

# NGÔN NGỮ LẬP TRÌNH JAVA

Giảng viên: TS. Vũ Hữu Tiến

Email: [tienvh@ptit.edu.vn](mailto:tienvh@ptit.edu.vn)

Mobile: 0939919396

# GIỚI THIỆU MÔN HỌC

---

- Mục đích môn học:
  - Cung cấp kiến thức nền tảng về ngôn ngữ lập trình Java.
  - Làm tiền đề cho các môn học: Lập trình Game, Lập trình ứng dụng trên di động, Lập trình mạng, Phát triển bài giảng e\_learning, Lập trình kỹ xảo đpt (tạo các plug-in trên Maya, After effect,...).

# GIỚI THIỆU MÔN HỌC

---

- Nội dung môn học:
  - **Chương 1:** Tổng quan về lập trình hướng đối tượng và ngôn ngữ lập trình Java
  - **Chương 2:** Lớp và đối tượng trong Java
  - **Chương 3:** Tính kế thừa và đa hình trong Java
  - **Chương 4:** Xử lý nhập/ xuất trong Java
  - **Chương 5:** Xử lý ngoại lệ trong Java
  - **Chương 6:** Lập trình đa luồng trong Java
  - **Chương 7:** Lập trình giao diện trong Java

# GIỚI THIỆU MÔN HỌC

---

- Học liệu:
  - Kathy Sierra , Bert Bates. *Head First Java*. O'reilly Media, Inc, 2005.
  - Lewis & Loftus. *Java software solutions foundation of program design*. Pearson, 5<sup>th</sup> Edition
  - Phạm Văn Thiều, Nguyễn Quang Thanh, Hà Thị Thanh Tâm, *Java cho sinh viên*, NXB Thống kê.

# GIỚI THIỆU MÔN HỌC

---

- Quy định của môn học:
  - Điểm chuyên cần: **10%**
  - Điểm kiểm tra giữa kỳ: **10%**
  - Điểm bài tập lớn: **20%**
  - Điểm thi cuối kỳ: **60%**

# TỔNG QUAN VỀ JAVA

---

- Lịch sử của Java:

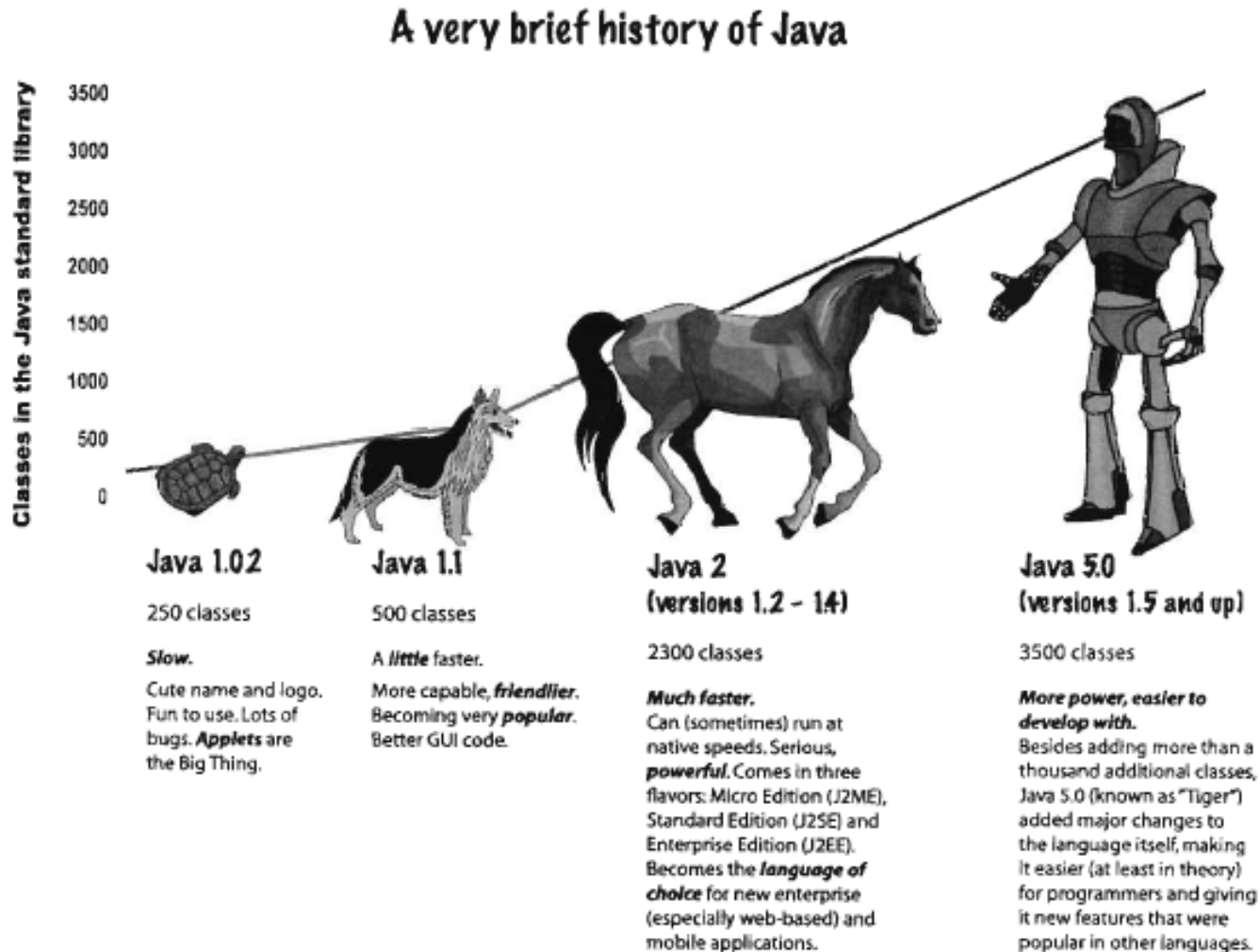
- James Gosling, Mike Sheridan và Patrick Naughton khởi xướng dự án về ngôn ngữ Java trong tháng 6/1991. Họ được gọi là “*Green Team*”.
- Sau đó, nó được gọi là “Oak” và được phát triển như là một phần của Green Project.
- Năm 1995, Oak được đổi tên thành Java.
- JDK 1.0 được công bố vào 23/1/1996.

James Gosling was a graduate of the U of C Computer Science program.



# TỔNG QUAN VỀ JAVA

- Lịch sử của Java:



# TỔNG QUAN VỀ JAVA

---

- Java là gì?
  - Java là một ngôn ngữ lập trình và là một Platform
    - **Ngôn ngữ lập trình:** Java là một ngôn ngữ lập trình có tính bảo mật cao, hướng đối tượng, bậc cao và mạnh mẽ.
    - **Platform:** Bất cứ môi trường phần cứng hoặc phần mềm nào mà trong đó một chương trình chạy, thì được biết đến như là một Platform. Với môi trường runtime riêng cho mình là JRE (Java Runtime Environment) và API (Application Programming Interface), Java được gọi là Platform.
  - Có rất nhiều thiết bị hiện tại đang sử dụng Java. Bao gồm:
    - Desktop App như media player, antivirus, reader, ...
    - Web App như [irctc.co.in](http://irctc.co.in), [javatpoint.com](http://javatpoint.com), ...
    - Enterprise App như các ứng dụng về xử lý nghiệp vụ ngân hàng, ...
    - Trên các thiết bị Mobile.



