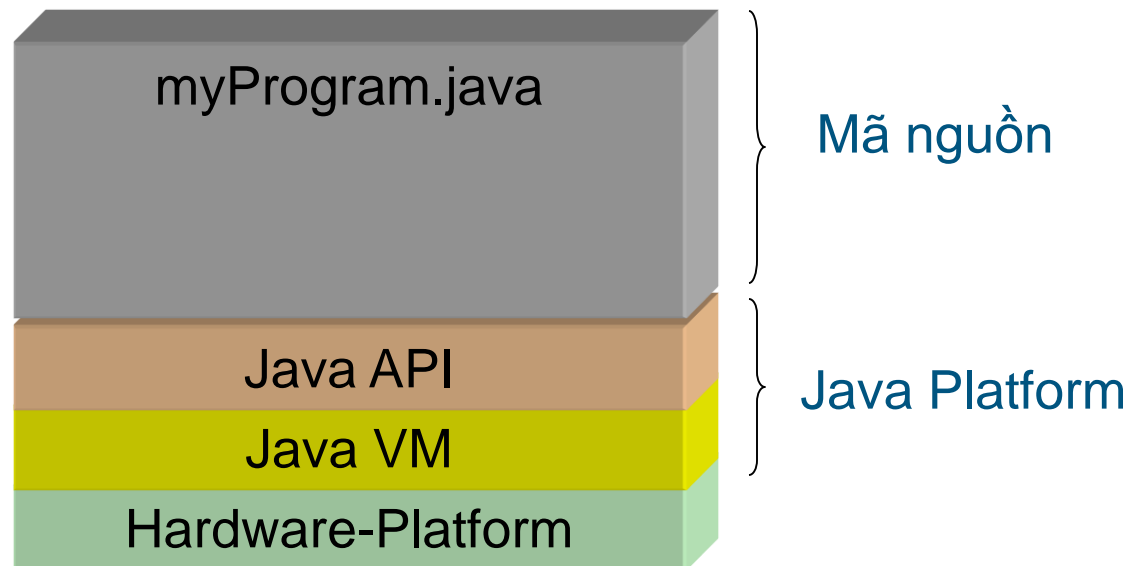


# JAVA CƠ BẢN

# CẤU TRÚC MỘT CHƯƠNG TRÌNH JAVA CƠ BẢN

# KIẾN TRÚC CỦA JAVA

- **Java Platform**
  - Java Virtual Machine (Java VM)
  - Java Application Programming Interface (Java API)

