Lecture 2
Introduction to Java

Course: Object Oriented Programming (CS F213)

PROF. ANITA AGRAWAL
BITS PILANI, K.K. BIRLA GOA CAMPUS

In today’s session…

INTRODUCTION TO JAVA

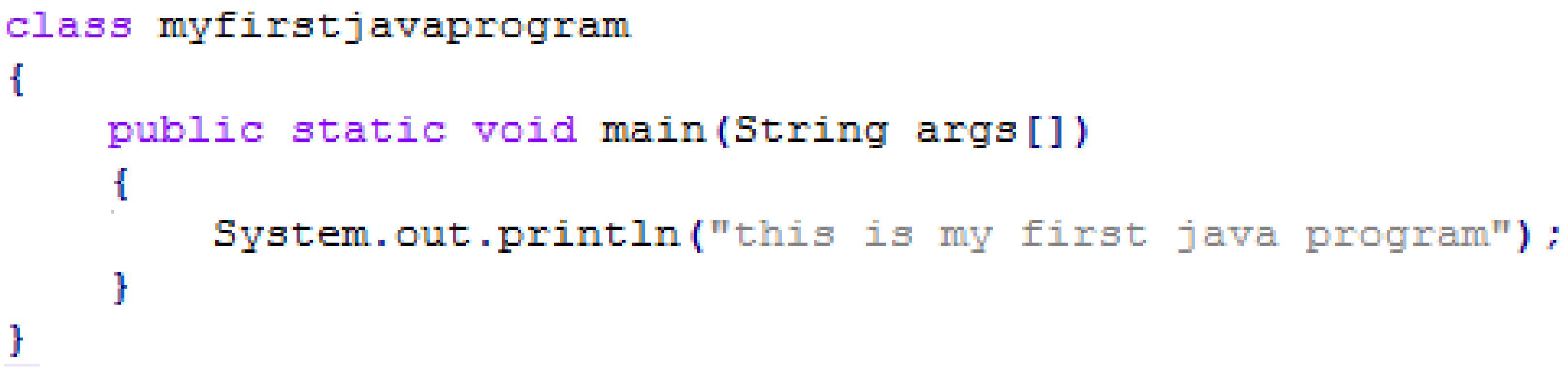
Java Program

- A Java program is defined as a collection of objects that communicate via invoking each other’s methods.

Anatomy of A Java Program

A. Reserved words / Keywords
B. Classes
C. Package
D. Modifiers
E. Statements
F. Blocks
G. Methods
H. The main method
I. Comments

My first program



```
class myfirstjavaprogram
{
    public static void main(String args[])
    {
        System.out.println("this is my first java program");
    }
}
```

Breaking down the program

- “class" keyword is used to declare a new class.
- “Myfirstjavaprogram" is the identifier which is the name of the class.
- The program begins executing at public static void main
- public:
  - It is an example of an access specifier.
  - Allows programmer to control the visibility of class members
  - Members may be accessed from outside the class in which they are declared.

- main() be declared as public since it will be called by outside it’s class

Breaking down the program (Contd.)

- **static:-**
  - Allows main() to be called without creating an object of the class.
  - It is necessary since main() is called by JVM before any objects are made
- **void:-**
  - Defines the return type, in this case it is void

- **main:-**

  - The system locates and runs the main method for a class when you run a program.
  - Other methods get executed when called by the main method.

- **String args[ ]:**

–Declares a variable (object) name **args**

–It is an array of String class instances

–args[ ] receives any command line arguments during program execution

System.out.println

- It is a *method*: a collection of statements that performs a sequence of operations to display a message on the console.

- It is used by invoking a statement with a string argument.

- The string argument is enclosed within parentheses. In this case, the argument is "This is my first java program!"

- You can call the same println method with a different argument to print a different message.

More specifically….

