MODUL I
Java

A. Tujuan

Pada modul Java ini, mahasiswa diharapkan untuk mampu:

1. Mengenal bahasa pemrograman Java.
2. Menggunakan IntelliJ IDEA untuk menulis kode Java.
3. Mengenal variable, value, dan tipe data yang dikelola di Java.
4. Melakukan operasi-operasi dasar terhadap variable dan value.

B. Teori Dasar

1. Sejarah Java

Java merupakan bahasa pemrograman yang dibuat oleh tim peneliti di perusahaan Sun Microsystems pada tahun 1991. Tim ini dipimpin oleh James Gosling, Mike Sheridan, dan Patrick Naughton dengan tujuan membuat bahasa baru yang memberi kemampuan komunikasi antar perangkat elektronik. Awalnya dinamai dengan Oak, dari pohon oak yang tumbuh diluar kantor Gosling. Java telah berkembang sejak versi 1 di tahun 1996, hingga saat ini versi 16 tahun 2021. Perusahaan Oracle membeli Sun Microsystems pada tahun 2009, sehingga Java kini dimiliki oleh Oracle.

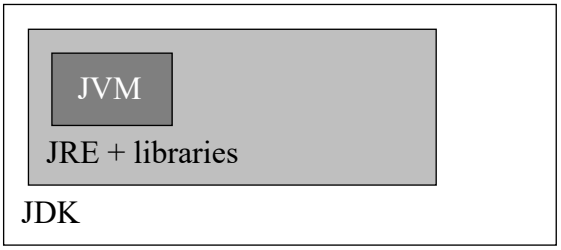
Gambar 1.1 Logo Java

Java telah digunakan oleh berbagai perusahaan ternama seperti Twitter, Netflix, Amazon, dan masih banyak lagi. Java populer digunakan untuk membuat aplikasi back end, big data, dan Android.

2. Java Development Kit

Program Java dibuat menggunakan sejumlah software. Java Development Kit atau JDK merupakan kumpulan software untuk mengembangkan program Java, sebagian diantaranya ialah:

a) Java Runtime Environment atau JRE, untuk menuliskan kode dan menjalankan program Java yang terdiri dari Java Virtual Machine atau JVM, Java Standard Library, dan tool konfigurasi.
b) Java Loader, untuk menginterpretasi file .class.
c) Java Compiler atau javac yang mengkonversi file .java menjadi bytecode.
d) Java Archiver atau jar yang menjadi format file program Java.
e) Java Documentation yang membuat dokumentasi bahasa Java.



