UNIVERSITAS INDONESIA

Veritas, Probitas, Iustitia

FAKULTAS ILMU KOMPUTER

Introduction to Java

Dasar – Dasar Pemrograman 2

Dinial Utami Nurul Qomariah

Tak Kenal Maka tak Sayang ☺

Mari Kita Kenalan Dulu Sama Siapa?

Tujuan

- Memahami prinsip dasar pemrograman dengan Java
- Memahami perbedaan antara Java dan Python
- Memahami jenis-jenis error dalam pemrograman

Credits

- Liang, Introduction to Java Programming, 11th Edition, Ch. 1
- Downey & Mayfield, Think Java: How to Think Like a Computer Scientist, Ch. 1
- Slide Kuliah Dasar-Dasar Pemrograman 2 Semester Genap 2019/2020

Outline

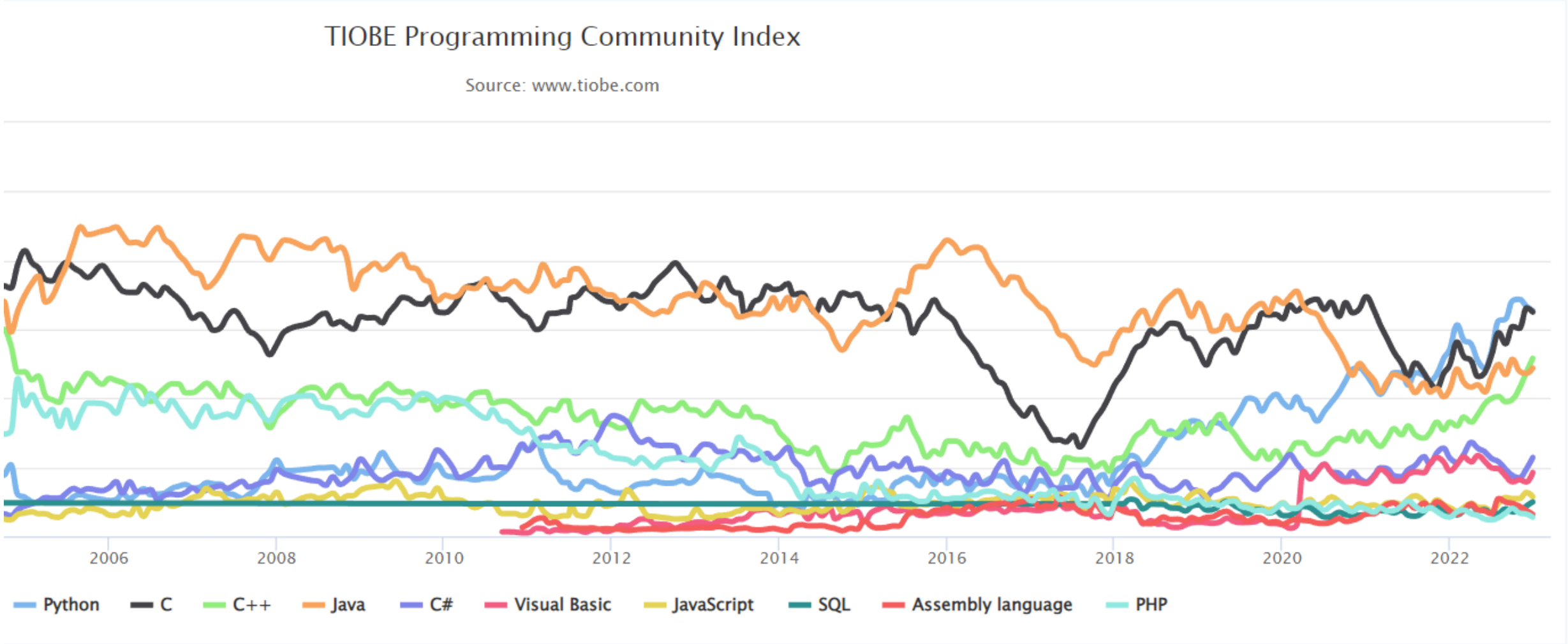
- Java: What and Why
- Java vs. Python
- Java program structure
- Compiling & running a Java program
- Java programming convention
- Programming errors

Kenapa Java Ada Apa Disana?

Why Java ?