System.out.println (“This is my first Java program”);

–This line displays the string on screen followed by a new line

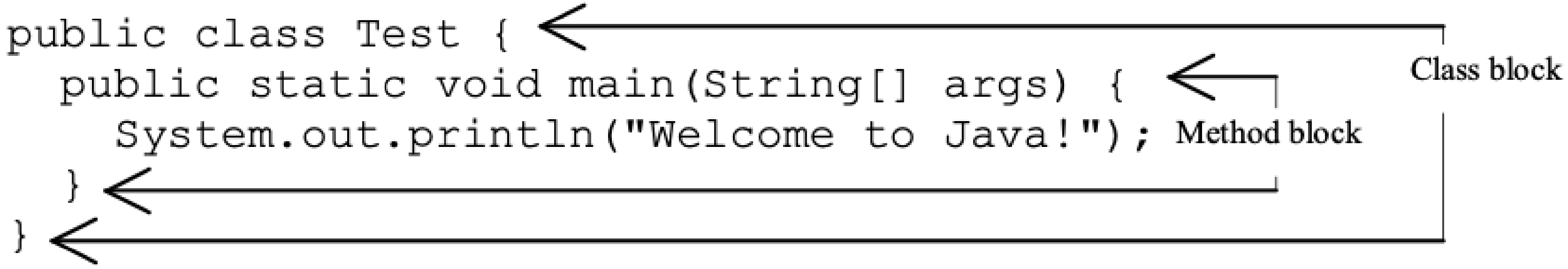
–println () is a built-in method

–System is a pre-defined class that provides access to the system

–out is the output stream

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!");
  }
}
```

Comments

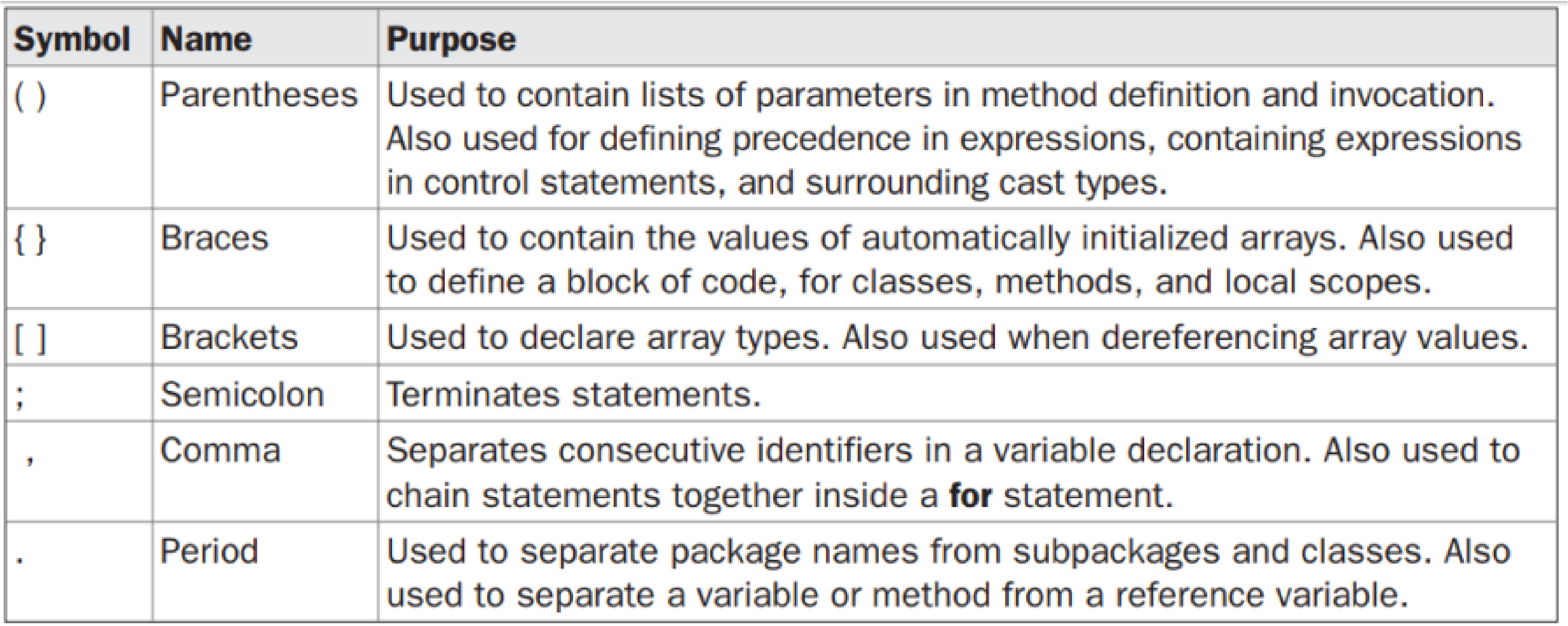
```
//my first java program
class myfirstjavaprogram
{
    public static void main(String args[])
      {

