

NHẬP MÔN LẬP TRÌNH JAVA

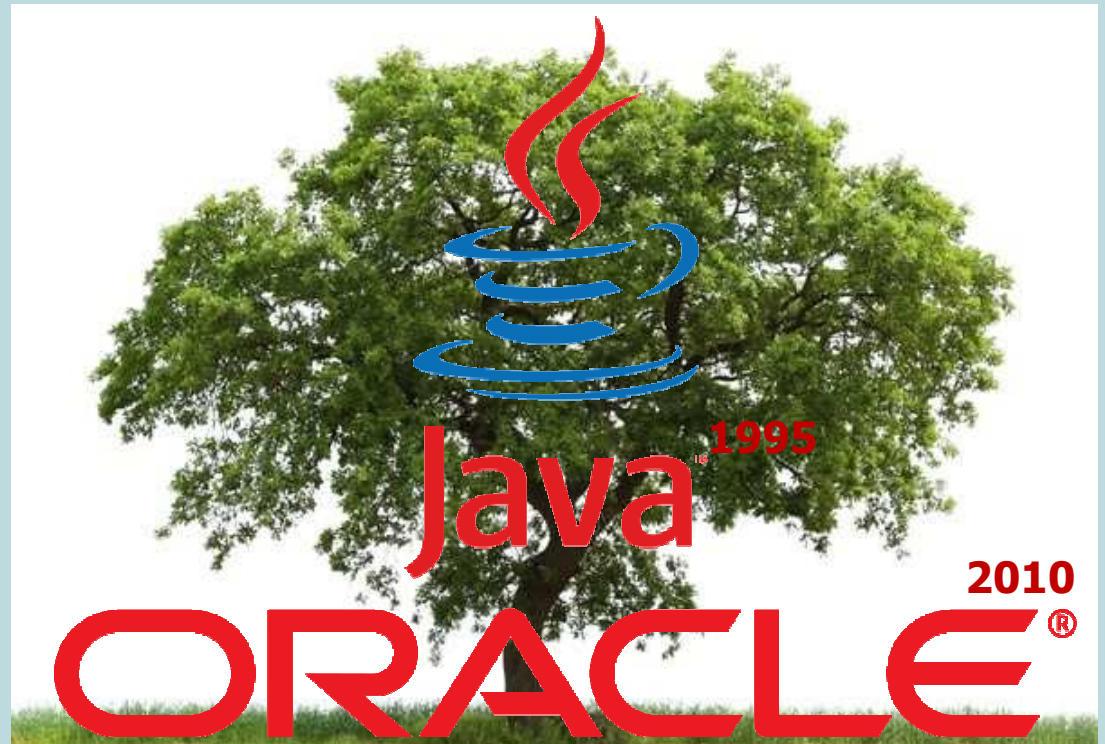
Nội dung chính

- 1. Lược sử phát triển Java**
- 2. Đặc điểm ngôn ngữ Java**
- 3. Cài đặt môi trường Java**
- 4. Kiểu dữ liệu**
- 5. Các phép toán**
- 6. Mảng, Chuỗi và Date**
- 7. Nhập xuất console**
- 8. Đọc ghi tập tin văn bản**

Lược sử phát triển Java



James Gosling – cha đẻ ngôn ngữ Java



Write Once, Run Anywhere

Lược sử phát triển Java

Phiên bản	Tên mã	Ngày phát hành
Java SE 1.0	Oak	21-01-1996
Java SE 1.1	-	19-02-1997
Java SE 1.2	Playground	08-12-1998
Java SE 1.3	Kestrel	08-05-2000
Java SE 1.4	Merlin	06-02-2002
Java SE 5	Tiger	30-09-2004
Java SE 6	Mustang	11-12-2006
Java SE 7	Dolphin	28-07-2011
Java SE 8	Spider	18-03-2014
Java SE 9	-	21-09-2017
Java SE 10	-	20-03-2018
Java SE 11		25-09-2018
Java SE 12		19-03-2019

Các phiên bản Java SE đã phát hành (nguồn Wikipedia)