# TỔNG QUAN VỀ JAVA

---

- Phân cấp ngôn ngữ lập trình
  - Ngôn ngữ máy (machine language)
  - Hợp ngữ (assembly language)
  - Ngôn ngữ bậc cao (high-level languages)
  - Ngôn ngữ thế hệ 4 (fourth-generation language)

High-Level Language	Assembly Language	Machine Language
a + b	ld [%fp-20], %o0 ld [%fp-24], %o1 add %o0, %o1, %o0	... 1101 0000 0000 0111 1011 1111 1110 1000 1101 0010 0000 0111 1011 1111 1110 1000 1001 0000 0000 0000 ...

**figure 1.20** A high-level expression and its assembly language and machine language equivalent

# TỔNG QUAN VỀ JAVA

---

- Java: Write Once, Run Anywhere

## Bước 1: Biên dịch (Compilation)

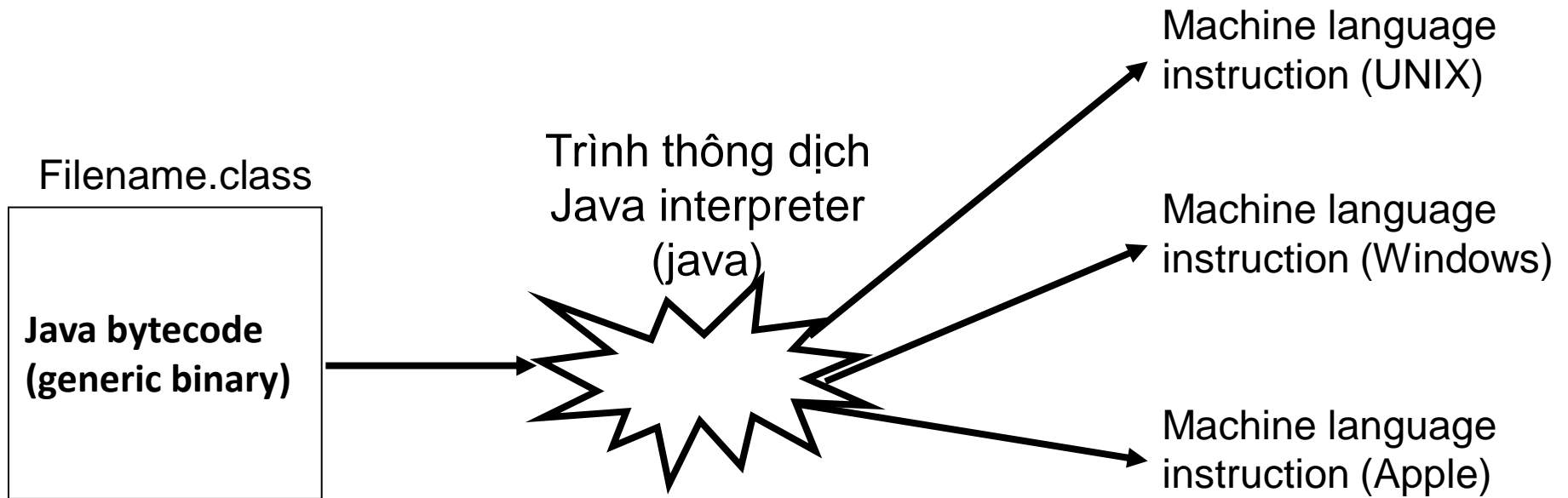


# TỔNG QUAN VỀ JAVA

---

- Java: Write Once, Run Anywhere

**Bước 2: Thông dịch (Interpreting) và thực thi (executing) mã nguồn**



# TỔNG QUAN VỀ JAVA

- Các bước thực hiện trong quá trình chạy một file Java

Smallest.java

```
public class Smallest
{
    public static void main (String[] args)
    {
    }
}
```

**javac**

Type "javac  
Smallest.java"

Smallest.class

(Java byte code)

```
100001000000001000
001001000000001001
```

:

:

# TỔNG QUAN VỀ JAVA

- Các bước thực hiện trong quá trình chạy một file Java

Smallest.class

(Java byte code)

```
100001000000001000
001001000000001001
:           :
```



java

Type "java Smallest"

(Platform/Operating specific binary)

```
101001110000001000
00100111001111001
:           :
```

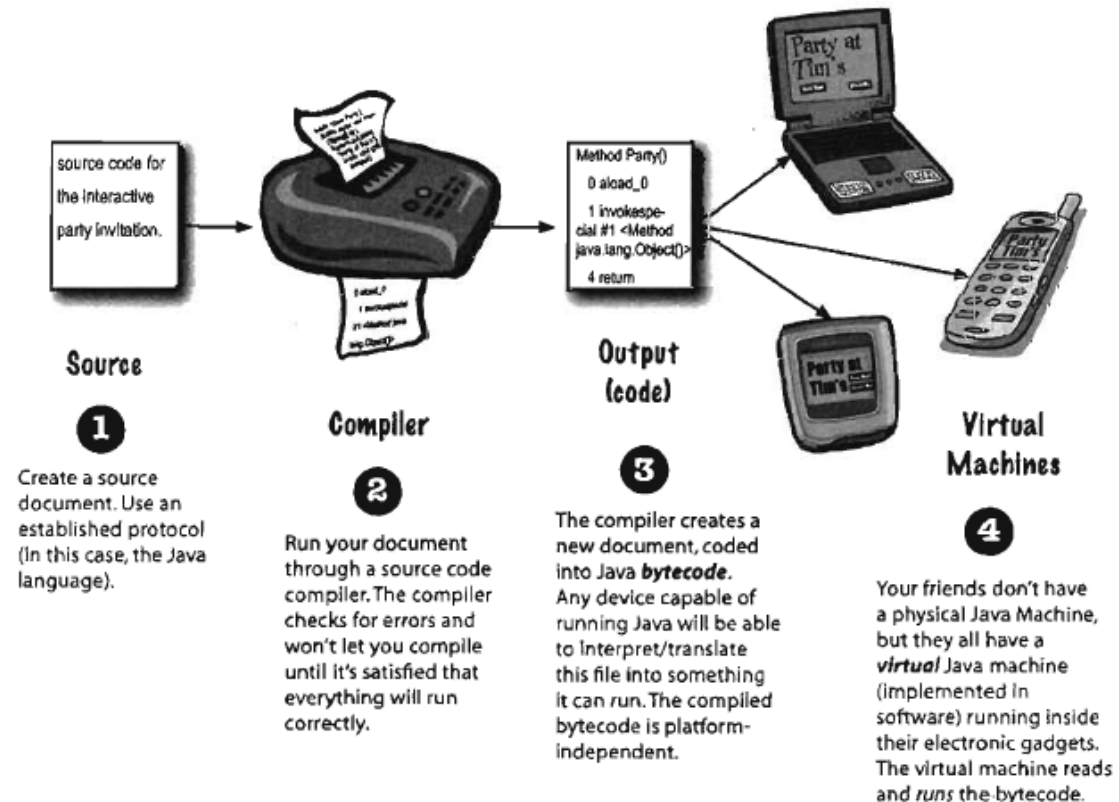


# TỔNG QUAN VỀ JAVA

- Các bước thực hiện trong quá trình chạy một file Java

## The Way Java Works

The goal is to write one application (in this example, an interactive party invitation) and have it work on whatever device your friends have.