            System.out.println("this is my first java program" );

       }
}
```

```
// This is a single line comment
/* These are multiple
        line comments */
```

Lexical constraints

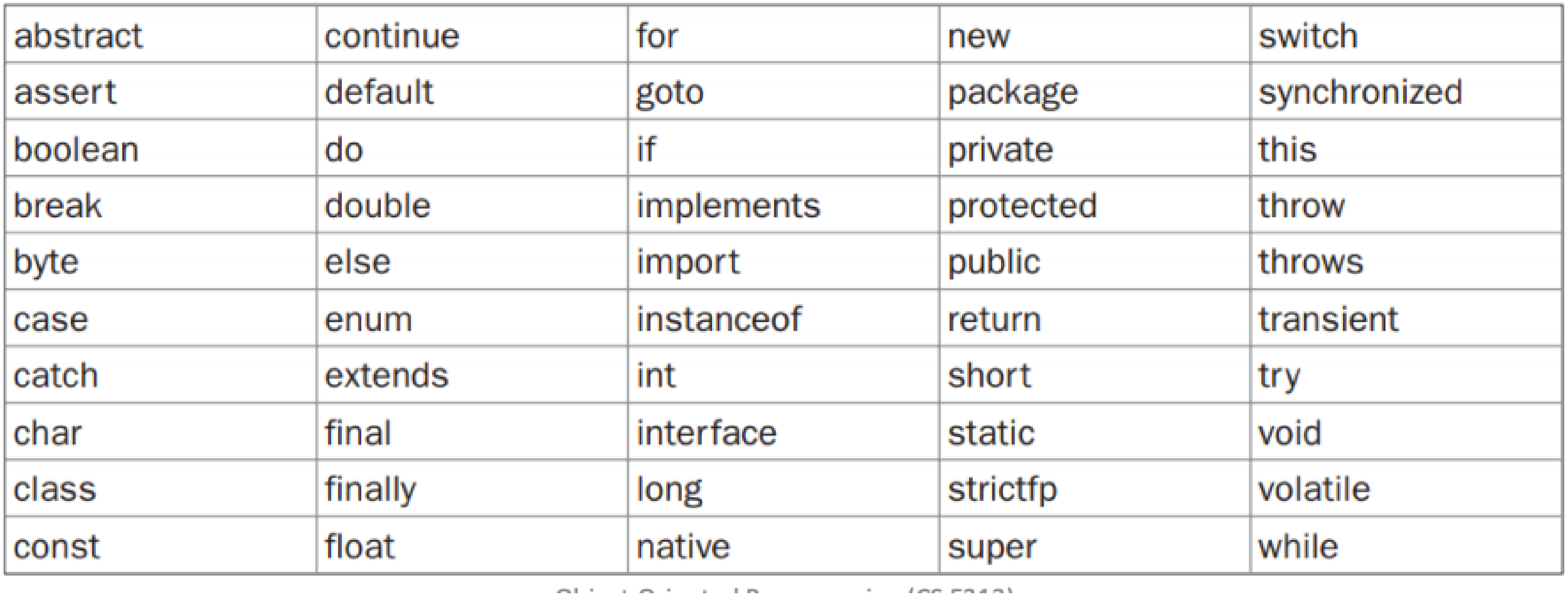
- Whitespace – Java is a free-form language (indentation is not necessary but desirable).
  - As long as there is one whitespace character between each token – Space, tab, and newline.
- Identifiers – Used for class, method, and variable name.
  - Combination of upper and lowercase letters, and numbers – Underscore_ and $ sign are allowed
- Literals – A constant value in Java is called literal
- Comments – /* */ and //

List of Separators



Symbol	Name	Purpose
( )	Parentheses	Used to contain lists of parameters in method definition and invocation. Also used for defining precedence in expressions, containing expressions in control statements, and surrounding cast types.
{ }	Braces	Used to contain the values of automatically initialized arrays. Also used to define a block of code, for classes, methods, and local scopes.
[ ]	Brackets	Used to declare array types. Also used when dereferencing array values.
;	Semicolon	Terminates statements.
,	Comma	Separates consecutive identifiers in a variable declaration. Also used to chain statements together inside a **for** statement.
.	Period	Used to separate package names from subpackages and classes. Also used to separate a variable or method from a reference variable.

Reserved words/ Keywords in Java



abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

Important points

- The name of the source file is very important.
- In previous example, the name of the file will be **myfirstjavaprogram.java**
- The name of the class should match the name of file that holds the program.
- A source file is officially called **compilation unit.**

- It is a text file that contains one or more class definitions.

- For Java compiler, source file must have **.java** extension.

- In Java, all code must reside inside a class

- Java is a **case-sensitive** programming language.

Example 2: If control statement