Đặc điểm ngôn ngữ Java

Đơn giản

An toàn

Hướng đối tượng

Khả chuyển

Độc lập nền tảng

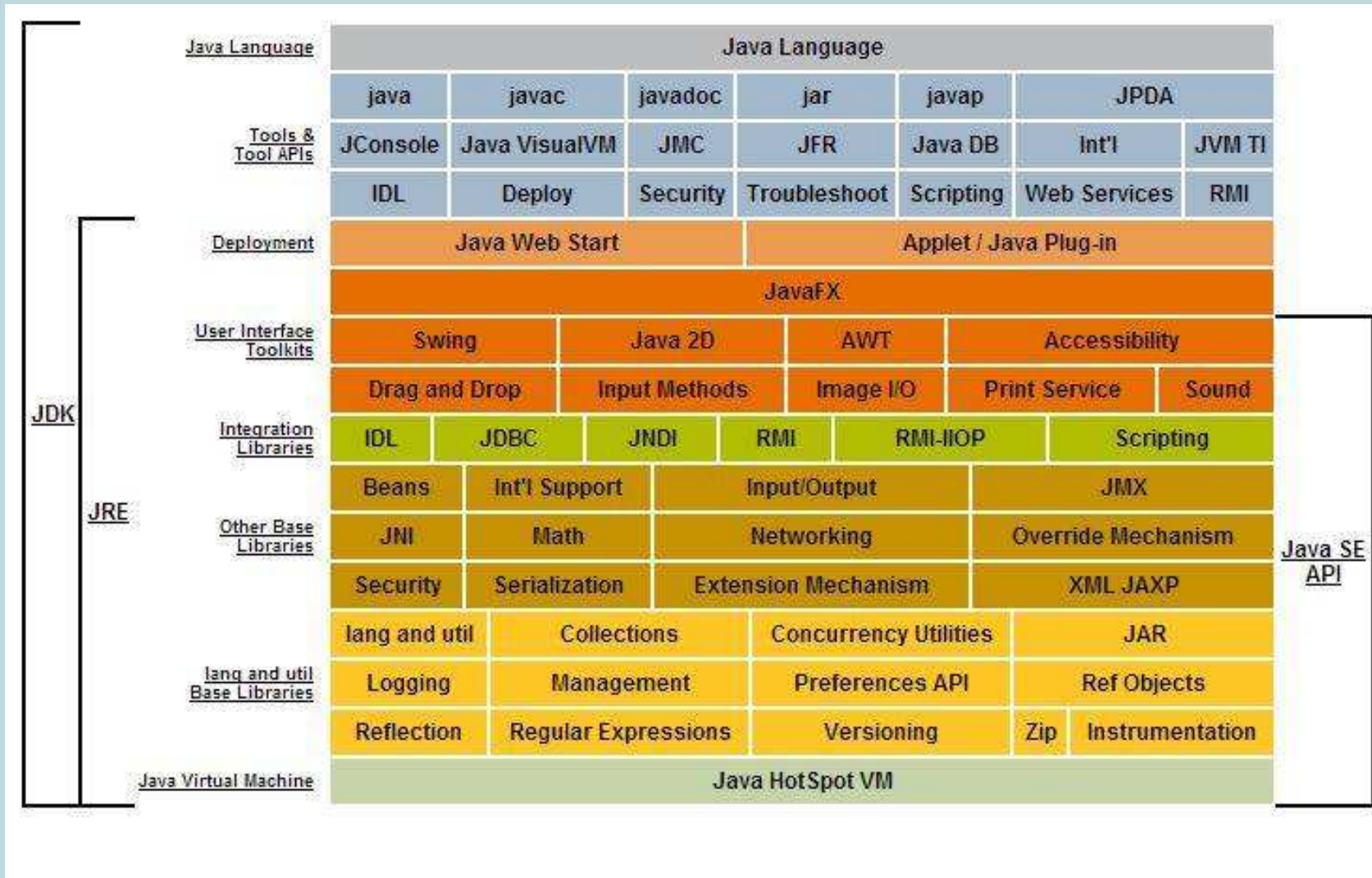
Đa luồng

Phân tán

Linh động

Mạnh mẽ

Cài đặt môi trường



Cài đặt môi trường

- **Cài JDK (Java Development Toolkit)**

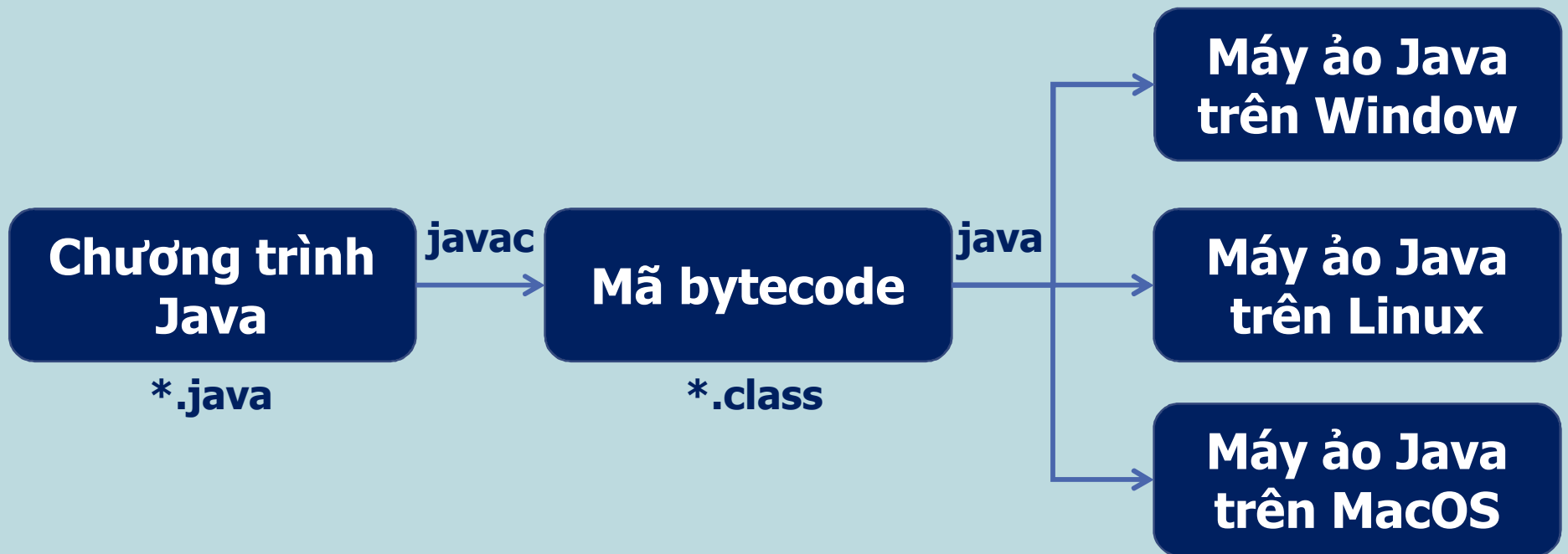
- <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

- **Cài đặt biến môi trường**

- <https://docs.oracle.com/javase/tutorial/essential/environment/paths.html>

Viết chương trình Java đầu tiên

- Minh họa bằng Notepad.
- Minh họa bằng NetBeans và Eclipse IDE.