- MK Wajib
- Banyak Digunakan
- Aplikasi Multi Platform
- OOP oriented
- Mobile Application
- Server Side Application
- **TERPAKSA!!!**

Perkembangan Bahasa Java



Sun microsystems

1991
Dikembangkan
James Gosling dkk

1994
Bahasa OAK
dari pohon OAK

1996
Release

1992
Green Project
(Website, browser, dll)

1994
Ganti nama
JAVA

2009
Sun Microsystems
dibeli oleh
Oracle

Overview Java

- Platform: Java Standart edition (SE), Java Enterprise edition (EE), Java micro edition (ME).
- Versi LTS Java SE: JDK 8, JDK 11, JDK 17
- Java Development Kit (**JDK**), mengandung
  - Java Runtime Environment (**JRE**): berisi Java Virtual Machine (**JVM**) dan standard library
  - Tools: javac, java, javadoc, jartool, etc

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Characteristics of Java

- Java Is Simple
- Java Is **Object-Oriented**
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

- Membungkus data dan prosedur dalam bentuk objek
- Mendukung reusability dan modularity

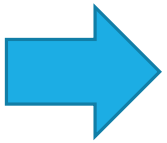
Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is **Interpreted**
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

- Program Java di-compile menjadi byte code.
- Byte code bersifat machine independent dan dapat dijalankan di mesin yang berbeda menggunakan Java interpreter.

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is **Architecture-Neutral**
- Java Is **Portable**
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic



- **WORA** - Write once, run anywhere
- Java Virtual Machine (JVM) memungkinkan program dijalankan di platform yang berbeda tanpa perlu melakukan recompile.

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS ILMU KOMPUTER

Java Vs Python

VS

Java Vs Python

	Python	**Java**
Tipe Bahasa	Bersifat **dynamically** typed	Bersifat **statically** typed
Object-oriented Programming (OOP)	Python mendukung OOP, tapi memungkinkan untuk menulis program Python tanpa memanfaatkan konsep OOP tersebut.	Java hanya mendukung **object-oriented programming.**

Java Vs Python

	Python	Java
Variabel	Variabel diperkenalkan cukup dengan memberi nilai variabel tersebut. Tipe data variabel tersebut bergantung pada data yang diberikan. **someVariable = 42**	Tipe data suatu variabel harus dideklarasikan secara eksplisit sebelum nilai diberikan. **int someVariable;** **someVariable = 42;** ATAU **int someVariable = 42;**

Java Vs Python

	Python	Java
Variabel	Tipe data sebuah variabel mungkin berubah jika nilai yang disimpan berubah. **someVariable = 42** **someVariable = 'Hello, world'**	Variabel yang sudah dideklarasikan sebagai sebuah tipe tidak dapat menerima nilai yang tipenya berbeda. **String someVariable = "hello";** **someVariable = 42;**

Java Vs Python

	Python	Java
Tipe Data	Semua data di Python diperlakukan sebagai **objek**. Tipe data bawaan Python: **1. int** **2. float** **3. bool** **4. str** **5. list** **6. etc.**	Java memiliki dua jenis tipe data: 1. **primitive types** 2. reference types **Tipe data primitif:** **1. byte (8-bit integers)** **2. short (16-bit integers)** **3. int (32-bit integers)** **4. long (64-bit integers)** **5. float (32-bit)** **6. double (64-bit)** **7. boolean (false/true)** **8. char (a single character)**

Java Vs Python

	Python	Java
Operator Perbandingan	Operator perbandingan **(>,<, >=, <=, ==, !=)** dapat diaplikasikan ke data numerik, string, dan sebagainya.	Sebagian besar operator perbandingan **(>, <, >=, <=)** hanya dapat diaplikasikan ke data primitif. Operator **==** dan **!=** dapat diaplikasikan ke tipe data reference, namun yang dicek adalah kesamaan alamat objeknya, bukan kesamaan nilainya.

Java Vs Python

	Python	Java
Penggunaan	Program Python dijalankan menggunakan interpreter. Sebuah baris statement dalam bahasa Python dapat dituliskan dan dieksekusi langsung menggunakan interpreter, tanpa perlu di-compile terlebih dahulu.	Sebagian besar operator perbandingan **(>, <, >=, <=)** hanya dapat diaplikasikan ke data primitif. Operator **==** dan **!=** dapat diaplikasikan ke tipe data reference, namun yang dicek adalah kesamaan alamat objeknya, bukan kesamaan nilainya.