Gambar 2 Arsitektur Java

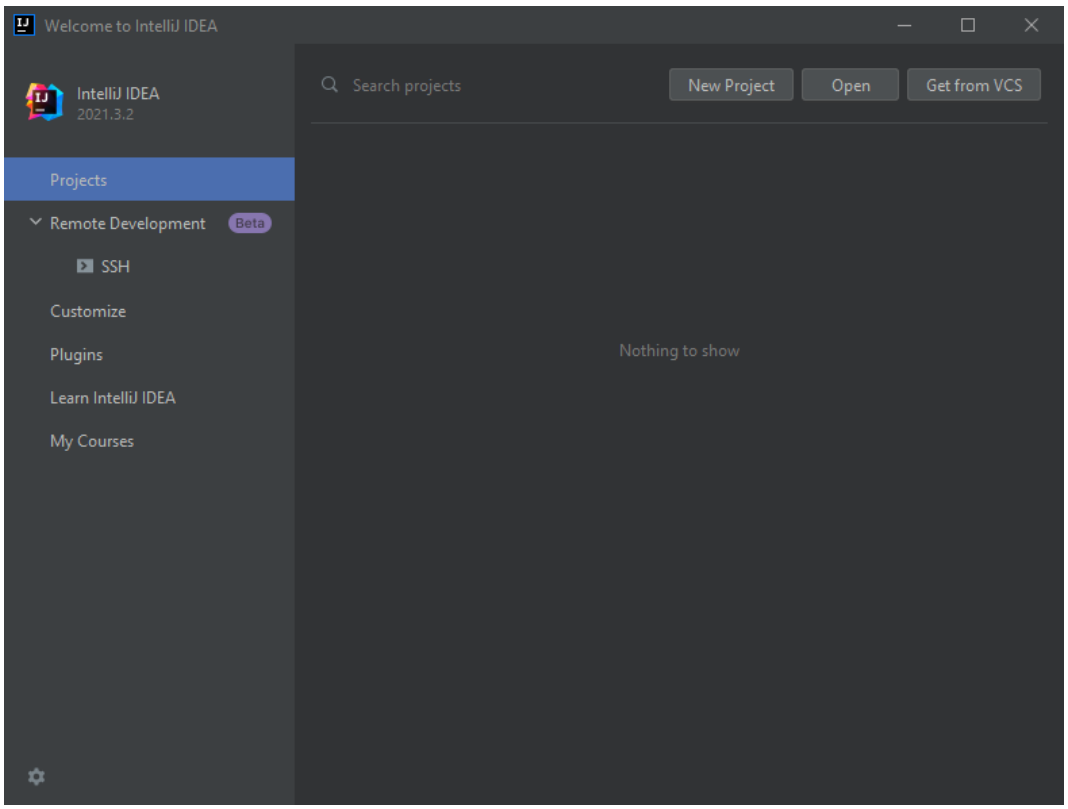
Program Java dikembangkan di komputer dalam sejumlah tahapan yang melibatkan JDK, yakni:

1) Edit. Kode Java diketik di editor atau IDE, kemudian disimpan ke sebuah file .java.
2) Compile. Suatu program yang disebut compiler menggunakan file .java tadi untuk membuat bytecode dan menyimpannya ke dalam sebuah file .class.
3) Load. Komponen dari JRE, yakni Java Class Loader, membaca file .class dan mengambil bytecode di dalamnya untuk disimpan di memory.
4) Verify. Bytecode verifier memeriksa validitas bytecode berdasarkan aturan Java.
5) Execute. JVM membaca bytecode dan melakukan translasi ke bahasa mesin agar bytecode dapat dibaca oleh komputer.

3. IntelliJ IDEA

Salah satu editor untuk menulis kode Java adalah IntelliJ IDEA. Ini merupakan editor Java dengan fitur yang lengkap sehingga disebut dengan Integrated Development Environment atau IDE untuk JVM, sehingga bisa digunakan untuk membangun program Java, Kotlin, Scala, dan Groovy.

Unduh IntelliJ IDEA di tautan https://www.jetbrains.com/. Ketika aplikasi dibuka, akan ditampilkan sebuah Welcome Screen seperti di gambar berikut.



Gambar 1.3 Welcome Screen IntelliJ IDEA

4. Java Syntax

Saat menuliskan kode Java, ada aturan-aturan yang harus dipatuhi. Jika aturan diabaikan, maka error akan muncul. Aturan penulisan kode Java disebut sebagai Java syntax. Salah satu aturannya ialah: setiap barisan perintah harus diakhiri dengan tanda titik koma. Aturan-aturan lainnya akan dibahas disetiap modul ini. Aturan tersebut antara lain:

a) Case-sensitive, yang mengartikan bahwa Java membedakan antara huruf kecil dengan huruf besar. Sebagai contoh, variabel bernama `usia` dengan `uSia` adalah dua variabel yang berbeda bagi Java.

b) Whitespace, merupakan setiap area kosong pada kode. Whitespace dapat berupa spasi, baris kosong, dan indentasi di setiap baris kode. Whitespace digunakan untuk mengorganisir kode agar mudah untuk dibaca.

c) Indentation, merupakan sebuah tab dari tombol tab di keyboard. Indentation dimulai dengan kurung kurawal pembuka atau opening curly brace `{`, lalu ditutup dengan tab sebelum kurung kurawal penutup atau closing curly brace `}`. Kode di dalam kurung kurawal atau curly brace disebut dengan **block of code** atau **kode blok**.