```
//if control statement

//if statement
class if_eg
{
        public static void main(String args[])
        {
                int n=10;
                if(n%2== 0)
                        System.out.println("The number is even");
                else
                        System.out.println("The number is odd");
        }
}
```

History of Java

- Initially called ‘Oak’, but after a while (1995) came to be known as ‘Java’, over coffee.

- **J**ames Gosling, **A**rthur **V**an Hoff and **A**ndy Bechtolsheim

Salient Features of Java

- ❑ Object-oriented programming language
- ❑ Structured in terms of ***classes***, which group data with operations on that data
- ❑ Can construct new classes by ***extending*** existing ones

Salient features of Java

- **Platform Independent:**

Unlike many other programming languages including C and C++, when compiled, it is not compiled into a code which is platform specific, rather into "**platform-independent byte code**".

- It runs on a variety of platforms such as Windows, Mac OS, and the various versions of unix

Salient Features of Java

- **Simple:**

Easy to learn

- **Secure:**

Enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption

Salient Features of Java

- **Architecture-neutral:**

Generates an architecture-neutral object file format which makes the compiled code executable on many processors with the presence of Java runtime system.

- **Portable:**

Being architecture neutral and having no implementation dependent aspects of the specification makes it portable

Salient Features of Java

- **Robust:**

Eliminates error-prone situations by emphasizing mainly on compile time error checking and runtime checking

- **Multithreaded:**

Programs can be written that can perform many tasks simultaneously

- **Interpreted:**

Java code is translated on the fly to native machine instructions and not stored anywhere.

Salient Features of Java

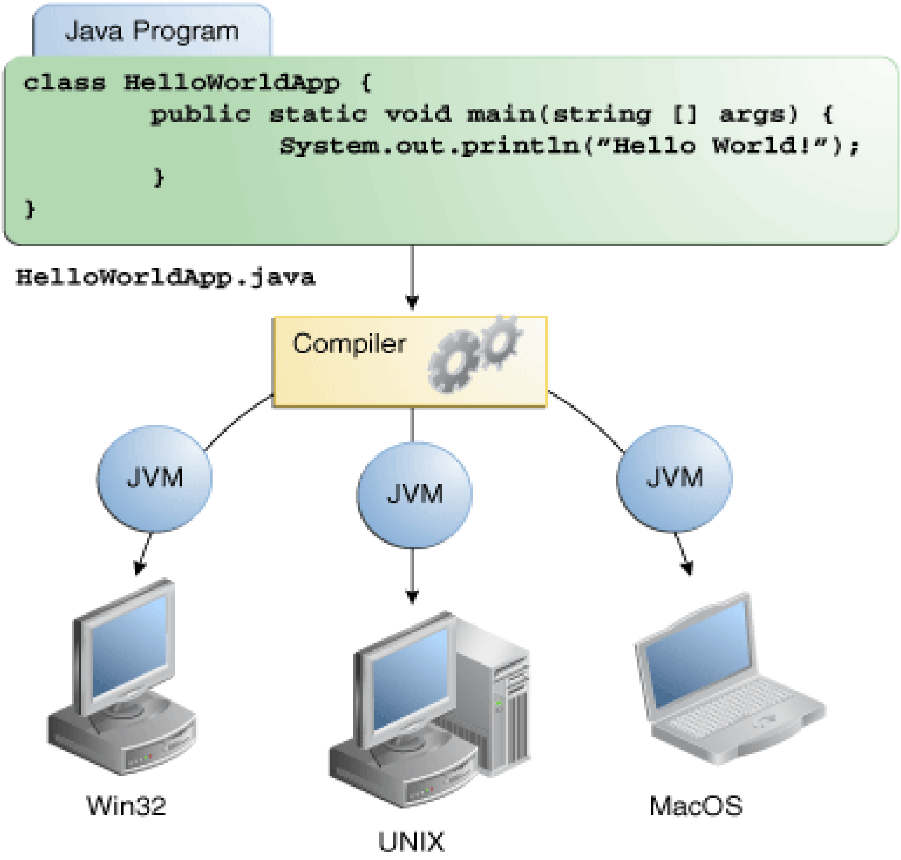
- **High Performance:**

With the use of Just-In-Time compilers, Java enables high performance

- **Dynamic:** More dynamic than C or C++
  - **Can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.**

ByteCode

- Output of Java Compiler is not executable code but is a **Bytecode**.
- Bytecode is a highly optimized set of instructions to be executed by Java Runtime System (Java Virtual Machine- JVM).
- JVM is the interpreter for Bytecode -> Convert into machine code for execution.

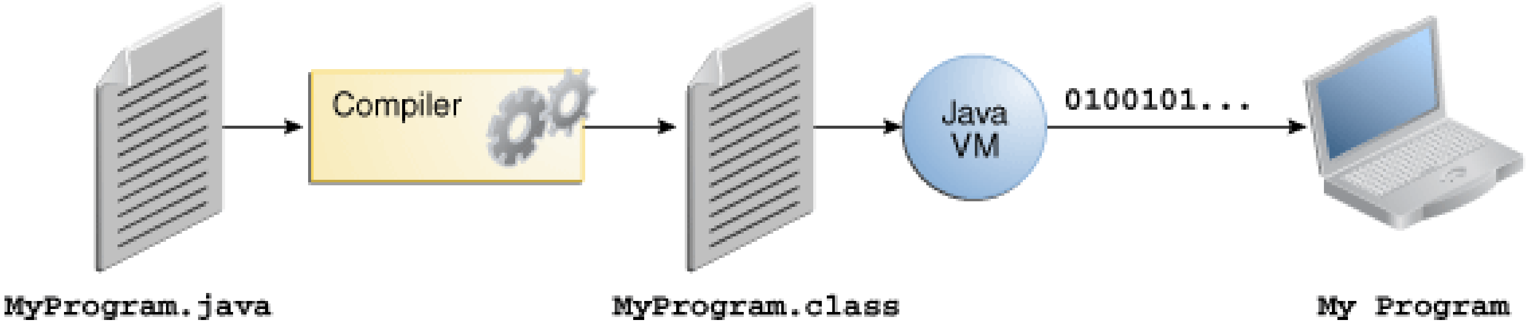


Java Program