# TỔNG QUAN VỀ JAVA

---

- JDK (Java Development Kit)
  - Java 6+ JDK (Java Development Kit), Standard Edition includes:
    - JDK (Java development kit) – for *developing* Java software (creating Java programs).
    - JRE (Java Runtime environment) – only good for *running* pre-created Java programs.
    - Java Plug-in – a special version of the JRE designed to run through web browsers.

# TỔNG QUAN VỀ JAVA

---

- JDK version
  - JDK 1.02 (1995)
  - JDK 1.1 (1996)
  - Java 2 SDK v1.2 (a.k.a JDK 1.2, 1998)
  - Java 2 SDK v1.3 (a.k.a JDK 1.3, 2000)
  - Java 2 SDK v1.4 (a.k.a JDK 1.4, 2002)



# TỔNG QUAN VỀ JAVA

---

- JDK edition
  - Java Standard Edition (J2SE)
    - J2SE can be used to develop client-side standalone applications or applets.
  - Java Enterprise Edition (J2EE)
    - J2EE can be used to develop server-side applications such as Java servlets and Java ServerPages.
  - Java Micro Edition (J2ME).
    - J2ME can be used to develop applications for mobile devices such as cell phones.

# TỔNG QUAN VỀ JAVA

---

- Một số công cụ cần cho việc lập trình Java
  - JDK  
(<http://www.oracle.com/technetwork/java/javase/downloads/index.html>)
  - Công cụ soạn thảo
    - Eclipse (<http://www.eclipse.org/downloads/>)
    - NetBeans (<https://netbeans.org/>)
    - Jcreator (<http://www.jcreator.org/download.htm>)

# TỔNG QUAN VỀ JAVA

---

- Các ứng dụng được phát triển bởi Java:
  - Un-network app: (1) Standalone Java program
  - Network app: non-standalone Java program
    - Internet:
      - (2) *Applet*
      - (3) *servlet*
      - (4) *JavaBean classes*
    - Intranet:
      - (5) *EJB* ( EnterpriseJavaBean )
      - (6) *RMI, etc*

# TỔNG QUAN VỀ JAVA

---

- Các ứng dụng được phát triển bởi Java:
  - Un-network app: (1) Standalone Java program

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

Public: là một Access Modifier cho phép tính nhìn thấy, nghĩa rằng nó là nhìn nhất với tất cả.

static : là một thuộc tính của phương thức main

void: kiểu trả về giá trị của phương thức

String----It always has an array of String objects as its formal parameter.  
the array contains any arguments passed to the program on the  
command line

the source file's name must match the class name which main method is in

# TỔNG QUAN VỀ JAVA

---

- Các ứng dụng được phát triển bởi Java:
  - Un-network app: (1) Standalone Java program

```
1 // Fig. 2.1: Welcome1.java
2 // A first program in Java
3
4 public class Welcome1 {
5     public static void main( String args[] )
6     {
7         System.out.println( "Welcome to Java Programming!" );
8     }
9 }
```

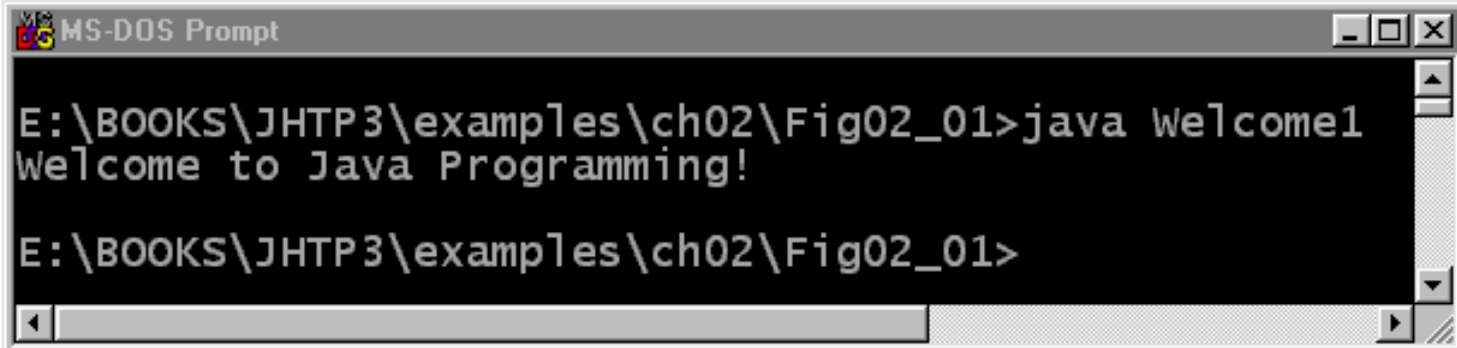
```
Welcome to Java Programming!
```

# TỔNG QUAN VỀ JAVA

---

- Các ứng dụng được phát triển bởi Java:
  - Un-network app: (1) Standalone Java program

```
1 // Fig. 2.1: Welcome1.java
2 // A first program in Java
3
4 public class Welcome1 {
5     public static void main( String args[] )
6     {
7         System.out.println( "Welcome to Java Programming!" );
8     }
9 }
```