Viết chương trình Java đầu tiên

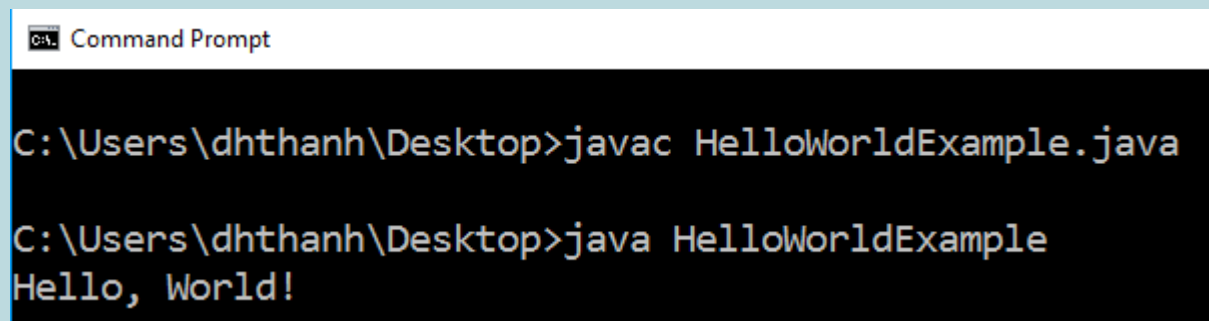
- Mở Notepad viết đoạn mã nguồn sau

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

- Lưu lại tập tin tên HelloWorldExample.java

Viết chương trình Java đầu tiên

- Mở Command Prompt (cmd) và chuyển (cd) đến thư mục chứa tập tin HelloWorld.java.
- Gõ lệnh **javac** để biên dịch thành bytecode.
- Gõ lệnh **java** để thực thi (thông dịch).



```
Command Prompt

C:\Users\dhthanh\Desktop>javac HelloWorldExample.java

C:\Users\dhthanh\Desktop>java HelloWorldExample
Hello, World!
```

Phát biểu nào sau đây sai?

