruf, maka method ini mengembalikan nilai false.

2. isDigit(char ch): Method ini mengembalikan nilai true jika karakter yang diberikan adalah angka, yaitu 0 sampai 9. Jika karakter bukan angka, maka method ini mengembalikan nilai false.

3. isWhitespace(char ch): Method ini mengembalikan nilai true jika karakter yang diberikan adalah spasi, tab, baris baru, atau karakter whitespace lainnya. Jika karakter bukan whitespace, maka method ini mengembalikan nilai false.

4. isUpperCase(char ch): Method ini mengembalikan nilai true jika karakter yang diberikan adalah huruf besar. Jika karakter bukan huruf besar, maka method ini mengembalikan nilai false.

5. isLowerCase(char ch): Method ini mengembalikan nilai true jika karakter yang diberikan adalah huruf kecil. Jika karakter bukan huruf kecil, maka method ini mengembalikan nilai false.

6. toUpperCase(char ch): Method ini mengembalikan karakter yang sama dengan karakter yang diberikan, tetapi dalam bentuk huruf besar. Jika karakter sudah huruf besar, maka method ini mengembalikan karakter yang sama.

7. toLowerCase(char ch): Method ini mengembalikan karakter yang sama dengan karakter yang diberikan, tetapi dalam bentuk huruf kecil. Jika karakter sudah huruf kecil, maka method ini mengembalikan karakter yang sama.

Dengan menggunakan method-method tersebut, Anda dapat melakukan manipulasi pada karakter untuk memenuhi kebutuhan program Anda.

4. Object string

Dalam Java, String adalah tipe data yang merepresentasikan urutan karakter (sequence of characters) yang tidak dapat diubah (immutable). String dianggap sebagai objek, sehingga memiliki sifat-sifat objek seperti dapat dibuat (instantiated), dapat digunakan sebagai parameter, dan dapat dikembalikan sebagai nilai dari suatu method.

Objek String dalam Java memiliki banyak method yang berguna untuk memanipulasi dan memeriksa string, seperti:

1. length(): Method ini mengembalikan panjang string (jumlah karakter dalam string).

2. charAt(int index): Method ini mengembalikan karakter pada posisi tertentu (index) dalam string.

3. substring(int startIndex): Method ini mengembalikan sub-string yang dimulai dari posisi tertentu (startIndex) hingga akhir string.

4. substring(int startIndex, int endIndex): Method ini mengembalikan sub-string yang dimulai dari posisi tertentu (startIndex) hingga posisi tertentu lain (endIndex), namun tidak termasuk karakter pada posisi endIndex itu sendiri.

5. equals(Object obj): Method ini membandingkan apakah string yang diberikan sama dengan string saat ini (objek yang memanggil method ini) dalam hal isi dan urutan karakter.

6. indexOf(char ch): Method ini mengembalikan indeks pertama dari karakter yang diberikan dalam string. Jika karakter tidak ditemukan, method ini mengembalikan nilai -1.

7. contains(CharSequence s): Method ini mengembalikan nilai true jika string yang diberikan (s) terdapat dalam string saat ini.

8. replace(char oldChar, char newChar): Method ini mengganti semua kemunculan karakter tertentu (oldChar) dalam string dengan karakter baru (newChar).

9. toUpperCase(): Method ini mengembalikan string dalam bentuk huruf besar.

10. toLowerCase(): Method ini mengembalikan string dalam bentuk huruf kecil.

Dalam Java, String juga dapat diinisialisasi menggunakan literal string (contoh: "Hello World") atau menggunakan constructor String (contoh: new String("Hello World")). Selain itu, Java juga menyediakan kelas StringBuilder dan kelas StringBuffer untuk memanipulasi string secara mutable (dapat diubah). Namun, kelas-kelas tersebut tidak memiliki beberapa method yang dimiliki oleh kelas String.

6. Latihan soal

- Sebuah program memiliki string MOBIL, tentukanlah panjang karakter, char pada index ke-3, kemudian ubahlah string tersebut menjadi mobil.

```
class latihan{

  public static void main(String[] args) {

    String a = "MOBIL";



    System.out.println(a.length());

    System.out.println(a.charAt(3));

    System.out.println(a.toLowerCase());

  }

}
```

- Buatlah program yang mengoutputkan apakah MOBIL == mobil.

```
class latihan{

  public static void main(String[] args) {

    String a = "MOBIL";

    String b = "mobil";

    System.out.println(a.equals(b));

  }
```

```
}
```

A. **Tipe Data Primitif**

Class merupakan model atau prototipe dari objek-objek yang akan digunakan dalam membuat program. Setiap objek yang dibuat dari class yang sama memiliki jenis atribut dan method yang sama.

B. **Membuat Class**

Untuk membuat kelas, gunakan kata kunci Class untuk deklarasinya:

```
public class Person {
    public String nama;
}
```

Membuat class dengan nama **Person** dengan atribut nama pada file **Person.java**. untuk membuat *Object* dari class **Person** adalah dengan menuliskan nama class kemudian diikuti dengan nama *Object* dan setelahnya mengunakan kata kunci **new**.

```
public class Person {
    public String nama;

    public static void main(String[] args) {
        Person Orang_1 = new Person();
    }
}
```

C. **Multiple Class**

Membuat objek kelas dan mengaksesnya di kelas lain. Metode ini sering digunakan untuk organisasi kelas yang lebih baik (satu kelas memiliki semua atribut dan metode, sedangkan kelas lainnya memegang metode main () (kode yang akan dieksekusi)).