Java Program Structure

Java Vs Python

- Suatu program terbuat dari satu atau lebih class.
- Suatu class terdiri satu atau lebih method.
- Suatu method terdiri dari statement program.
- Suatu aplikasi Java selalu terdiri sebuah method main.

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

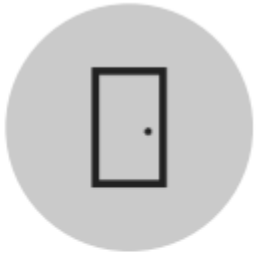
Anatomy of a Java Program

CLASS NAME

MAIN METHOD

STATEMENTS

STATEMENT TERMINATOR



RESERVED WORDS

COMMENTS

BLOCKS

Class Name

- Every Java program must have at least one class.
- Each class has a name.
- By convention, class names start with an **uppercase** letter.
- In this example, the class name is Welcome.

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Main Method

- Line 2 defines the main method.
- In order to run a class, the class must contain a method named main.
- The program is executed from the main method.

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Statement

- A statement represents an action or a sequence of actions.
- The statement System.out.println("Welcome to Java!")
- in the program is a statement to display the greeting "Welcome to Java!“.

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Statement Terminator

- Every statement in Java ends with a semicolon (;).

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Reserved Words

- Words that have a specific meaning to the compiler
- Cannot be used for other purposes in the program.
- For example the word class, it understands that the word after class is the name for the class.

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Blocks

- A pair of braces in a program forms a block that groups components of a program.

```
public class Test {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Class block

Method block

Special Symbols

Character	Name	Description
{}	Opening and closing braces	Denotes a block to enclose statements.
()	Opening and closing parentheses	Used with methods.
[]	Opening and closing brackets	Denotes an array.
//	Double slashes	Precedes a comment line.
" "	Opening and closing quotation marks	Enclosing a string (i.e., sequence of characters).
;	Semicolon	Marks the end of a statement.

Special Symbols

- {…} , (…) , […] , // , “…” , ;

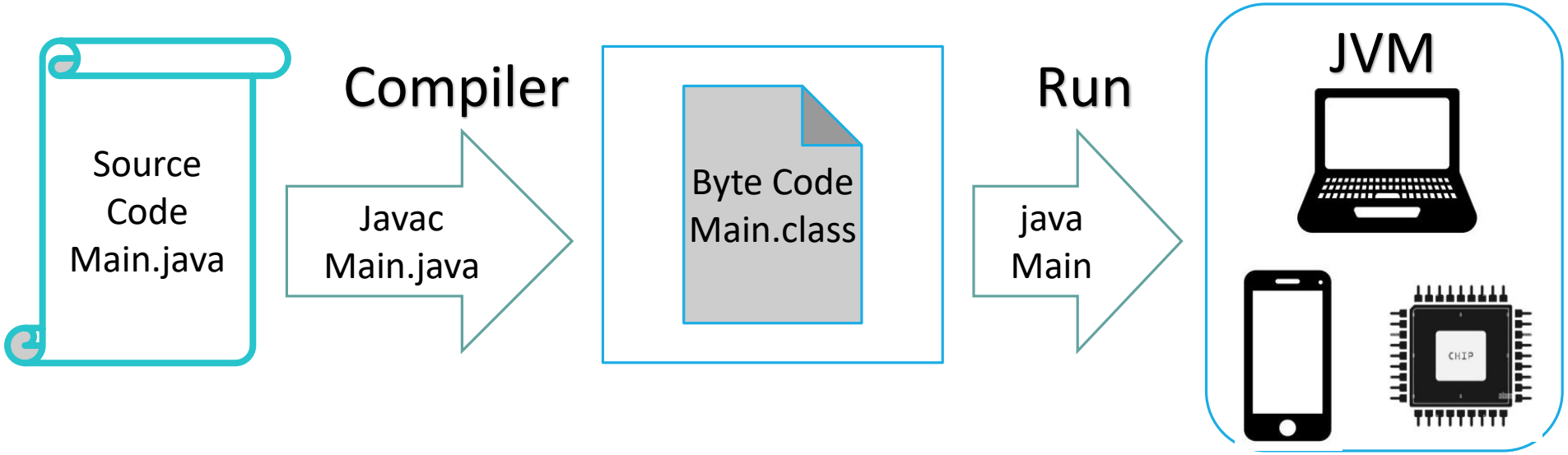
```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Compiling And Running Java Program