- A. Java là ngôn ngữ vừa biên dịch, vừa thông dịch.**
- B. Java được xây dựng trên nền tảng của ngôn ngữ C/C++.**
- C. Java là ngôn ngữ vừa hỗ trợ lập trình thủ tục, vừa hỗ trợ lập trình hướng đối tượng.**
- D. Chương trình Java có thể chạy trên bất kỳ nền tảng nào mà không cần biên dịch lại.**

Phát biểu nào sau đây đúng?

- A. Lập trình viên Java phải chủ động giải phóng các vùng nhớ không sử dụng trước khi chương trình kết thúc.**
- B. Ngôn ngữ Java hỗ trợ khả năng đa kế thừa như trong C++**
- C. Trong Java, mọi thực thể hoạt động trong hệ thống đều là đối tượng.**
- D. Chương trình Java thực thi phụ thuộc trên từng nền tảng.**

Biến

- Bắt đầu bằng ký tự (a-z), (a-Z), _, \$
- Không bắt đầu bằng số (0-9), không chứa ký tự đặc biệt.
- Không trùng từ khoá.

KiểuDữLiệu **tênBiến** [= GiáTrịKhởiGán]

Kiểu cơ bản

Số nguyên
(**byte**, **int**, **long**)

Số thực
(**float**, **double**)

Ký tự
(**char**)

Luận lý
(**boolean**)

Kiểu tham chiếu

Array

String

Object

```
int soLuong;  
int dem = 1;  
double heSo = 1.67;  
String ketQua;
```

Biến

- Biến hằng

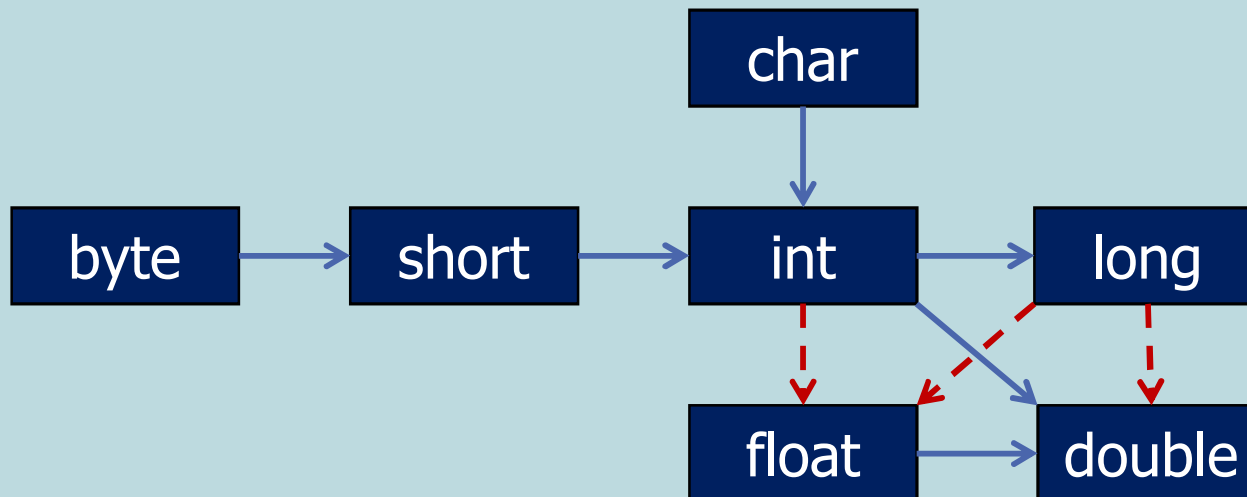
```
final KiểuDữLiệu TÊN_BIẾN = GiáTrịKhởiGán
```

```
final int SO_LUONG = 30;  
final double PI = 3.14;
```