5. Java First Program

Kode yang dipelajari di modul ini harus diketik diantara perintah `public static void main (String[]args)` di baris ke-2 dan tanda tutup kurung di baris ke-4 di contoh kode berikut. Kode di baris ke-2 disebut dengan istilah **main method**, yakni titik awal dari setiap program Java. Setiap program Java hanya berisi satu buah main method.

```
1 public class HelloWorld {
2     public static void main(String[] args) {
3         // Write your code here
4     }
5 }
```

Kedua tanda tutup kurung di baris ke-4 dan ke-5 bisa diturunkan ke baris di bawahnya jika kode Anda terdiri dari beberapa baris. Kode di atas memiliki dua kode blok, yakni dari baris ke-2 hingga baris ke-4, serta dari baris ke-3.

6. Variable

Java dapat menyimpan data seperti judul buku, harga buku, dan lain-lain di dalam memori komputer. Data akan disimpan melalui variable menjadi value. Variable adalah nama lokasi memori dari data yang tersimpan. Sintaks dari variable dituliskan dengan urutan: tipe variable, nama dari variable, tanda sama dengan, value yang akan disimpan dan diakhiri titik koma. Nama dari variable dapat disebut dengan istilah **identifier**.

Nama variable ditulis dengan awalan huruf kecil, dan jika terdiri dari dua kata atau lebih, maka seluruhnya digabungkan, dan huruf pertama di kata selanjutnya harus huruf kapital. Perhatikan penulisan kode deklarasi atau pembuatan variable di bawah ini.

```
1 int bookPrice = 175000;
2 System.out.println(bookPrice);
```

```
Output
175000
```

Pada baris ke-1, kode `int` adalah tipe variable, sedangkan `bookPrice` adalah nama variable, dan `175000` merupakan value yang disimpan oleh variable. Perintah di kode ini disebut dengan istilah **initialization** atau **inisialisasi**, yakni penyimpanan value ke variable.

Sedangkan di baris ke-2, yakni `System.out.println` adalah perintah untuk menampilkan value. Perintah ini membutuhkan nama variable yang menyimpan value dan harus ditulis diantara tanda kurung buka dan tutup.

7. Tipe Data

Data yang dikelola oleh Java dapat dikelompokkan menjadi dua tipe, yakni:

a) Primitive, seperti byte, short, int, long, float, double, boolean, dan char. Tipe data ini tidak membutuhkan keyword `new` saat inisialisasi. Ketika inisialisasi, suatu value harus diberikan kepada variable dengan tipe data primitive.
b) Non-primitive, seperti String, Array, dan Class, yang membutuhkan keyword `new` saat inisialisasi.

Data yang berbentuk bilangan bulat merupakan tipe byte, short, int, atau long. Perbedaannya terletak pada banyaknya digit angka di dalam value. Khusu value untuk tipe data long harus diakhiri dengan huruf `L`.

```
1 byte quantity = 9;
2 short numberOfPages = 299;
3 int bookPrice = 175000;
4 long stockInWarehouse = 15000000000L;
```

Sedangkan yang berbentuk bilangan pecahan bertipe float dan double. Value untuk tipe data float harus diakhiri dengan huruf `f`.

```
1 float discount = .1f;
2 double PI = 3.1415926535d;
```

Data yang hanya berbentuk nilai kebenaran, yakni true dan false ialah boolean.

```
1 boolean isAvailable = true;
```

Data yang hanya terdiri dari satu karakter, bertipe char. Value ditulis dengan diawali dan diakhiri mengetik petik tunggal.

```
1 char paperQualityGrade = 'A';
```

Sedangkan untuk value yang terdiri dari sejumlah karakter merupakan bertipe String. Value ditulis diawali dan diakhiri dengan petik ganda.

```
1 String title = "Harry Potter and The Chamber of Secrets";
```

8. Assignment

Jika suatu variable telah memiliki value, maka value tersebut bisa diubah menjadi value yang lain melalui **assignment**. Assignment ialah penetapan value kepada variable. Langkah ini ditulis dengan urutan: nama variable, tanda sama dengan, dan value yang baru.

```
1 int bookPrice = 175000;
2 bookPrice = 120000;
3 System.out.println(bookPrice);
Output
120000
```

Kode di baris-1 adalah initialization untuk variable `bookPrice` dengan value `175000`, sedangkan kode di baris ke-2 adalah assignment value baru, yakni `120000` ke variable `bookPrice`.