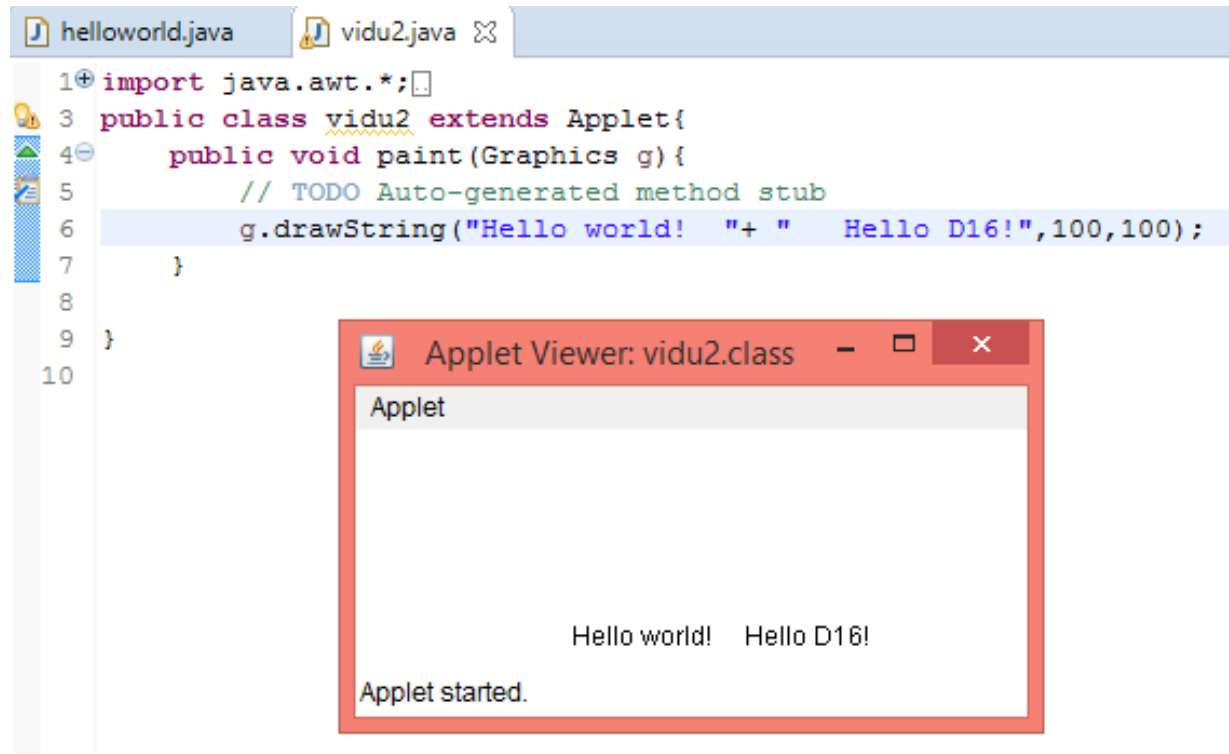


MS-DOS Prompt

```
E:\BOOKS\JHTP3\examples\ch02\Fig02_01>java Welcome1
Welcome to Java Programming!
E:\BOOKS\JHTP3\examples\ch02\Fig02_01>
```

# TỔNG QUAN VỀ JAVA

- Các ứng dụng được phát triển bởi Java:
  - Internet:
    - (2) *Applet*

A screenshot of a Java IDE. The top part shows a code editor with two tabs: 'helloworld.java' and 'vidu2.java'. The 'vidu2.java' tab is active, showing the following code:

```
1 import java.awt.*;
3 public class vidu2 extends Applet{
4     public void paint(Graphics g){
5         // TODO Auto-generated method stub
6         g.drawString("Hello world!  "+ "    Hello D16!",100,100);
7     }
8
9 }
10
```

The bottom part of the screenshot shows a window titled 'Applet Viewer: vidu2.class'. Inside the window, the text 'Hello world! Hello D16!' is displayed in the center, and 'Applet started.' is displayed at the bottom.

```
1 <html>
2 <applet code="vidu2.class" width=300 height=40>
3 </applet>
4 </html>
```

# TỔNG QUAN VỀ JAVA

---

- Lập trình hướng đối tượng (Object-oriented Programming)
  - *Đối tượng*
  - *Lớp*
  - *Thuộc tính*
  - *Phương thức*
  - *Đóng gói*
  - *Kế thừa*
  - *Đa hình*



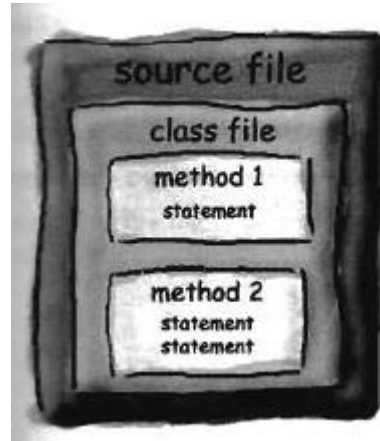
# TỔNG QUAN VỀ JAVA

---

- Lập trình hướng đối tượng (Object-oriented Programming)
  - *Đối tượng*: Là một thực thể có trạng thái và hành vi. Ví dụ như ô tô thể thao, ô tô địa hình, quả bóng bàn, quả bóng đá, ...
  - *Lớp*: Là một tập hợp các đối tượng. Ví dụ: Lớp ô tô, lớp bóng,...

# TỔNG QUAN VỀ JAVA

- Cấu trúc code của Java



## What goes in a **SOURCE** file?

A source code file (with the *.java* extension) holds one *class* definition. The class represents a *piece* of your program, although a very tiny application might need just a single class. The class must go within a pair of curly braces.

```
public class Dog {  
  
    }  
class
```

- Source file:

Source code (.java) chứa các định nghĩa về các lớp

Mỗi lớp bao gồm

*Public: Phạm vi truy cập*

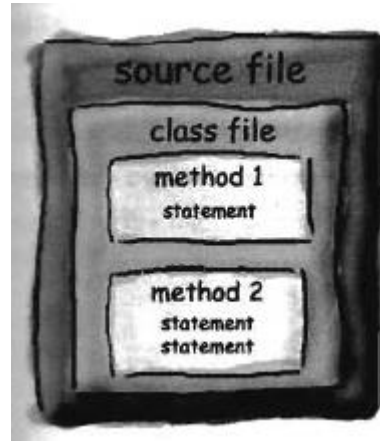
*Từ khóa: class*

*Tên lớp*

*Ngoặc {}*

# TỔNG QUAN VỀ JAVA

- Cấu trúc code của Java



## What goes in a class?

A class has one or more *methods*. In the Dog class, the *bark* method will hold instructions for how the Dog should bark. Your methods must be declared *inside* a class (in other words, within the curly braces of the class).

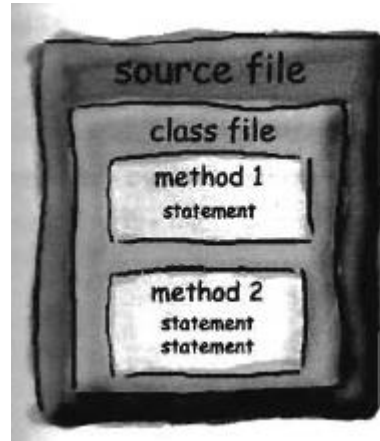
```
public class Dog {  
    void bark() {  
  
    }  
}
```

method

- Lớp:
  - Mỗi lớp có một hoặc nhiều phương thức (method).
  - Phương thức phải được khai báo trong phạm vi của một lớp.

# TỔNG QUAN VỀ JAVA

- Cấu trúc code của Java



## What goes in a method?

Within the curly braces of a method, write your instructions for how that method should be performed. Method *code* is basically a set of statements, and for now you can think of a method kind of like a function or procedure.

```
public class Dog {
    void bark() {
        statement1;
        statement2;
    }
}
```