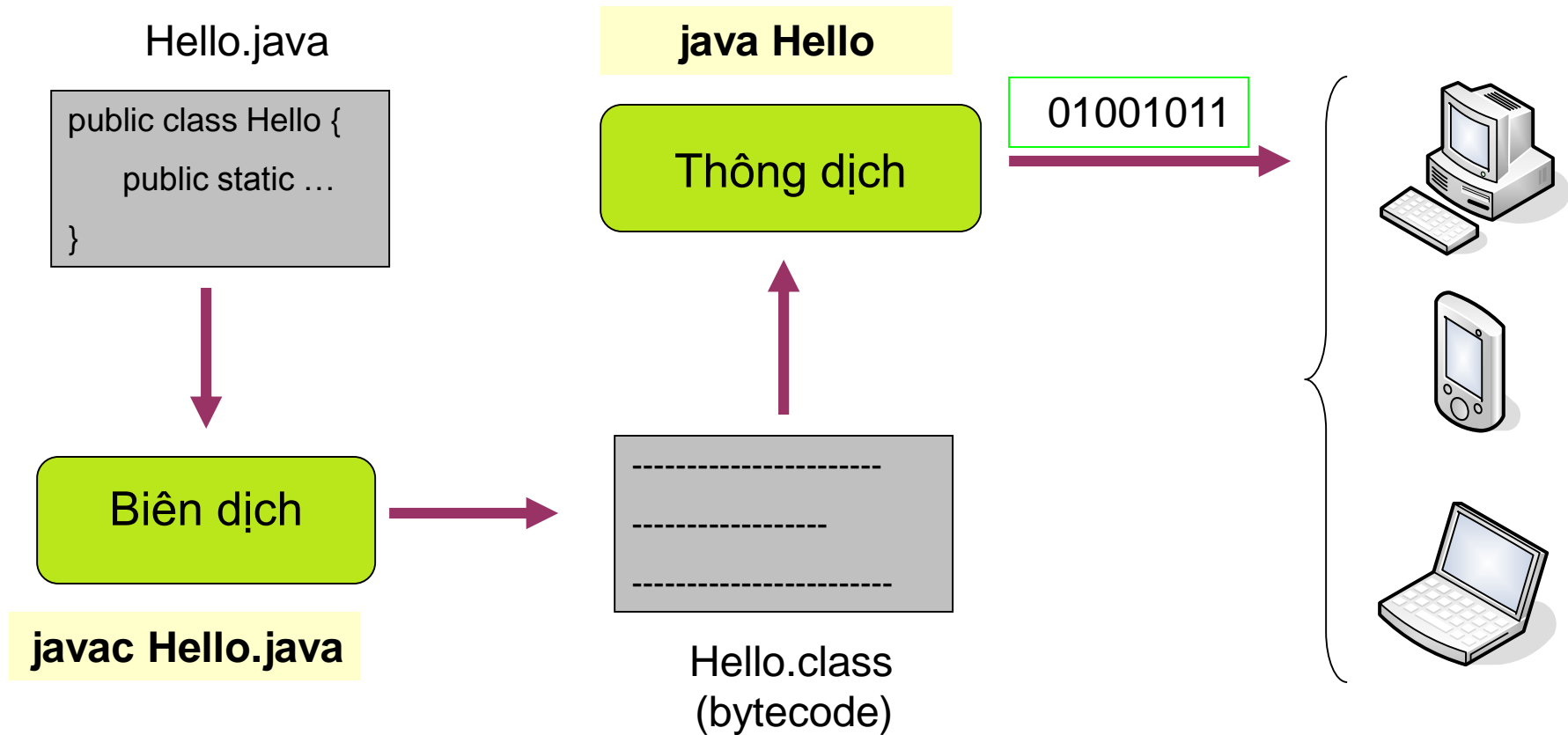


# KIẾN TRÚC CỦA JAVA

- Thư viện lớp Java: bộ JDK bao gồm rất nhiều lớp chuẩn đã được xây dựng sẵn.
- Lập trình viên thường sử dụng các lớp chuẩn để phát triển ứng dụng.
- Các gói chuẩn của Java:
  - java.lang
  - java.applet
  - java.awt
  - java.io
  - java.util
  - java.net
  - java.awt.event
  - java.rmi
  - java.security
  - java.sql

# CÁC BƯỚC PHÁT TRIỂN

- Các bước phát triển một chương trình bằng Java:



# CẤU TRÚC MỘT CHƯƠNG TRÌNH CƠ BẢN

```
1 // Tên file : Hello.java
2 /* Tác giả : Barak Obama*/
3
4 public class Hello
5 {
6 // Phương thức main, điểm bắt đầu của chương trình
7 public static void main( String args[] )
8 {
9     System.out.println( "Hello World" );
10
11 } // Kết thúc phương thức main
12
13 }
```

Tên lớp chứa hàm main phải giống tên file

Điểm bắt đầu và kết thúc của lớp

**Dấu hiệu chú thích =>**  
Làm cho chương trình dễ hiểu hơn. Trình biên dịch sẽ bỏ qua những dòng có dấu chú thích

**Khai báo lớp**

Mỗi CT phải có ít nhất một khai báo lớp

Hiển thị dãy ký tự ra màn hình

Phương thức main() sẽ được gọi đầu tiên. Mỗi CT thực thi phải có một phương thức main()

Các câu lệnh phải kết thúc bằng dấu chấm phẩy

# PHƯƠNG THỨC MAIN

- **Phương thức main():** là điểm bắt đầu thực thi một ứng dụng.
- Mỗi ứng dụng Java phải chứa một phương thức main có dạng như sau: **public static void main(String[] args)**
- Phương thức main chứa ba bổ từ đặc tả sau:
  - **public:** chỉ ra rằng phương thức main có thể được gọi bởi bất kỳ đối tượng nào.
  - **static:** chỉ ra rằng phương thức main là một phương thức lớp.
  - **void:** chỉ ra rằng phương thức main sẽ không trả về bất kỳ một giá trị nào.