Secara default, value bisa diubah dan juga bisa diatur agar tidak bisa diubah menggunakan keyword **final**.

```
1 final String paperSize = "A4";
2 paperSize = "F4";
3 System.out.println(paperSize);
Output
cannot assign a value to final variable paperSize
```

Kode di baris ke-1 adalah initialization variable `paperSize` dengan keyword `final`, sehingga kode di baris ke-2 akan memunculkan error karena ingin mengubah value.

9. Casting

Khusus untuk tipe data primitive, value dengan tipe data primitive tertentu bisa diubah menjadi tipe data primitive lainnya melalui teknik casting. Ada dua tipe casting, yaitu:

a) Widening Casting, yakni konversi dari tipe data berukuran kecil menjadi tipe data berukuran besar.
b) Narrowing Casing, konversi dari tipe data berukuran besar menjadi tipe data berukuran kecil.

```
1 int oneDigit = 7;
2 double twoDigit = oneDigit;
3 System.out.println(twoDigit);
Output
7.0
```

Kode di baris-2 adalah widening casting terhadap value `7` di dalam variable `oneDigit` yang bertipe int menjadi double di variable `twoDigit`.

```
1 double twoDigit = 9.0;
2 int oneDigit = (int) twoDigit;
3 System.out.println(oneDigit);
Output
9
```

Kode di baris ke-2 adalah narrowing casting terhadap value 9.0 di dalam variable `twoDigit` bertipe double menjadi int di variable `oneDigit`. Narrowing casting membutuhkan penulisan tipe data di dalam kurung setelah tanda sama dengan.

10. Operator

Sejumlah value dan variable bisa diterapkan suatu operasi, seperti operasi matematika menggunakan operator. Salah satu operator matematika adalah tanda plus untuk penjumlahan. Seperti halnya penulisan rumus matematika, tanda plus diletakkan diantara kedua value atau variable.

```
1 int sum1 = 1 + 2;
2 int sum2 = sum1 + 3;
3 int sum3 = sum1 + sum2;
4 System.out.println(sum3);
```

Output

```
9
```

Operator pada Java dikelompokkan menjadi:

a) Arithmetic operator untuk melakukan operasi matematika dasar.

b) Assignment operator untuk melakukan penetapan value ke variable.

c) Comparison operator untuk membandingkan dua value.

d) Logical operator untuk menentukan nilai kebenaran dari sejumlah variable atau value.

Operator arithmetic digunakan untuk melakukan operasi dasar di matematika.

```
1 int x = 2;
2 int y = 1;
3 int addition = x + y;
4 int substraction = x - y;
5 int multiplication = x * y;
6 float division = x / y;
7 int modulus = x % y;
8 int xPlusOne = x++;
9 int xMinusOne = y--;
```

Operator assignment digunakan untuk memberikan value ke variable.

```
1 int five = 5;
2 five += 1;
3 five -= 3;
4 five *= 2;
5 five /= 2;
```

Operator comparison digunakan untuk membandingkan dua value. Operator comparison terdiri dari:

a) == untuk membandingkan apakah kedua value sama.

b) != untuk membandingkan apakah kedua value tidak sama.

c) > untuk membandingkan apakah value di kiri operator lebih besar dari value di kanan operator.

d) < untuk membandingkan apakah value di kanan operator lebih besar dari value di kiri operator.

e) >= untuk membandingkan apakah value di kiri operator lebih besar atau sama dengan value di kanan operator.

f) <= untuk membandingkan apakah value di kanan operator lebih besar atau sama dengan value di kiri operator.

```
1 int johnAge = 17;
2 int janeAge = 15;
3 System.out.println(johnAge == janeAge);
```

Output

```
false
```

Kode di baris ke-3 melakukan operasi perbandingan atas value dari variable `johnAge` dengan `janeAge`. Karena 17 tidak sama dengan 15, maka output-nya adalah false.