**statements**

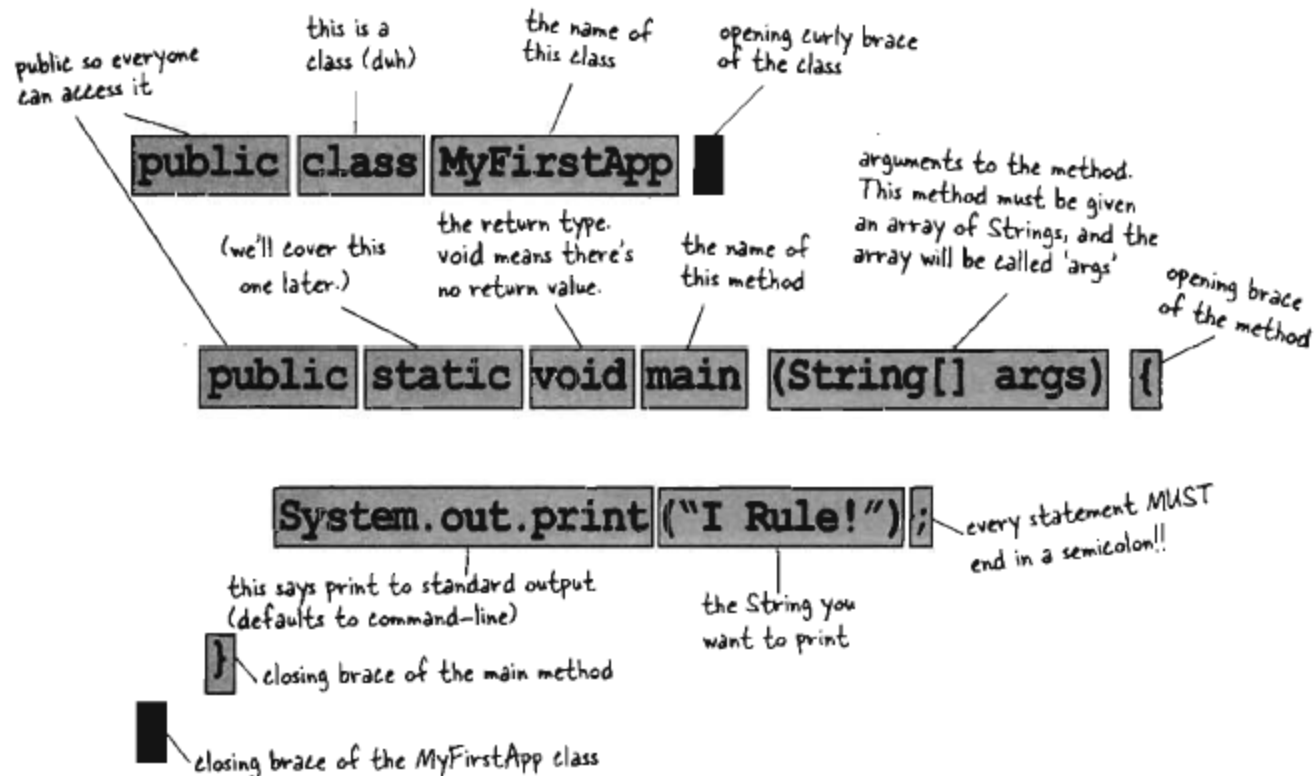
### – Phương thức:

- Mỗi phương thức bao gồm các câu lệnh để mô tả cách hoạt động của phương thức.
- Có thể coi mỗi phương thức như một hàm hoặc thủ tục.

# TỔNG QUAN VỀ JAVA

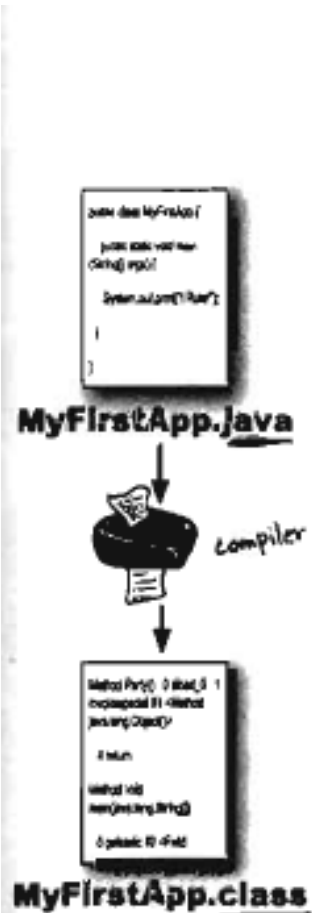
- Phân tích Class
  - Khi bắt đầu chạy một ứng dụng Java, JVM sẽ tìm đọc Class.
  - Sau đó, JVM đọc phương thức *main* và thực hiện các lệnh trong phương thức main.

Một ứng dụng có thể có nhiều Class nhưng chỉ có duy nhất một phương thức main.



# TỔNG QUAN VỀ JAVA

- Ví dụ:



```
public class MyFirstApp {

    public static void main (String[] args) {
        System.out.println("I Rule!");
        System.out.println("The World");
    }

}
```

## 1 Save

MyFirstApp.java

## 2 Compile

javac MyFirstApp.java

## 3 Run

```
File Edit Window Help Screen
% java MyFirstApp
I Rule!
The World
```

# TỔNG QUAN VỀ JAVA

---

- Anatomy of a Java program

- Comments
- Package
- Reserved words
- Modifiers
- Statements
- Blocks
- Classes
- Methods
- The main method

## Example

```
//This application program
//prints Welcome to Java!
package chapter1;

public class Welcome {
    public static void
    main(String[] args) {

        System.out.println("Welcome
        to Java!");
    }
}
```

# TỔNG QUAN VỀ JAVA

---

- Anatomy of a Java program
  - Comments:
    - Multi-line documentation
      - `/*` Start of documentation
      - `*/` End of documentation
    - Documentation for a single line
      - `//` Everything until the end of the line is a comment
  - Package: The second line in the program (package chapter1;) specifies a package name, chapter1, for the class Welcome. Forte compiles the source code in `Welcome.java`, generates `Welcome.class`, and stores `Welcome.class` in the `chapter1` folder.
    - Cách khai báo: `Import xx.xxx.`
    - `Import java.awt.*;`
    - `Import java.applet.Applet;`



# TỔNG QUAN VỀ JAVA

---

- Anatomy of a Java program
  - Reserved word (hoặc keywords): Từ khóa là những từ có nghĩa cụ thể đối với trình biên dịch và chỉ có một mục đích duy nhất trong chương trình. Ví dụ từ khóa: *class*, *public*, *static*, *void*, ...
  - *Modifier*: Java sử dụng một số từ khóa để chỉ ra phạm vi sử dụng của dữ liệu, phương thức và lớp. Các từ khóa này được gọi là Modifier. Ví dụ từ khóa Private, Public, ...
  - *Statement*: Lệnh được dùng để mô tả một hoạt động hoặc một chuỗi các hoạt động của chương trình. Ví dụ lệnh System.out.println("Welcome to Java!") dùng để hiển thị chuỗi "Welcome to Java!"

