Introduction to java and elementary programming

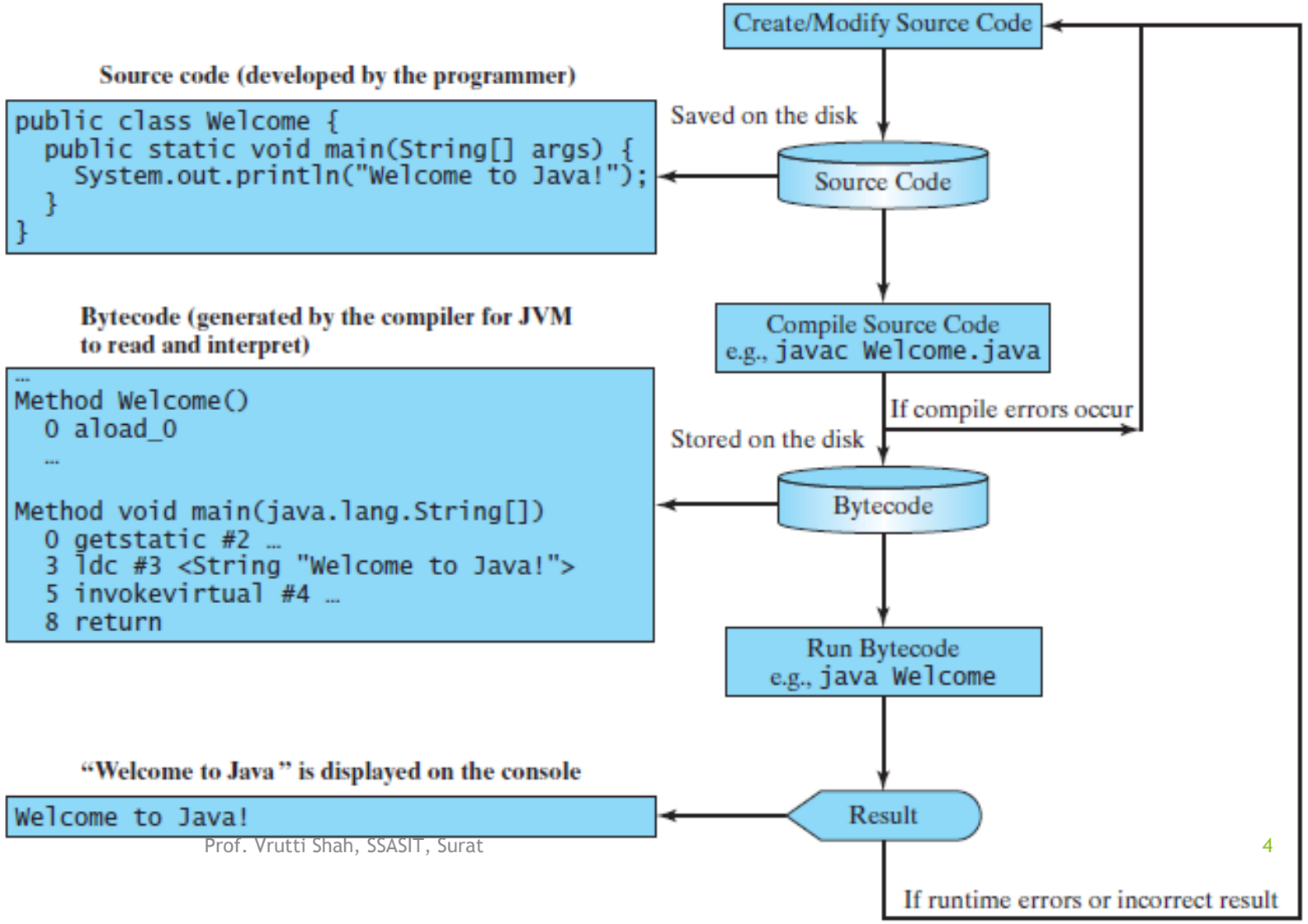
The Java Language Specification, API, JDK, and IDE

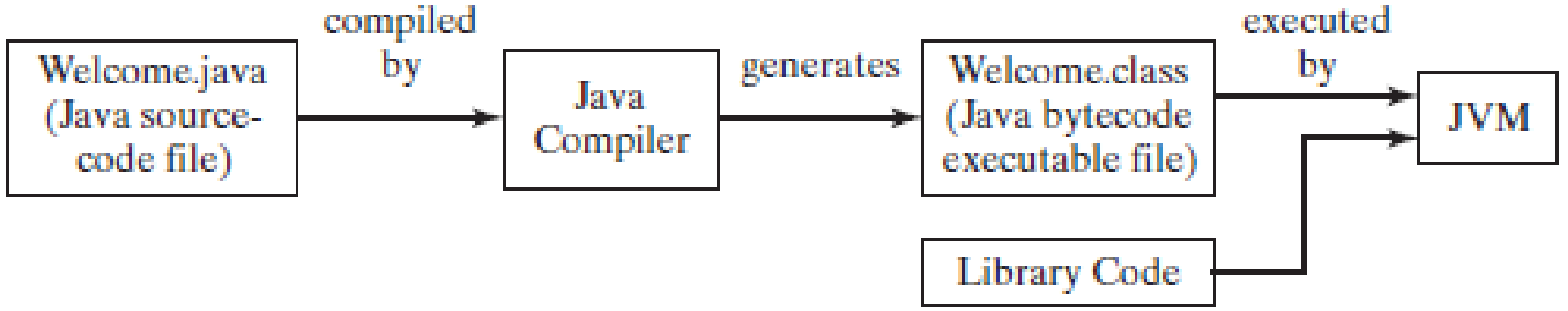
- What is Computer language?
- *Java language specification* is a technical definition of the Java programming language's syntax and semantics
- *API - application program interface* contains predefined classes and interfaces for developing Java programs.
- Java comes in 3 editions: Java SE, Java EE, Java ME
- JDK - *Java Development Toolkit*
  - consists of a set of separate programs for developing and testing Java programs
- IDE – *integrated development environment* for developing Java programs quickly
- Java development tool (e.g., NetBeans, Eclipse, and TextPad)—software that provides IDE, can be used instead of JDK