Sedangkan operator logical digunakan menentukan nilai true dan false menggunakan logika matematika. Operator logical terdiri dari:

a) Logical AND menggunakan tanda && yang hanya mengembalikan nilai true jika kedua statement bernilai true.

b) Logical OR menggunakan tanda || yang hanya mengembalikan nilai false jika kedua statement bernilai false.

c) Logical NOT menggunakan tanda seru ! yang memutarbalikkan nilai statement.

```
1 boolean isTrue = true;
2 System.out.println(isTrue && false);
```

Output

```
false
```

Kode di baris ke-2 akan menerapkan operasi logika matematika dengan memeriksa nilai variable `isTrue` dan value `false`. Karena variable `isTrue` bernilai `true` maka hasil operasi logika matematika ini adalah `false`.

11. String Concatenation

Khusus untuk operator tanda plus, jika diterapkan ke value bertipe String, maka akan terjadi penggabungan String, bukan penjumlahan. Hal ini disebut dengan istilah **string concatenation**.

```
1 String firstName = "John";
2 String lastName = "Doe";
3 System.out.println(firstName + lastName);
```

Output

```
JohnDoe
```

Pada baris ke-3 akan dilakukan string concatenation antara variable `firstName` dengan `lastName` karena keduanya memiliki value bertipe String.

Apabila salah satu value merupakan bilangan, tetap akan dilakukan string concatenation, dan bukan operasi matematika.

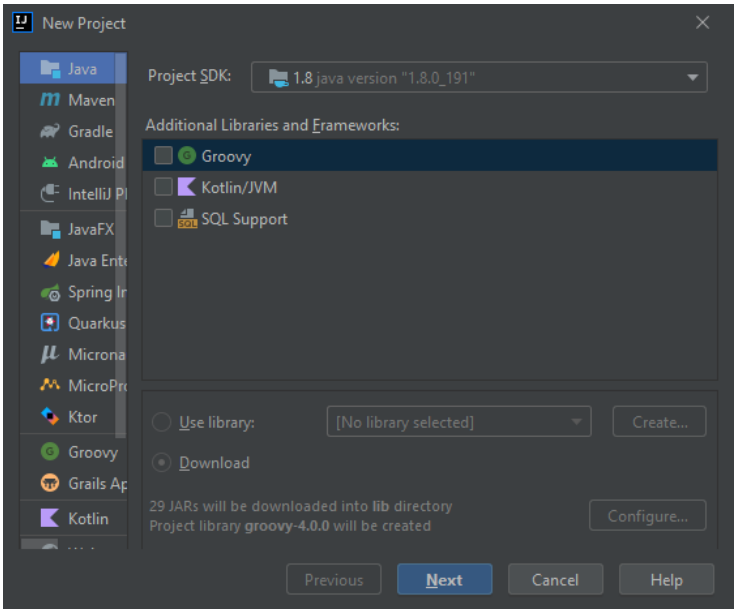
```
1 int number = 7;
2 String word = "Eleven";
3 System.out.println(number + word);
```