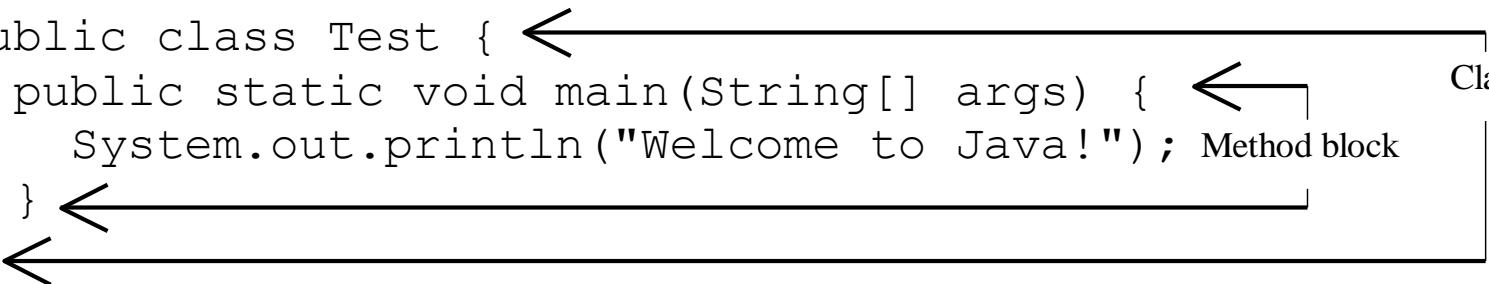
# TỔNG QUAN VỀ JAVA

---

- Anatomy of a Java program

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

# TỔNG QUAN VỀ JAVA

---

- Anatomy of a Java program
  - Một số lưu ý: Tất cả các lệnh của Java đều phải kết thúc bằng dấu chấm phẩy (;)

## **Dạng tổng quát:**

Instruction1;

Instruction2;

Instruction3;

:  
:

## **Ví dụ:**

int num = 0;

System.out.println(num);

:  
:

# CẤU TRÚC LỆNH TRONG JAVA

- Đưa dữ liệu ra màn hình

- Cấu trúc lệnh:

`System.out.print(<string or variable name one> + <string or variable name two>..);`

Hoặc

`System.out.println(<string or variable name one> + <string or variable name two>..);`

- Ví dụ:

Kết quả:

```
helloworld.java  ✖  vidu2.java
1
2 public class helloworld{
3     public static void main(String[] args){
4         int num = 123;    // More on this shortly
5         System.out.println("Hello D16!");
6         System.out.print(num);
7         System.out.println("num="+num);
8     }
9 }
```

```
Hello D16!
123num=123
```

# CẤU TRÚC LỆNH TRONG JAVA

- Đưa dữ liệu ra màn hình
  - Một số kiểu định dạng:

Escape sequence	Description
\t	Horizontal tab
\r	Carriage return
\n	New line
\"	Double quote
\\	Backslash

```

2 public class helloworld{
3     public static void main(String[] args){
4
5         System.out.print("You are smart \t students \n");
6         System.out.println("Good\rluck");
7         System.out.println("\"PTIT\" dot (\\) com");
8
9     }
10 }
    
```

```

You are smart      students
Good
luck
"PTIT" dot (\\) com
    
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Biến trong Java
  - Biến là một ô nhớ dùng để lưu giá trị
  - Lưu ý tên biến:
    - Thường bắt đầu bằng một chữ cái
    - Có thể chứa một số lượng bất kỳ các chữ cái hoặc chữ số
    - Có thể chứa dấu gạch dưới hoặc dấu \$
    - Có thể dài tùy ý
    - Phân biệt chữ hoa và chữ thường

# CẤU TRÚC LỆNH TRONG JAVA

---

- Biến trong Java

- Khai báo biến:

*<type of information> <name of variable>;*

- Ví dụ:

*char myFirstInitial;*

- Khởi tạo biến:

*char myFirstInitial = 'j';*

*int age = 30;*

# CẤU TRÚC LỆNH TRONG JAVA

- Một số kiểu biến trong Java

Type	Description
byte	8 bit signed integer
short	16 bit signed integer
int	32 bit signed integer
long	64 bit signed integer
float	32 bit signed real number
double	64 bit signed real number
char	16 bit Unicode character (ASCII and beyond)
boolean	1 bit true or false value
String	A sequence of characters between double quotes ("" )



# CẤU TRÚC LỆNH TRONG JAVA

---

- Vị trí khai báo biến

```
public class <name of class>
{
    public static void main (String[] args)
    {
        // Local variable declarations occur here
        << Program statements >>
    }
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Vị trí khai báo biến
  - Lưu ý: Luôn khởi tạo biến trước khi sử dụng

```
public class OutputExample1
{
    public static void main (String [] args)
    {
        int num;
        System.out.print(num);
    }
}
```

**OutputExample1.java:7: error: variable num might not have been initialized**     **System.out.print(num);**

# CẤU TRÚC LỆNH TRONG JAVA

---

- Hằng số
  - Hằng số là một loại biến có tên và giá trị không bị thay đổi trong suốt chương trình
  - Khai báo hằng:  

```
final <constant type> <CONSTANT NAME> = <value>;
```
  - Ví dụ:  

```
final int SIZE = 100;
```
  - Lưu ý: Một khi hằng số đã được khai báo thì không thể thay đổi giá trị của nó.

# CẤU TRÚC LỆNH TRONG JAVA

- Hằng số
  - Lưu ý: Một khi hằng số đã được khai báo thì không thể thay đổi giá trị của nó.

```
*helloworld.java ✕  
  
1  class Bike9{  
2      final int speedlimit=90; //bien final  
3      void run() {  
4          speedlimit=400; // báo lỗi  
5          System.out.print("Speed:" + speedlimit);  
6      }  
7  }  
8  public class helloworld{  
9  
10     public static void main(String[] args){  
11         Bike9 obj=new Bike9();  
12         obj.run();  
13     }  
14 }
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Hằng số
  - Lý do nên sử dụng hằng số:
    - Làm cho chương trình dễ hiểu và dễ sửa đổi hơn.

```
populationChange = (0.1758 – 0.1257) * currentPopulation;
```

Vs.

```
final float BIRTH_RATE = 17.58;  
final float MORTALITY_RATE = 0.1257;  
int currentPopulation = 1000000;  
populationChange = (BIRTH_RATE - MORTALITY_RATE) * currentPopulation;
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Một số quy ước về đặt tên biến
  - Không được đặt tên biến trùng với keywords.
  - Tên biến nên toát được mục đích của biến
  - Tránh sử dụng dấu \$
  - Nếu tên biến là một từ thì tất cả các ký tự nên để chữ thường
    - Ví dụ: `int count`; `String name`;
  - Nếu tên biến gồm nhiều từ thì nên viết hoa chữ cái đầu của từ thứ 2 trở đi.
    - Ví dụ: `String firstName`; `int personNumber`;

# CẤU TRÚC LỆNH TRONG JAVA

- Một số từ khóa

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

# CẤU TRÚC LỆNH TRONG JAVA

- Một số toán tử tăng/giảm

Precedence level	Operator	Description	Associativity
1	expression++ expression--	Post-increment Post-decrement	Right to left
2	++expression --expression + - ! ~ (type)	Pre-increment Pre-decrement Unary plus Unary minus Logical negation Bitwise complement Cast	Right to left



# CẤU TRÚC LỆNH TRONG JAVA

- Một số toán tử tính toán

Precedence level	Operator	Description	Associativity
3	* / %	Multiplication Division Remainder/modulus	Left to right
4	+ -	Addition or String concatenation Subtraction	Left to right
5	<< >>	Left bitwise shift Right bitwise shift	Left to right

# CẤU TRÚC LỆNH TRONG JAVA

- Một số toán tử so sánh

Precedence level	Operator	Description	Associativity
6	< <= > >=	Less than Less than, equal to Greater than Greater than, equal to	Left to right
7	= = !=	Equal to Not equal to	Left to right
8	&	Bitwise AND	Left to right
9	^	Bitwise exclusive OR	Left to right