JVM, JRE, dan JDK

- Untuk dapat menjalankan sebuah program Java, sebuah mesin harus memiliki Java Virtual Machine (JVM).
- Java Runtime Environment (JRE) hanya mengandung komponen-komponen penting yang diperlukan untuk menjalankan program Java, seperti JVM dan Java standard libraries.
- Java Development Kit (JDK) mengandung fungsionalitas untuk menjalankan sekaligus membangun program Java.

Compile and Run Java Program



- Java is both compiled and interpreted.
- Source code Java di-compile menjadi byte code.
- Byte lebih cepat diinterpretasikan.
- Byte code dapat dijalankan di mesin yang lain.
- JVM → interpreter yang menjalankan byte code ke bahasa mesin.

Java in Action

```
public class test {
        //amazing code
}
```

Source code

compiler

Jalankan compiler

```
0x4556449
0x3334455
....
```

Java Virtual Machine (JVM)

Compiling

- Write your program in a text editor (e.g. notepad, notepad++)
- Save file .java, for example: **`Main.java`**
  - Carefull class name case sensitive!
- Open command prompt
- Compile Java source code
  - **`javac Main.java`**

Running

- Program will run “on” Java Virtual Machine (JVM)
- Execute byte codes
  - `java Main`
- Done!

Java Progamming Convention

Writing a Java Program

We could write something like this, but..

1

```
public class Main { public static void
main(String[] args) {
System.out.println("Hello world") ;
}}
```

2

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello world") ;
    }
}
```

This one reads better <_'

3

```
public class Main
{
    public static void main(String[] args)
    {
        System.out.println("Hello world") ;
    }
}
```

Programming Style and Documentation

- Appropriate Comments
- Naming Conventions
- Proper Indentation and Spacing Lines
- Block Styles

Appropriate Comments

- Include a summary at the beginning of the program to explain what the program does
  - its key features,
  - its supporting data structures, and
  - any unique techniques it uses.
- Include your name, class section, instructor, date, and a brief description at the beginning of the program.
- Three types of Java comments:
  - Single line comment: // …..
  - Multi-line comment: /* ….. */
  - Javadoc comment: /** ..... */

Naming Conventions

- Choose meaningful and descriptive names.
- Class names:
  - Capitalize the first letter of each word in the name. For example,

    the class name `ComputeExpression`.

Proper Indentation and Spacing

- Indentation
  - Indent two spaces.
- Spacing
  - Use blank line to separate segments of the code.

Proper Indentation and Spacing

- Use end-of-line style for braces.

Next-line style