# CHÚ THÍCH TRONG JAVA

- Ngôn ngữ Java hỗ trợ ba kiểu chú thích sau:

`/* text */`

`// text`

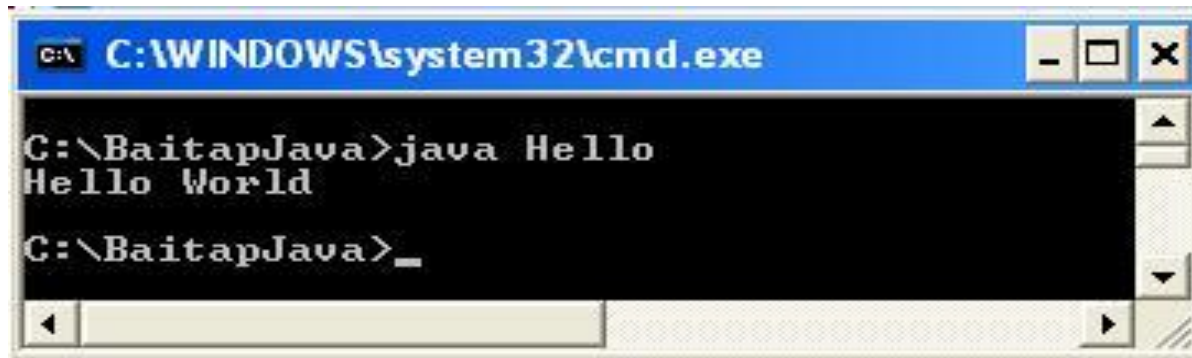
`/** documentation */` công cụ javadoc trong bộ JDK sử dụng chú thích này để chuẩn bị cho việc tự động phát sinh tài liệu.

- Dấu mở và đóng ngoặc nhọn “{” và “}” là bắt đầu và kết thúc một khối lệnh.
- Dấu chấm phẩy “;” để kết thúc một dòng lệnh.
- Java được tổ chức theo lớp (class). Các lệnh và các hàm (kể cả hàm main) phải thuộc một lớp nào đó, chúng không được đứng bên ngoài của lớp.



# BIÊN DỊCH VÀ THỰC THI

- Biên dịch chương trình
  - Vào chế độ Console của Windows
  - Gõ câu lệnh ***javac Hello.java***
  - Nếu không có thông báo lỗi, file *Hello.class* sẽ được tạo ra
- Thực thi chương trình
  - Gõ câu lệnh ***java Hello*** (không cần *.class*)



```
C:\WINDOWS\system32\cmd.exe

C:\BaitapJava>java Hello
Hello World

C:\BaitapJava>_
```

The image shows a Windows Command Prompt window with a blue title bar. The title bar text is "C:\WINDOWS\system32\cmd.exe". The command prompt shows the user's current directory as "C:\BaitapJava". The user has entered the command "java Hello", and the output "Hello World" is displayed on the next line. The prompt is now waiting for the next command, indicated by an underscore "\_".

# HÀNG, BIẾN, KIỂU DỮ LIỆU TOÁN TỬ

# TỪ KHÓA (keyword)

- Từ khóa cho các kiểu dữ liệu cơ bản : **byte, short, int, long, float, double, char, boolean.**
- Từ khóa cho phát biểu lặp: **do, while, for, break, continue.**
- Từ khóa cho phát biểu rẽ nhánh: **if, else, switch, case, default, break.**
- Từ khóa đặc tả đặc tính một method: **private, public, protected, final, static, abstract, synchronized.**
- Hằng (literal): **true, false, null.**
- Từ khóa liên quan đến method: **return, void.**
- Từ khóa liên quan đến package: **package, import.**
- Từ khóa cho việc quản lý lỗi: **try, catch, finally, throw, throws.**
- Từ khóa liên quan đến đối tượng: **new, extends, implements, class, instanceof, this, super.**