# CẤU TRÚC LỆNH TRONG JAVA

- Một số toán tử logic

Precedence level	Operator	Description	Associativity
10		Bitwise OR	Left to right
11	&&	Logical AND	Left to right
12		Logical OR	Left to right

# CẤU TRÚC LỆNH TRONG JAVA

- Một số toán tử logic

Precedence level	Operator	Description	Associativity
13	=	Assignment	Right to left
	+=	Add, assignment	
	-=	Subtract, assignment	
	*=	Multiply, assignment	
	/=	Division, assignment	
	%=	Remainder, assignment	
	&=	Bitwise AND, assignment	
	^=	Bitwise XOR, assignment	
	=	Bitwise OR, assignment	
	<<=	Left shift, assignment	
	>>=	Right shift, assignment	

# CẤU TRÚC LỆNH TRONG JAVA

---

- Một số toán tử - Ví dụ

```
public class helloworld{  
    public static void main(String[] args){  
        int num = 5;  
        System.out.println(num);  
        num++;  
        System.out.println(num);  
        ++num;  
        System.out.println(num);  
        System.out.println(++num);  
        System.out.println(num++);  
    }  
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Một số toán tử - Ví dụ

```
public class helloworld{  
    public static void main(String[] args){  
        int num1;  
        int num2;  
        num1 = 5;  
        num2 = ++num1 * num1++;  
        System.out.println("num1=" + num1);  
        System.out.println("num2=" + num2);  
    }  
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Một số toán tử - Ví dụ

```
public class helloworld{  
    public static void main(String[] args){  
        int num = 5;  
        float fl;  
        System.out.println(num);  
        num = num * -num;  
        System.out.println(num);  
    }  
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Truy cập thư viện trong Java

- Lệnh:

- ```
import <Full library name>;
```

- Ví dụ

- ```
import java.util.Scanner;
```



# CẤU TRÚC LỆNH TRONG JAVA


---

- Truy cập thư viện trong Java

```
import java.util.Scanner;
```

```
main (String [] args)
{
    Scanner <name of scanner> = new Scanner (System.in);
    <variable> = <name of scanner> .<method> ();
}
```

**Creating a  
scanner object  
(something  
that can scan  
user input)**



**Using the capability of  
the scanner object  
(actually getting user  
input)**



# CẤU TRÚC LỆNH TRONG JAVA

---

- Truy cập thư viện trong Java

File name: **MyInput.java**

```
import java.util.Scanner;

public class MyInput
{
    public static void main (String [] args)
    {
        String str1;
        int num1;
        Scanner in = new Scanner (System.in);
        System.out.print ("Type in an integer: ");
        num1 = in.nextInt ();
        System.out.print ("Type in a line: ");
        in.nextLine ();
        str1 = in.nextLine ();
        System.out.println ("num1:" +num1 +"\\t str1:" + str1);
    }
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Truy cập thư viện trong Java
  - `nextInt ()`
  - `nextLong ()`
  - `nextFloat ()`
  - `nextDouble ()`
  - `nextLine ();`

# CẤU TRÚC LỆNH TRONG JAVA

---

- Truy cập thư viện trong Java
  - Ví dụ: *Viết một giao diện Game đưa ra thông báo có các tùy chọn để người chơi chọn lựa bằng cách nhập chữ cái đầu. Sau đó hiển thị chữ cái đầu của người chơi đã nhập.*

## GAME OPTIONS

(a)dd a new player  
(l)oad a saved game  
(s)ave game  
(q)uit game

# CÂU TRÚC LỆNH TRONG JAVA

---

- Truy cập thư viện trong Java

```
import java.util.Scanner;
public class helloworld{
    public static void main(String[] args){
        final int FIRST = 0;
        String selection;
        Scanner firstChar = new Scanner (System.in);
        System.out.println("GAME OPTIONS");
        System.out.println("(a)dd a new player");
        System.out.println("(l)oad a saved game");
        System.out.println("(s)ave game");
        System.out.println("(q)uit game");
        System.out.print("Enter your selection: ");
        selection = firstChar.nextLine ();
        System.out.println ("Selection: " + selection.charAt(FIRST));
    }
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Truy cập thư viện trong Java
  - Ví dụ: *Viết chương trình để người dùng nhập hai số. Chương trình tính tổng hai số và kết quả được gán vào biến thứ nhất. Chương trình có sử dụng hàm calSum để tính tổng.*
  - Gợi ý:
    - Khai báo hàm: *Static void calSum(){*  
  
*}*
    - Tính tổng rồi lưu vào biến thứ nhất:  
  
*a+=b;*

# CẤU TRÚC LỆNH TRONG JAVA

---

- Truy cập thư viện trong Java

```
import java.util.Scanner;
public class helloworld{
    static void calSum(){
        int a,b;
        Scanner firstNum =new Scanner(System.in);
        Scanner secondNum =new Scanner(System.in);
        System.out.println("Nhập số thứ nhất:");
        a= firstNum.nextInt();
        System.out.println("Nhập số thứ hai:");
        b=secondNum.nextInt();
        a+=b;
        System.out.println("Tổng hai số là: " + a);
    }
    public static void main(String[] args){
        calSum();
    }
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java
  - if
  - if, else
  - if, else-if
  - switch

Logical Operation	Java
AND	&&
OR	
NOT	!
EQUAL	==



# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java

## Format:

```
if (Boolean Expression)  
    Body
```

## Example:

```
if (x != y)  
    System.out.println("X and Y are not equal");  
if (x == y)  
    System.out.println("X and Y are equal");  
if ((x > 0) && (y > 0))  
{  
    System.out.println("X and Y are positive");  
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java

## Format:

*if (Boolean expression)*

*Body of if*

*else*

*Body of else*

## Example:

*if (x < 0)*

*System.out.println("X is negative");*

*else*

*System.out.println("X is non-negative");*

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java

## Format:

```
if (Boolean Expression)  
    Body
```

## Example:

```
if (x != y)  
    System.out.println("X and Y are not equal");  
if (x == y)  
    System.out.println("X and Y are equal");  
if ((x > 0) && (y > 0))  
{  
    System.out.println("X and Y are positive");  
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java
  - Ví dụ: Cho phép người dùng nhập tuổi để tham gia bầu cử. Nếu tuổi  $< 18$ : hiển thị thông báo “Không hợp lệ”. Nếu tuổi  $\geq 18$  hiển thị thông báo “Hợp lệ”. Chương trình dùng phương thức `ageCheck` để kiểm tra tuổi.