Sample Program : Welcome.java

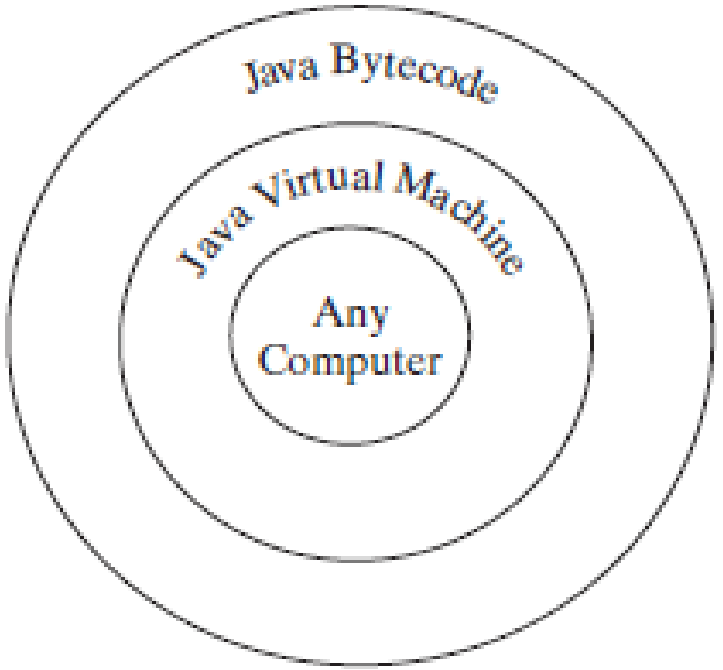
```
1 public class Welcome {
2     public static void main(String[] args) {
3     // Display message Welcome to Java! on the console
4     System.out.println("Welcome to Java!");
5     }
6 }
```

Creating, Compiling, and Executing a Java Program




Welcome.java
(Java source-code file)
compiled by
Java Compiler
generates
Welcome.class
(Java bytecode executable file)
executed by
JVM
Library Code

(a)


Java Bytecode
Java Virtual Machine
Any Computer

(b)

- Java is a high-level language, but Java bytecode is a low-level language
- *Bytecode*
  - *similar to machine instructions*
  - *Can run on any platform that has a Java Virtual Machine (JVM)*
- virtual machine is a program that interprets Java bytecode
- You can execute the bytecode on any platform with a JVM (interpreter)

Programming Style and Documentation

- *make a program easy to read and help programmers prevent errors*
- *Programming style* deals with what programs look like
- *Documentation* is the body of explanatory remarks and comments pertaining to a program

Appropriate Comments and Comment Styles

- Include a summary at the beginning of the program that explains
  - what the program does
  - its key features
  - any unique techniques it uses
- In a long program include comments
  - introduce each major step
  - explain anything that is difficult to read
- line comments (beginning with //), block comments (beginning with /*)
- javadoc comments. javadoc comments begin with /** and end with */
  - Use for commenting on an entire class or an entire method

Proper Indentation and Spacing

- A consistent indentation style makes programs clear and easy to read, debug, and maintain
- *Indentation* is used to illustrate the structural relationships between a program's components or statements
  - System.out.println(3+4*4); Bad Style
  - System.out.println(**3** + **4** * **4**); Good Style

Block Styles

- block is a group of statements surrounded by braces

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

Next-line style

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

End-of-line style

Programming Errors

- categorized into three types:
  - syntax errors
  - Runtime errors
  - logic errors
- Syntax Errors
  - Errors that are detected by the compiler are called *syntax errors* or *compile errors*

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

- Runtime Errors
  - *Runtime errors* are errors that cause a program to terminate abnormally
  - Input mistakes typically cause runtime errors

```
public class ShowRuntimeErrors {
  public static void main(String[] args) {
    System.out.println(1 / 0);
  }
}
```

- Logic Errors
  - *Logic errors* occur when a program does not perform the way it was intended to

```
public class ShowLogicErrors {
  public static void main(String[] args) {
    System.out.println("Celsius 35 is Fahrenheit degree ");
    System.out.println((9 / 5) * 35 + 32);
  }
}
```

- Common Errors
  - Missing a closing brace
  - missing a semicolon
  - missing quotation marks for strings
  - misspelling names

Writing a Simple Program

- computing the area of a circle
- Algorithm:
  - 1. Read in the circle's radius.
  - 2. Compute the area using the following formula:
    - *area* = *radius* * *radius* * p
  - 3. Display the result