```
class HelloWorldApp {
        public static void main(string [] args) {
                System.out.println(”Hello World!”);
        }
}
```

HelloWorldApp.java

Java program: Stages of execution



Bytecode- Contd.

- Advantages:
  - It makes it easier to execute a program
  - Only the JVM needs to be implemented
- Disadvantages:
  - Slower performance

Requirements

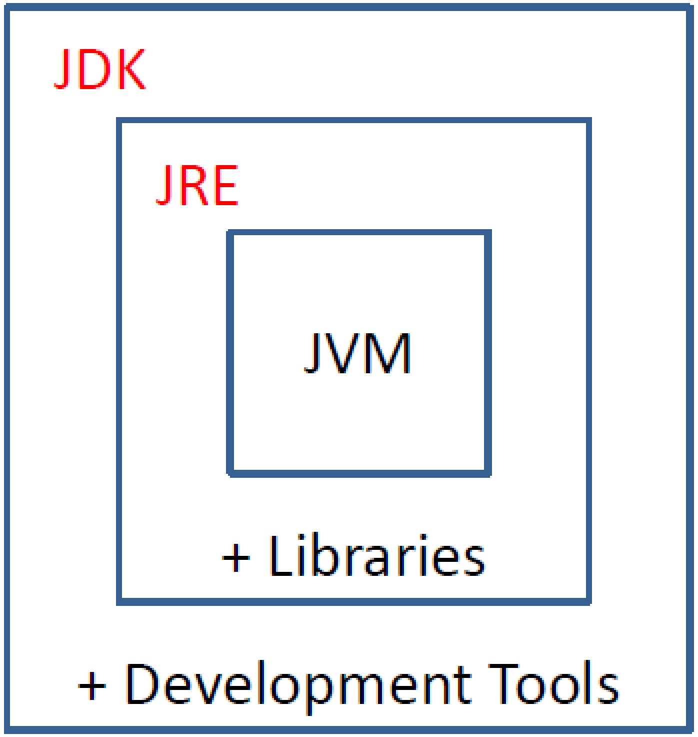
- A text editor
- The Java Development Kit (JDK)
  - Includes the Java Compiler
- The Java Runtime Environment (JRE)
  - Includes Java Virtual Machine (JVM)

JVM vs. JRE vs. JDK

- JVM – Java Virtual machine(JVM) – JVM is responsible for executing the java bytecode line by line hence it is also known as interpreter – JVM allows Java to be a "portable language" (write once, run anywhere)

- JRE – Java Runtime Environment – An installation package which provides environment to only run (not develop) the java program onto your machine. JRE includes JVM and libraries – JRE is only used by them who only wants to run the Java Programs i.e. end users of your system.

- JDK – Java Development Kit – The JDK is a superset of the JRE, and contains everything that is in the JRE, plus tools such as the compilers and debuggers necessary for developing programs and applications.



JDK

JRE

JVM

+ Libraries

+ Development Tools

Compiling a Java Program

- Execute the java compiler (**javac**)
  - –C:\>javac myfirstjavaprogram.java
- Javac compiler will create myfirstjavaprogram**.class** file
- .class file contains the **bytecode** version of the program
- Java bytecode is not an executable code
- To execute a bytecode, we need JVM

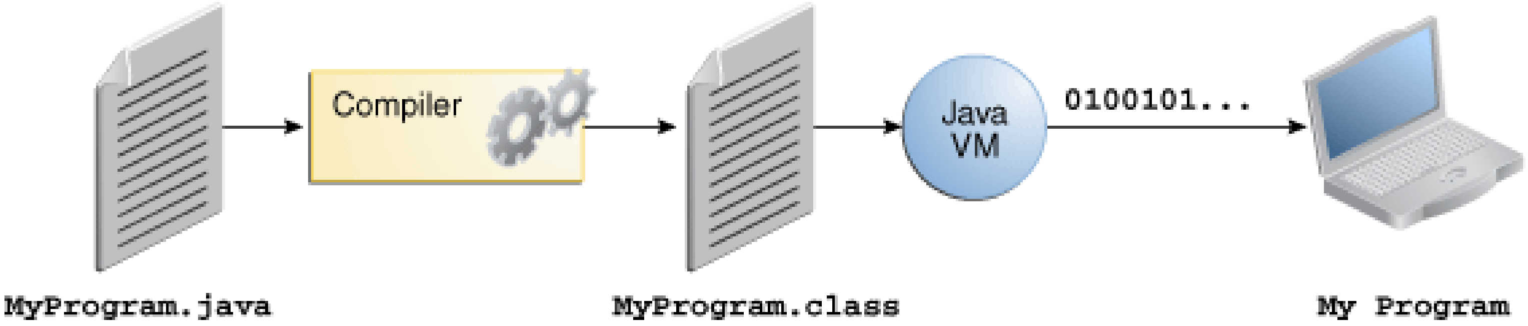
Executing a Java Program

- To run the program, we need Java Application Launcher called **java**
  - –C:\>java myfirstjavaprogram (not myfirst…   .java)

- When the program is run, the following output will be displayed:

```
This is my first Java program
```

Java program: Stages of execution



JDK editions

A. Java Standard Edition (J2SE)

J2SE can be used to develop client-side standalone applications or applets.

B. Java Enterprise Edition (J2EE)

J2EE can be used to develop server-side applications such as Java servlets and Java ServerPages.

C. Java Micro Edition (J2ME).

J2ME can be used to develop applications for mobile devices such as cell phones.

Next: ………….Data Types and Variables