Các phép toán

Nhóm phép toán	Ký hiệu	Ví dụ
Phép toán số học	$+$, $-$, $*$, $/$, $\%$	$5 \% 3 \rightarrow 2$ $5.0 / 2 \rightarrow 2.5$
Phép gán	$=$ $+=$, $-=$, $*=$, $/=$, $\%=$	<code>int a = 1;</code> <code>a++ + ++a</code> \rightarrow 4
Phép tăng, giảm	$++$, $--$	
Phép toán quan hệ	$==$, $!=$, $>$, $>=$, $<$, $<=$	$5 != 5 \rightarrow \text{false}$ $"5" != 5 \rightarrow \text{true}$
Phép luận lý	$\&\&$, $\ \ $, $!$	
Các hàm toán học	<code>Math.<tên-hàm>()</code>	<code>Math.abs(-5)</code> \rightarrow 5 <code>Math.sqrt(4)</code> \rightarrow 2 <code>Math.pow(2, 3)</code> \rightarrow 8

Chuyển đổi giữa các kiểu dữ liệu



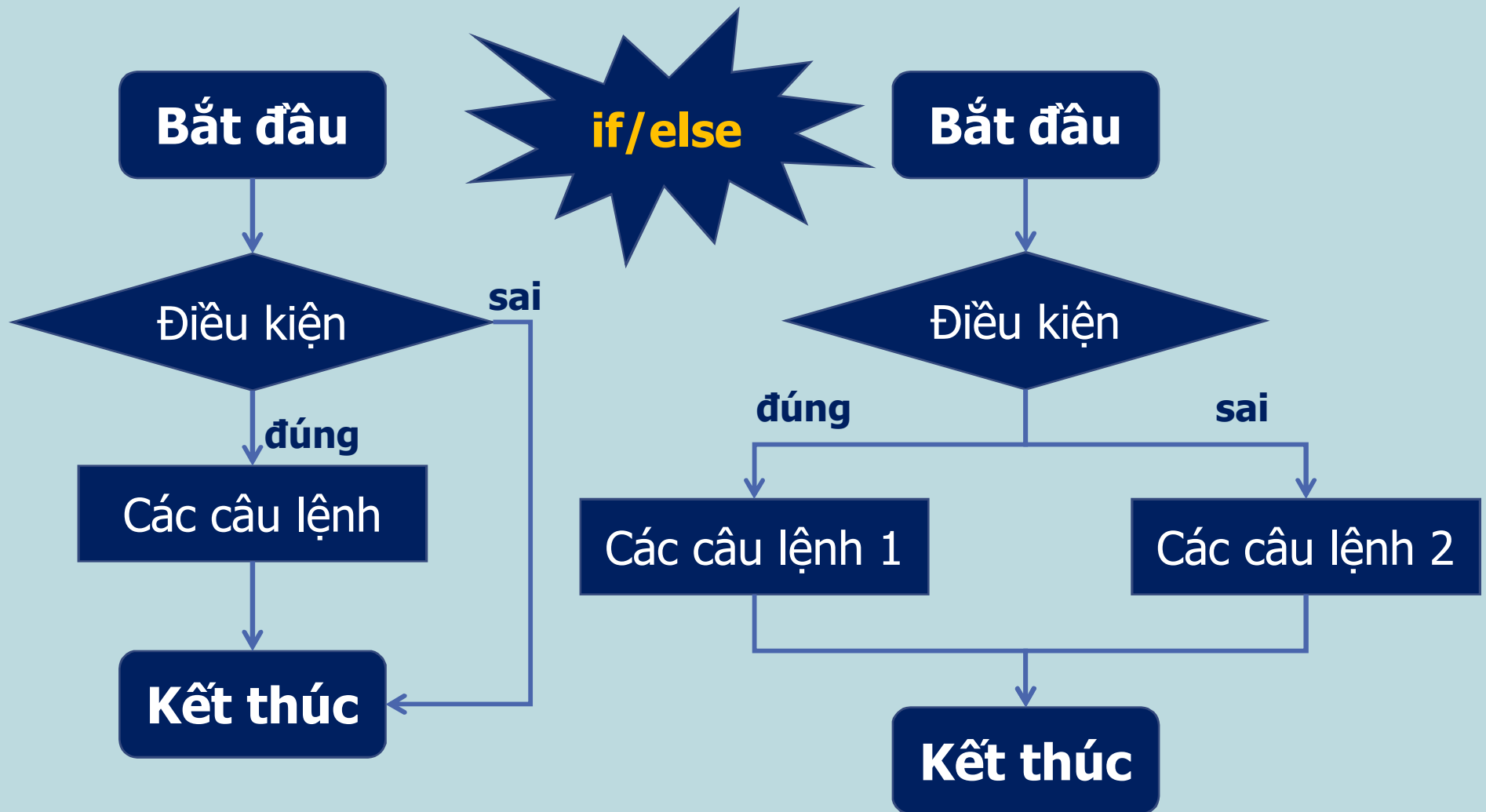
(KiểuDữLiệu) tênBiến;