# TỪ KHÓA (keyword)

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	

# ĐỊNH DANH (identifier)

- Định danh là dùng biểu diễn tên của biến, của phương thức, của lớp.
- Trong Java, định danh có thể sử dụng ký tự chữ, ký tự số và ký tự dấu.
- Ký tự đầu tiên phải là ký tự chữ, dấu gạch dưới (\_), hoặc dấu dollar (\$).
- Có sự phân biệt giữa ký tự chữ hoa và chữ thường.  
Ví dụ: Hello, \_prime, var8, tvLang

# BIẾN (variable)

- Biến là vùng nhớ dùng để lưu trữ các giá trị của chương trình.
- Mỗi biến gắn liền với một kiểu dữ liệu và một định danh duy nhất gọi là tên biến.
- Tên biến thông thường là một chuỗi các ký tự (Unicode), ký số.
- Tên biến phải bắt đầu bằng một chữ cái, một dấu gạch dưới hay dấu dollar.
- Tên biến không được trùng với các từ khóa (xem lại các từ khóa trong java).
- Tên biến không có khoảng trắng ở giữa tên.
- Trong java, biến có thể được khai báo ở bất kỳ nơi đâu trong chương trình.

# BIẾN (variable)

- **Cách khai báo**

<kiểu\_dữ\_liệu> <tên\_biến>;

<kiểu\_dữ\_liệu> <tên\_biến> = <giá\_trị>;

- **Gán giá trị cho biến**

<tên\_biến> = <giá\_trị>;

- **Biến công cộng (toàn cục):** là biến có thể truy xuất ở khắp nơi trong chương trình, thường được khai báo dùng từ khóa public, hoặc đặt chúng trong một class.
- **Biến cục bộ:** là biến chỉ có thể truy xuất trong khối lệnh nó khai báo

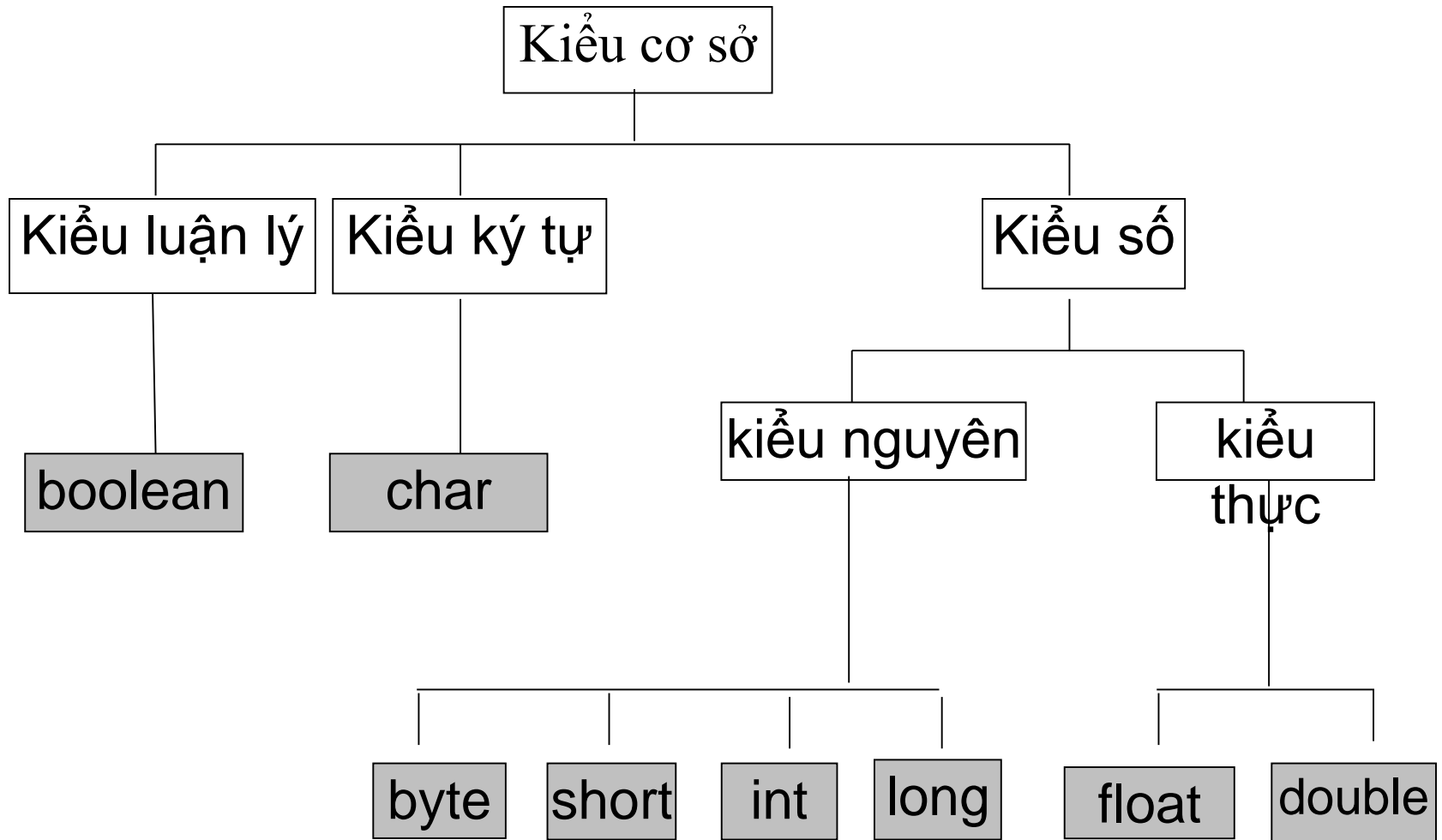
# KIỂU DỮ LIỆU (data type)