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java

```
import java.util.Scanner;
public class helloworld{
    public static void main(String[] args) {
        ageCheck();
    }
    static void ageCheck(){
        int age;
        System.out.print("Input your age:");
        Scanner s=new Scanner(System.in);
        age=s.nextInt();
        if(age<10)
            System.out.println("Not Eligible");
        else
            System.out.println("OK");
    }
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java

**Format (character-based switch):**

switch (*character variable name*)

{

case '<*character value*>':

*Body*

    break;

case '<*character value*>':

*Body*

    break;

-----

-----

default:

*Body*

}

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh rẽ nhánh trong Java
  - Ví dụ: Viết chương trình “Thi trắc nghiệm” như sau
  - Đưa ra câu hỏi: “ Cách khai báo đúng trong Java là:” và các câu trả lời:
    - a. `int 1x=10;`
    - b. `int x=10;`
    - c. `float x=10.0;`
    - d. `string x="10";`

Chương trình kiểm tra ký tự nhập vào. Nếu đáp án đúng thì in ra thông báo “Đúng”, nếu đáp án sai thì in ra thông báo “Sai”.

# CẤU TRÚC LỆNH TRONG JAVA

- Lệnh rẽ nhánh trong Java

```
import java.util.Scanner;
public class helloworld{
    public static void main(String[] args) {
        ansCheck();
    }
    static void ansCheck(){
        System.out.println("Cách khai báo đúng trong Java là:");
        System.out.println("\t a. int 1x=10;");
        System.out.println("\t b. int x=10;");
        System.out.println("\t c. float x=10.0;");
        System.out.println("\t d. string x=\"10\";");
        System.out.print("Câu trả lời:");

        Scanner s=new Scanner(System.in);
        String ans = s.nextLine();
        switch(ans.charAt(0)){
            case 'a': System.out.println("wrong!"); break;
            case 'b': System.out.println("Correct!"); break;
            case 'c': System.out.println("wrong!"); break;
            case 'd': System.out.println("wrong!"); break;
            default: System.out.println("wrong!"); break;
        }
    }
}
```



# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java
  - Java Pre-test loops
    - For
    - While
  - Java Post-test loop
    - Do-while
  - Post-test kiểm tra biểu thức logic sau khi mỗi vòng lặp. Điều này có nghĩa là vòng lặp được thực hiện ít nhất một lần.
  - Pre-test kiểm tra biểu thức trước mỗi vòng lặp.

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java

## **Format:**

```
while (Boolean expression)  
    Body
```

## **Example:**

```
int i = 1;  
while (i <= 4)  
{  
    // Call function  
    createNewPlayer();  
    i = i + 1;  
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java

**Format:**

*for (initialization; Boolean expression; update control)*  
*Body*

**Example:**

```
for (i = 1; i <= 4; i++)  
{  
    // Call function  
    createNewPlayer();  
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java

**Format:**

```
do
    Body
while (Boolean expression);
```

**Example:**

```
char ch = 'A';
do
{
    System.out.println(ch);
    ch++;
}
while (ch <= 'K');
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java
  - Ví dụ: Viết chương trình hỏi người chơi có tiếp tục chơi hay thoát khỏi chương trình. Nếu người chơi chọn “q” hoặc “Q” thì thoát.

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java

```
import java.util.Scanner;
public class helloworld{
    public static void main(String[] args) {
        final int FIRST = 0;
        Scanner in = new Scanner(System.in);
        char answer = ' ';
        String temp;
        while ((answer != 'q') && (answer != 'Q'))
        {
            System.out.print("Play again? Enter 'q' to quit: ");
            temp = in.nextLine();
            answer = temp.charAt(FIRST);
        }
    }
}
```

# CÂU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java
  - Ví dụ: Viết chương trình in ra màn hình ký tự “\*” như sau:

```
*  
**  
***  
****  
*****  
*****  
*****  
*****  
*****  
*****  
*****
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Lệnh lặp trong Java
  - Ví dụ: Viết chương trình in ra màn hình ký tự “\*” như sau:

```
import java.util.Scanner;  
public class helloworld{  
    public static void main(String[] args) {  
        for(int i=1;i<=10;i++){  
            for(int j=1;j<=i;j++){  
                System.out.print("*");  
            }  
            System.out.print("\n");  
        }  
    }  
}
```



# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng trong Java
  - Mảng là một tập hợp các phần tử có kiểu tương tự nhau và có vị trí ô nhớ liên kề.
  - Mảng trong Java là một đối tượng chứa các phần tử có kiểu dữ liệu giống nhau.
  - Chỉ có thể lưu trữ một tập hợp cố cố định các phần tử trong một mảng trong Java.
  - Mảng trong Java là dựa trên chỉ mục (index), phần tử đầu tiên của mảng được lưu trữ tại chỉ mục 0.

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng trong Java
  - Ưu điểm của mảng:
    - **Tối ưu hóa code:** từ đó chúng ta có thể thu nhận và sắp xếp dữ liệu một cách dễ dàng.
    - **Truy cập ngẫu nhiên:** chúng ta có thể lấy bất cứ dữ liệu nào ở tại bất cứ vị trí chỉ mục nào.
  - Nhược điểm của mảng:
    - **Giới hạn kích cỡ:** Chúng ta chỉ có thể lưu trữ kích cỡ cố định số phần tử trong mảng. Nó không tăng kích cỡ của nó tại runtime.

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng trong Java
  - Khai báo một biến mảng  
`Kiểu_dữ_liệu[] tên_Biến;`  
`int[] firstArray;`
  - Khai báo và tạo một mảng:  
`Kiểu_dữ_liệu[] tên_Biến;`  
`tên_Biến = new Kiểu_dữ_liệu[Số_phần_tử];`  
`int[] firstArray=new int[10];`
  - Trong JAVA, kiểu int dài 4 bytes, vì vậy độ dài mảng sẽ là:  $4 * 10 = 40$  bytes

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng trong Java
  - Ví dụ: Viết chương trình nhập vào số phần tử của mảng, sau đó khởi tạo mảng có số phần tử do người dùng nhập vào. In số phần tử của mảng ra màn hình (`firstArray.length`)

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng trong Java

```
import java.util.Scanner;
public class helloworld{
    public static void main(String[] args) {
        int sizeArray;
        System.out.println("Nhập số phần tử của mảng:");
        Scanner in=new Scanner(System.in);
        sizeArray=in.nextInt();
        int[] firstArray = new int[sizeArray];
        System.out.println("Số phần tử của mảng:" + firstArray.length);
    }
}
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng trong Java
  - Bài tập: Tìm số phần tử bé nhất và lớn nhất trong một mảng

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng 2 chiều trong Java
  - `float[][] temperature=new float[10][365];`
  - 10 arrays each having 365 elements
  - First index: specifies array (row)
  - Second Index: specifies element in that array (column)
  - In JAVA float is 4 bytes, total Size= $4*10*365=14,600$  bytes

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng 2 chiều trong Java
  - Tất cả các mảng không nhất thiết phải có cùng độ dài  
`float[][] samples;`  
`samples=new float[6][]; //defines # of arrays`  
`samples[2]=new float[6];`  
`samples[5]=new float[101];`
  - Java không yêu cầu phải khởi tạo đầy đủ tất cả các mảng



# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng 2 chiều trong Java
  - Java không yêu cầu phải khởi tạo đầy đủ tất cả các mảng

```
int[][] uneven = { { 1, 9, 4 }, { 0, 2 }, { 0, 1, 2, 3, 4 } };
```

```
//Three arrays
```

```
//First array has 3 elements
```

```
//Second array has 2 elements
```

```
//Third array has 5 elements
```

# CẤU TRÚC LỆNH TRONG JAVA

---

- Khai báo mảng 2 chiều trong Java
  - Bài tập: Nhân hai ma trận kích thước do người dùng nhập vào.