```
public class Test
{
  public static void main(String[] args)
  {
    System.out.println("Block Styles");
  }
}
```

End-of-line style

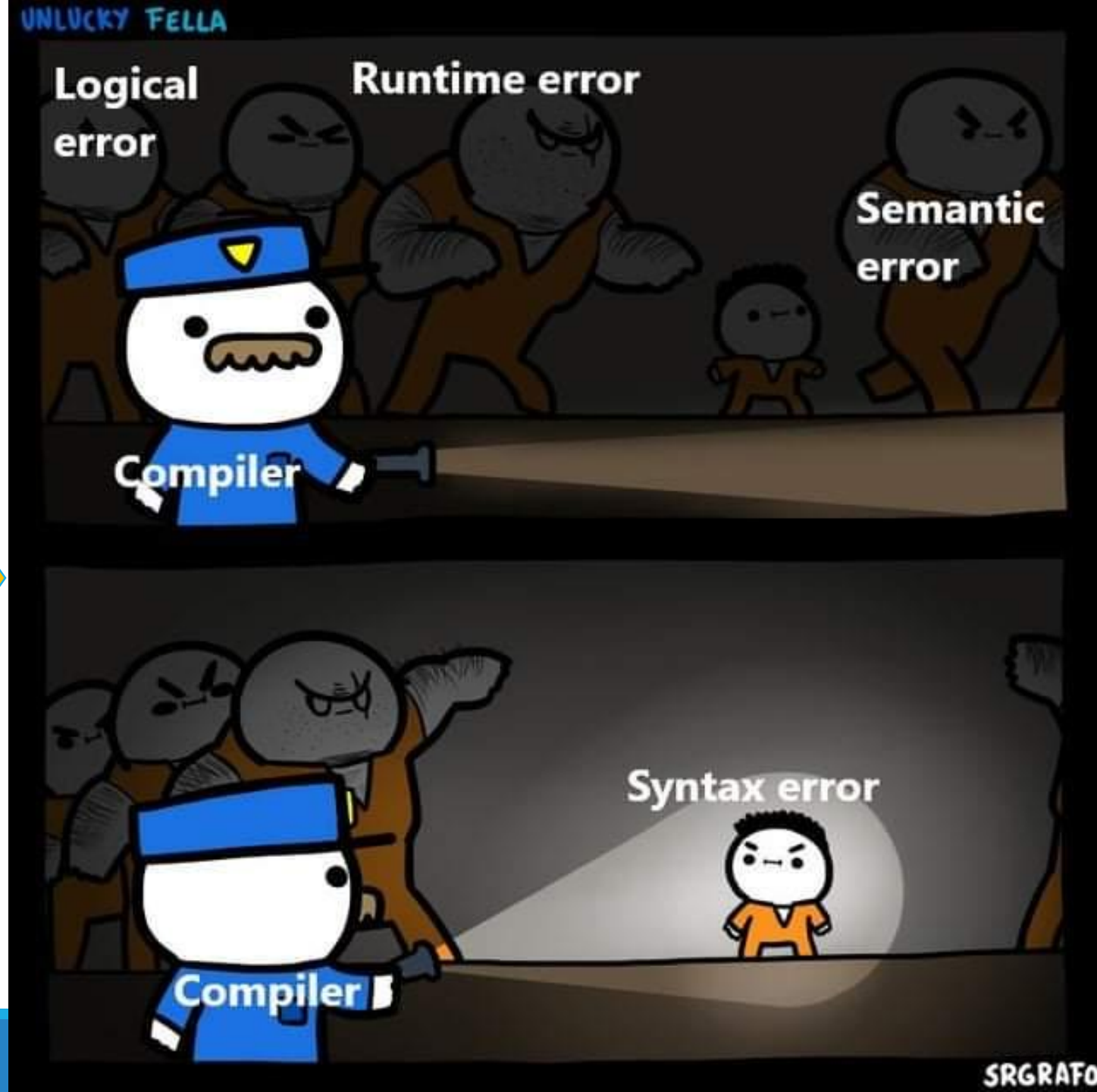
```
public class Test {
  public static void main(String[] args) {
    System.out.println("Block Styles");
  }
}
```

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

Java™

No Error
No Code
☺



Tipe Error

- Syntax Error
- Runtime Error
- Logic Error

Syntax Error

Dikenal juga dengan istilah compile-time errors

```
public class Main {
   int class = 10;
   public static void main(String[] args){
      System.out.println("Hello world") ;
   }
}
```

```
C:\Users\Dinial\Documents\java>javac Main.java
Main.java:2: error: <identifier> expected
        int class = 10;
            ^
Main.java:2: error: <identifier> expected
        int class = 10;
                  ^
2 errors
```

Runtime Error

```
public class Main {
    int class = 10/0;
    public static void main(String[] args){
        System.out.println("Hello world") ;
    }
}
```

Compile sukses

Error saat runtime

```
C:\Users\Dinial\Documents\java>javac Main.java

C:\Users\Dinial\Documents\java>java Main
Exception in thread "main" java.lang.Arithmetic
Exception: / by zero
        at Main.main(Main.java:3)
```

Logic Error

Bisa di compile
Dan di Run
Tapi hasil benar
Atau tidaknya
belum tau

```
public class Main {
    public static void main(String[] args){
        int Fahrenhet = 5/2*100;
        System.out.println("Nilai suhu =" +
Fahrenhet) ;
    }
}
```

```
C:\Users\Dinial\Documents\java>Javac Main.java

C:\Users\Dinial\Documents\java>Java Main
Nilai suhu =200
```

UNIVERSITAS INDONESIA

Veritas, Probitas, Iustitia

FAKULTAS ILMU KOMPUTER

To Be Continued ….