Kiểu dữ liệu:

- Kiểu dữ liệu cơ sở (Primitive data type)
- Kiểu dữ liệu tham chiếu hay dẫn xuất (reference data type)
- Kiểu dữ liệu cơ sở của Java bao gồm các nhóm sau: số nguyên, số thực, ký tự, kiểu luận lý (logic)
- Kiểu dữ liệu tham chiếu là các kiểu dữ liệu đối tượng. Ví dụ như: String, Byte, Character, Double, Boolean, Integer, Long, Short, Font,... và các lớp do người dùng định nghĩa.



# KIỂU DỮ LIỆU CƠ SỞ (primitive type)



# Kiểu dữ liệu cơ sở

- Kiểu số nguyên

Kiểu	Kích thước	Khoảng giá trị
<b>byte</b>	8 bits	-256...255
<b>short</b>	16 bits	-32768...32767
<b>int</b>	32 bits	$-2^{32} \dots 2^{32} - 1$
<b>long</b>	64 bits	$-2^{64} \dots 2^{64} - 1$

- Kiểu số thực

Kiểu	Kích thước	Khoảng giá trị
<b>float</b>	32 bits	-3.4e38...3.4e38
<b>double</b>	64 bits	-1.7e308...1.7e308

# Kiểu dữ liệu cơ sở

- Kiểu **boolean**: Nhận giá trị true hoặc false
- Kiểu **char**: Kiểu ký tự theo chuẩn Unicode

Một số hằng ký tự:

Ký tự	Ý nghĩa
\b	Xóa lùi (BackSpace)
\t	Tab
\n	Xuống hàng
\r	Dấu enter
\”	Nháy kép
\’	Nháy đơn
\\	Số ngược
\f	Đẩy trang
\uxxxx	Ký tự unicode

# Kiểu dữ liệu cơ sở

## Kiểu số nguyên

- Bốn kiểu số nguyên khác nhau là: byte, short, int, long
- Kiểu mặc định của các số nguyên là kiểu int
- Không có kiểu số nguyên không dấu
- Không thể chuyển biến kiểu int và kiểu boolean như trong ngôn ngữ C/C++

VD:   int x = 0;  
       long y=100;  
       int a=1,b,c;

# Kiểu dữ liệu cơ sở

- Kiểu số nguyên:

```
boolean b = false;
```

```
if (b == 0)
```

```
{
```

```
    System.out.println("Xin chào");
```

```
}
```

*Lúc biên dịch, đoạn chương trình trên sẽ báo lỗi vì ta không được so sánh biến kiểu boolean với biến kiểu int.*

# Kiểu dữ liệu cơ sở

**Một số lưu ý đối với các phép toán trên số nguyên:**

- Nếu hai toán hạng kiểu long thì kết quả là kiểu long.
- Một trong hai toán hạng không phải kiểu long sẽ được chuyển thành kiểu long trước khi thực hiện phép toán.
- Nếu hai toán hạng đầu không phải kiểu long thì phép tính sẽ thực hiện với kiểu int.
- Các toán hạng kiểu byte hay short sẽ được chuyển sang kiểu int trước khi thực hiện phép toán.