Output

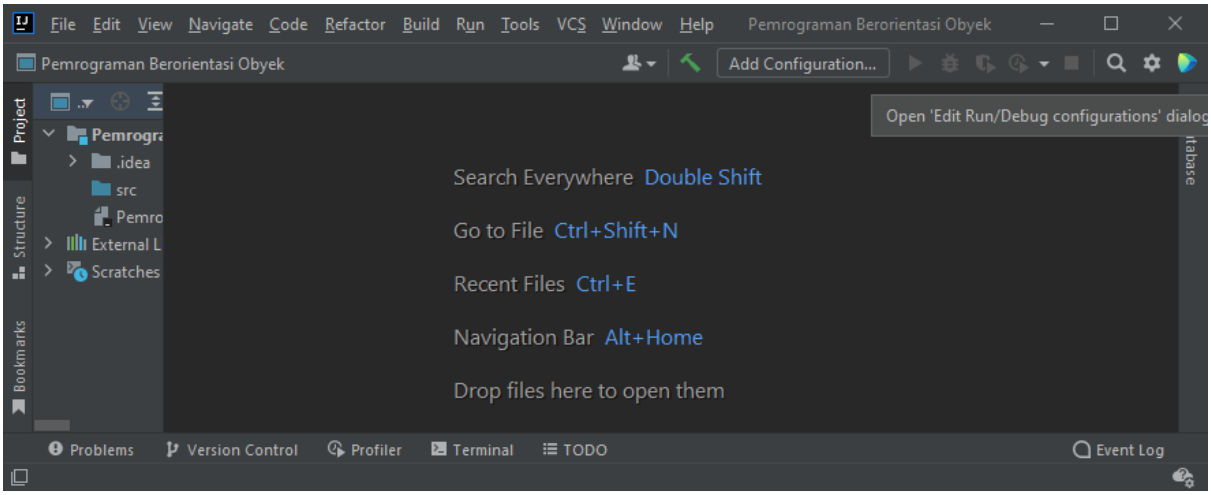
```
7Eleven
```

C. Langkah Kerja

Jalankan aplikasi IntelliJ IDEA, pada Welcome Screen yang muncul, klik tombol **New Project**. Kotak dialog New Project akan muncul. Pada sidebar di sebelah kiri, klik **Java** dan pilih **Project SDK** untuk menggunakan Java versi 1.8. Klik tombol **Next**.



Klik **Next** lagi saat memilih project template, kemudian ketik nama project di **Project name** dengan nama **Pemrograman Berorientasi Obyek**. Tentukan lokasi penyimpanan file dan folder project ini di **Project location**. Kemudian klik **Finish**. Jendela utama IntelliJ IDEA akan muncul.



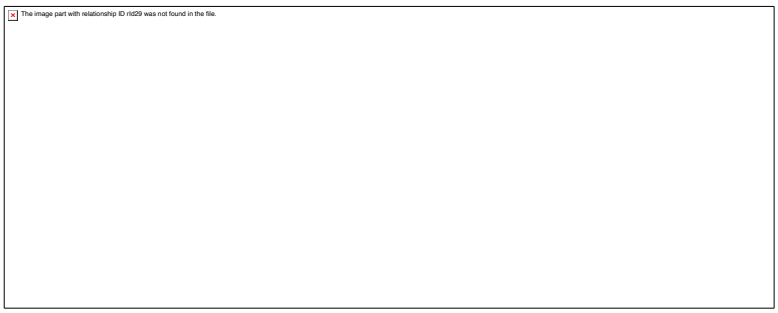
Selanjutnya kita buat sebuah java class di dalam project ini. Pada Project tool window, klik kanan folder **src** lalu pilih **New**, dan klik **Java Class**. Ketik **HelloWorld**. IntelliJ IDEA akan menampilkan kode berikut.

```
1  public class HelloWorld {
2  }
```

Kita akan tampilkan sebuah teks menggunakan program Java ini dengan menambahkan kode lainnya.

```
1  public class HelloWorld {
2      public static void main(String[] args) {
3          System.out.println("Hello World");
4      }
5  }
```

Baris ke-3 adalah perintah untuk menampilkan teks Hello World ke layar. Jalankan program ini dengan memilih menu **Run**, lalu klik **Run** dan pilih **HelloWorld**. Teks akan tampil di Run tool window yang berada di bawah layar.



Selanjutnya ketik kode di subbab Teori Dasar dan jalankan di IntelliJ IDEA.

D. Tugas

Buatlah sebuah file .java bernama PrimitiveExercise. Inisialisasikan variable bookStock, bookIsAvailable, dan bookGrade dengan value 7, true, dan A. Tampilkan seluruh value ke output.