Person.java

```
public class Person {
    public String nama;
}
```

DataDiri.java

```
public class DataDiri {
    public static void main(String[] args) {
        Person Orang_1 = new Person();
    }
}
```

D. **Mengakses Atribut pada Class**

Atribut dapat diakses dari class yang sudah dibuat dengan membuat object terlebih dahulu, dan dengan menggunakan syntax titik (.) untuk mengakses atribut tersebut.

```
public class Person {
    public String nama = "Senja";

    public static void main(String[] args) {
        Person Orang_1 = new Person();
        System.out.println(Orang_1.nama);
    }
}
```

Membuat object dengan nama **Orang_1** dan mencetak dengan nilai nama.

E. **Memodifikasi Nilai dari Atribut**

Modifikasi nilai pada atribut dapat dilakukan dengan cara menetukan nilai yang akan digunakan untuk memodifikasi dari atribut

```
public class Person {
    public String nama = "Senja";
    public int umur;
    public static void main(String[] args) {
        Person Orang_1 = new Person();
        Orang_1.umur = 20;
        System.out.println("Nama :"+ Orang_1.nama);
        System.out.println("Umur :"+ Orang_1.umur);
    }
}
```

Atribut umur pada class **Person** yang tidak mempunyai nilai (null) kemudian dilakukan set nilai atribut pada baris ke-6 dengan code Orang_1.umur = 20; sehingga ketika dicetak akan mengeluarkan nilai sebesar nilai yang telah di-set menggunakan object tersebut.

F. **Method**

Method dideklarasikan di dalam kelas, dan digunakan untuk melakukan aksi tertentu yang berguna untuk melakukan sebuah pekerjaan.

```
public class Main {
  static void myMethod() {
    System.out.println("Hello World!");
  }

  public static void main(String[] args) {
    myMethod();
  }
}
```

Method diatas digunakan untuk mencetak kata “Hello World!”.

G. **Mengakses Method Menggunakan Object**

Membuat sebuah object dengan nama “citah” kemudian akses method **kucing** dan **run** pada class **Hewan.** Selanjutnya jalankan program.

```java
public class Hewan {
    public void kucing(){
        System.out.println("Citah Merupakan Kucing Besar");
    }
    public void run(int maxSpeed){
        System.out.println("Citah dapat berlari hingga "+ maxSpeed);
    }

    public static void main(String[] args) {
        Hewan citah = new Hewan();
        citah.kucing();
        citah.run(130);
    }
}
```

H. **Percobaan 1**

Dalam percobaan ini akan dibuat **class Mahasiswa** yang memiliki atribut sebagai berikut:

- **nim**, bertipe String
- **nama**, bertipe String
- **alamat**, bertipe String

dan memiliki method sebagai berikut:

- **setNim**(), berfungsi untuk mengeset data nim
- **setNama**(), berfungsi untuk mengeset data nama
- **setAlamat**(), berfungsi untuk mengeset data alamat
- **getNim**(), berfungsi untuk membaca data nim
- **getNama**(), berfungsi untuk membaca data nama
- **getAlamat**(), berfungsi untuk membaca data alamat

```java
public class Mahasiswa {
    private String nim;
    private String nama;
    private String alamat;

    public void setNim(String pNim) {
        nim = pNim;
    }
    public void setNama(String pNama) {
        nama = pNama;
    }
    public void setAlamat(String pAlamat) {
        alamat = pAlamat;
    }
    public String getNim() {
        return nim;
    }
    public String getNama() {
        return nama;
    }
    public String getAlamat() {
        return alamat;
    }
}
```

Simpan program di atas dengan nama file Mahasiswa.java, lalu lakukan kompilasi. Ingat program diatas tidak dapat dieksekusi karena bukan class utama (main class). Class utama adalah class yang memiliki method “public static void main()”.

```java
public class TestMahasiswa {
    public static void main(String[] args) {
        Mahasiswa mhs1 = new Mahasiswa();
        Mahasiswa mhs2 = new Mahasiswa();
        mhs1.setNim("01");
        mhs1.setNama("Wita");
        mhs1.setAlamat("Mojokerto");
        System.out.println("Nim dari mhs1 = " +mhs1.getNim());
        System.out.println("Nama dari mhs1 = " +mhs1.getNama());
        System.out.println("Alamat dari mhs1 = "+mhs1.getAlamat());
        mhs2.setNim("02");
        mhs2.setNama("Hamdan");
        mhs2.setAlamat("Malang");
        System.out.println("Nim dari mhs2 = " +mhs2.getNim());
        System.out.println("Nama dari mhs2 = " +mhs2.getNama());
        System.out.println("Alamat dari mhs2 = "
        +mhs2.getAlamat());
    }
}
```

I. **Tugas rumah**

1. Analisa dari percobaan 1.
2. Buatlah program untuk menghitung Volume Limas segitiga dengan konsep OOP dengan ketentuan sebagai berikut:
   - Data lebar alas, tinggi alas dan tinggi diinputkan
   - Objek limas segitiga dibuat dari class Limas_Segitiga
   - Class Limas_Segitiga memiliki atribut lebar alas, tinggi alas dan tinggi dan method setLebar,setTinggiAlas, setTinggi, getAlas, getTinggiAlas , getTinggi serta hitungVolume.

   Tulis hasil analisa pada program menghitung Volume Limas Segitiga