# KIỂU DỮ LIỆU CƠ SỞ

## *Kiểu dấu chấm động:*

- Kiểu float có kích thước 4 byte và giá trị mặc định là 0.0f
- Kiểu double có kích thước 8 byte và giá trị mặc định là 0.0d
- Khai báo và khởi tạo giá trị cho các biến kiểu dấu chấm động:

float x = 100.0/7;

double y = 1.56E6;

# Kiểu dữ liệu cơ sở

***Một số lưu ý với các phép toán trên số dấu chấm động:***

- Nếu mỗi toán hạng đều có kiểu dấu chấm động thì phép toán chuyển thành phép toán dấu chấm động.
- Nếu có một toán hạng là double thì các toán hạng còn lại sẽ được chuyển thành kiểu double trước khi thực hiện phép toán.
- Biến kiểu float và double có thể ép chuyển sang kiểu dữ liệu khác (trừ kiểu boolean).



# Kiểu dữ liệu cơ sở

- Kiểu ký tự
  - Biểu diễn các ký tự trong bộ mã Unicode
  - $2^{16} = 65536$  ký tự khác nhau :
  - từ `'\u0000'` đến `'\uFFFF'`
- Kiểu luận lý (boolean)
  - Hai giá trị: **true** hoặc **false**
  - Giá trị mặc định: **false**
  - Không thể chuyển thành kiểu nguyên và ngược lại

# Kiểu dữ liệu cơ sở

- Kiểu mảng

- Khai báo: `int[] iarray; hoặc int iarray[];`
- Cấp phát: `iarray = new int[100];`
- Khởi tạo:

`int[] iarray = {1, 2, 3, 5, 6};`

`char[] carray = {'a', 'b', 'c'};`

*Chú ý: Luôn khởi tạo hoặc cấp phát mảng trước khi sử dụng*

- Một số khai báo không hợp lệ:

`int[5] iarray;`

`int iarray[5];`

# Kiểu dữ liệu cơ sở

## Kiểu mảng

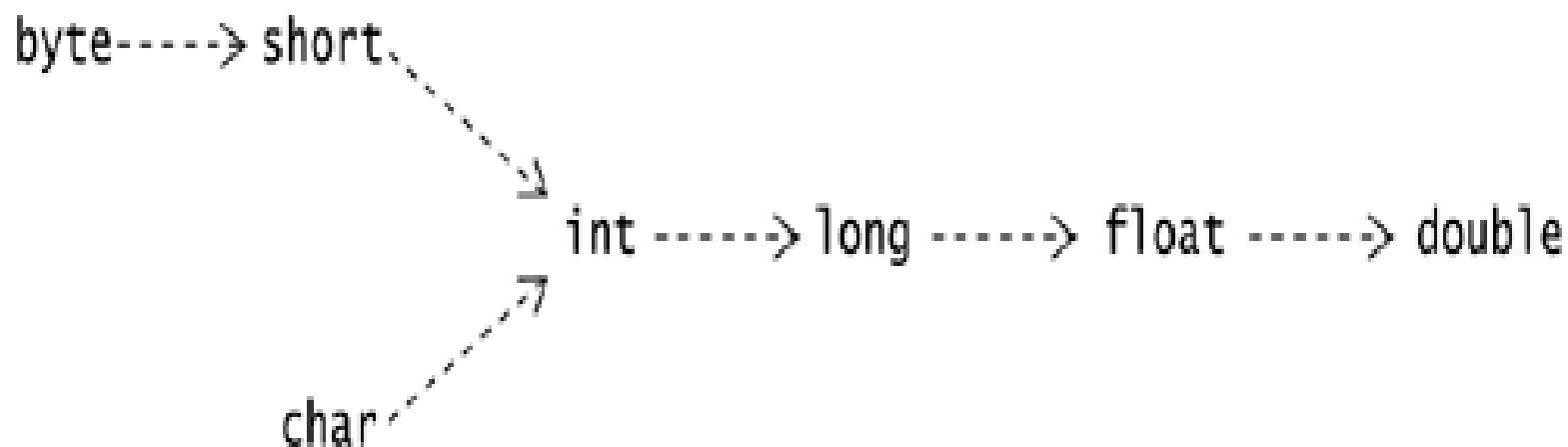
- Truy cập mảng
  - `iarray[3] = 0;`
  - `carray[1] = 'z';`

*Chú ý: Chỉ số của mảng được tính từ 0*

- Lấy số phần tử mảng:  
`iarray.length`

# Kiểu dữ liệu cơ sở

- Khi gặp phải sự không tương thích kiểu dữ liệu chúng ta phải tiến hành chuyển đổi kiểu dữ liệu cho biến hoặc biểu thức



# Kiểu dữ liệu cơ sở

- **Toán tử ép kiểu:**

`<tên biến> = (kiểu_dữ_liệu) <tên_biến>;`

`float fNum = 2.2;`

`int iCount = (int) fNum`

- **Ép kiểu rộng** (widening conversion): từ kiểu nhỏ sang kiểu lớn (không mất mát thông tin)
- **Ép kiểu hẹp** (narrow conversion): từ kiểu lớn sang kiểu nhỏ (có khả năng mất mát thông tin)

# HẰNG (LITERAL)

- Hằng là một giá trị bất biến trong chương trình
- Tên hằng được đặt theo qui ước giống như tên biến
- Tiếp vĩ ngữ: *l, L, f, F, d, D*  
*int i=1;*  
*long i=1L;*
- Hằng ký tự: là một ký tự đơn nằm giữa 2 dấu nháy đơn.

# HẰNG (LITERAL)

- **Hằng chuỗi:** là tập hợp các ký tự được đặt giữa hai dấu nháy kép “”. Một hằng chuỗi không có ký tự nào là một hằng chuỗi rỗng.

Ví dụ: “Hello Wolrd”

Lưu ý: Hằng chuỗi không phải là một kiểu dữ liệu cơ sở nhưng vẫn được khai báo và sử dụng trong các chương trình

# TOÁN TỬ (OPERATOR)

- Toán tử số học

Toán tử	Ý nghĩa
+	Cộng
-	Trừ
*	Nhân
/	Chia nguyên
%	Chia dư
++	Tăng 1
--	Giảm 1



# TOÁN TỬ (OPERATOR)

- Toán tử quan hệ & logic

Toán tử	Ý nghĩa
==	So sánh bằng
!=	So sánh khác
>	So sánh lớn hơn
<	So sánh nhỏ hơn
>=	So sánh lớn hơn hay bằng
<=	So sánh nhỏ hơn hay bằng
	OR (biểu thức logic)
&&	AND (biểu thức logic)
!	NOT (biểu thức logic)

# TOÁN TỬ (OPERATOR)

- Toán tử gán (assignment)

Toán tử	Ví dụ	Giải thích
=	int c=3, d=5, c=4;	
+=	c+=7	c=c+7
-=	d-=4	d=d-4
*=	e*=5	e=e*5
/=	f/=3	f=f/3
%=	g%=9	g=g%9

# TOÁN TỬ (OPERATOR)

- **Toán tử điều kiện**

<điều kiện> ? <biểu thức 1> : <biểu thức 2>

*int x = 10;*

*int y = 20;*

*int Z = (x < y) ? 30 : 40;*

# ĐỘ ƯU TIÊN CỦA CÁC PHÉP TOÁN

- Độ ưu tiên của các phép toán trong ngôn ngữ Java cũng gần giống như ngôn ngữ C/C++. Thứ tự ưu tiên từ trái qua phải và từ trên xuống dưới như bảng sau:

1	.	[]	()	
2	++	--	!	~
3	*	/	%	
4	+	-		
5	<<	>>	>>>	
6	<	>	<=	>=
7	==	!=		
8	&			
9	^			
10	&&			
11				
12	?:			
13	=			

# MỘT VÍ DỤ VỀ PHÉP TOÁN

## Ví dụ:

```
import java.lang.*;
import java.io.*;
class VariableDemo
{
    static int x, y;
    public static void main(String[] args)
    {
        x = 10;
        y = 20;
        int z = x+y;
        System.out.println("x = " + x);
        System.out.println("y = " + y);
        System.out.println("z = x + y =" + z);
        System.out.println("So nho hon la so:" +
            Math.min(x, y));
        char c = 80;
        System.out.println("ky tu c la: " + c);
    }
}
```

# CÁC CẤU TRÚC ĐIỀU KHIỂN TRONG JAVA

# CÁC CẤU TRÚC ĐIỀU KHIỂN

- Điều khiển rẽ nhánh:
  - Mệnh đề **if-else**
  - Mệnh đề **switch-case**
- Vòng lặp (Loops):
  - Vòng lặp **while**
  - Vòng lặp **do-while**
  - Vòng lặp **for**

# MỆNH ĐỀ IF - ELSE

- Mệnh đề if/else

```
import java.util.Date;
public class TestIf
{
    public static void main( String args[ ] )
    {
        Date today = new Date();
        if( today.getDay() == 0 )
            System.out.println("Hom nay la chu nhat\n");
        else
            System.out.println("Hom nay khong la chu
nhat\n" );
    }
}
```



# MỆNH ĐỀ SWITCH - CASE

- Mệnh đề switch/case

```
import javax.swing.JOptionPane;
public class TestSwitch
{
    public static void main(String[] args)
    {
        char c;
        String str=JOptionPane.showInputDialog(null,"Nhập vào ký tự?");
        c = str.charAt(0);
        switch(c)
        {
            case 'a': case 'e': case 'i': case 'o': case 'u':
                System.out.println("Ký tự này là nguyên âm");
                break;
            default:
                System.out.println("Ký tự này là phụ âm");
        }
        System.exit(0); // kết thúc chương trình
    }
}
```

# VÒNG LẶP WHILE

- Vòng lặp while
  - while (<biểu thức boolean>)  
    <khối lệnh>;

```
// Tính tổng các số lẻ từ 1 đến 100
int tong = 0, i = 1;
while (i<=100)
{
    tong+=i; i+=2;
}
System.out.println(tong);
```

# VÒNG LẶP DO - WHILE

- Vòng lặp do/while

do

{

<khối lệnh>;

} while <biểu thức boolean>;

```
// Tính tổng các số lẻ từ 1 đến 100
```

```
int tong = 0, i=1;
```

```
do
```

```
{
```

```
    tong+=i; i+=2;
```

```
} while (i<=100);
```

```
System.out.println(tong);
```

# VÒNG LẶP FOR

- Vòng lặp for

- for(<khởi tạo>; <điều kiện lặp>; <bước nhảy>)
- <khởi lệnh>;

// Chương trình tính tổng các số lẻ từ 1 đến 100

```
public class TestFor
{
    public static void main(String[] args)
    {
        int tong = 0;
        for(int i=1; i<=100; i+=2)
            tong+=i;
        System.out.println(tong);
    }
}
```

# NHẬP DỮ LIỆU TỪ BÀN PHÍM

## Nhập dữ liệu từ bàn phím

- Ví dụ nhập một số nguyên và một số thực

```
import java.io.*;
public class TestInput
{
    public static void main(String[] args) throws Exception
    {
        BufferedReader inStream =
            new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Nhap mot so nguyen:");
        String siNumber = inStream.readLine();
        int iNumber = Integer.parseInt(siNumber);
    }
}
```

# NHẬP DỮ LIỆU TỪ BÀN PHÍM

## Nhập dữ liệu từ bàn phím

```
System.out.print("Nhap mot so thuc:");  
String sfNumber = inStream.readLine();  
float fNumber = Float.parseFloat(sfNumber);  
  
System.out.println("So nguyen:" + iNumber);  
System.out.println("So thuc:" + fNumber);  
}  
}
```

**HẾT**