—> *Phép chuyển đổi không mất thông tin*

- - > *Phép chuyển đổi có thể mất thông tin*

Ép kiểu

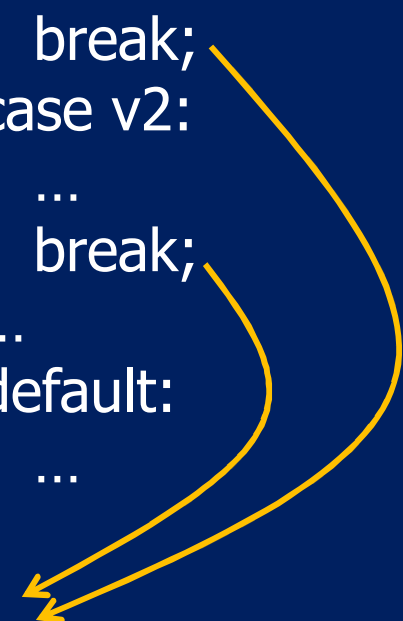
Cấu trúc điều khiển



Cấu trúc điều khiển

switch / case

```
switch (ĐiềuKiện) {  
  case v1:  
    ...  
    break;  
  case v2:  
    ...  
    break;  
  ...  
  default:  
    ...  
}  
...
```



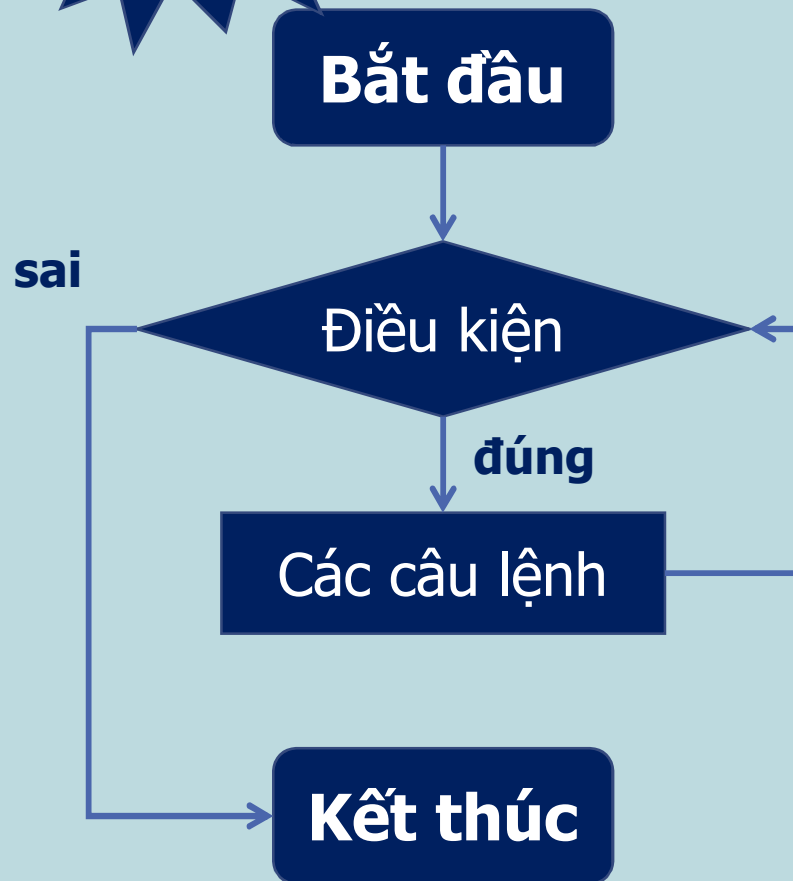
```
if (ĐiềuKiện là v1) {  
  ...  
} else if (ĐiềuKiện là v2) {  
  ...  
} ... else {  
  ...  
}
```

Kiểu dữ liệu không dùng trong biểu thức điều kiện lệnh switch?

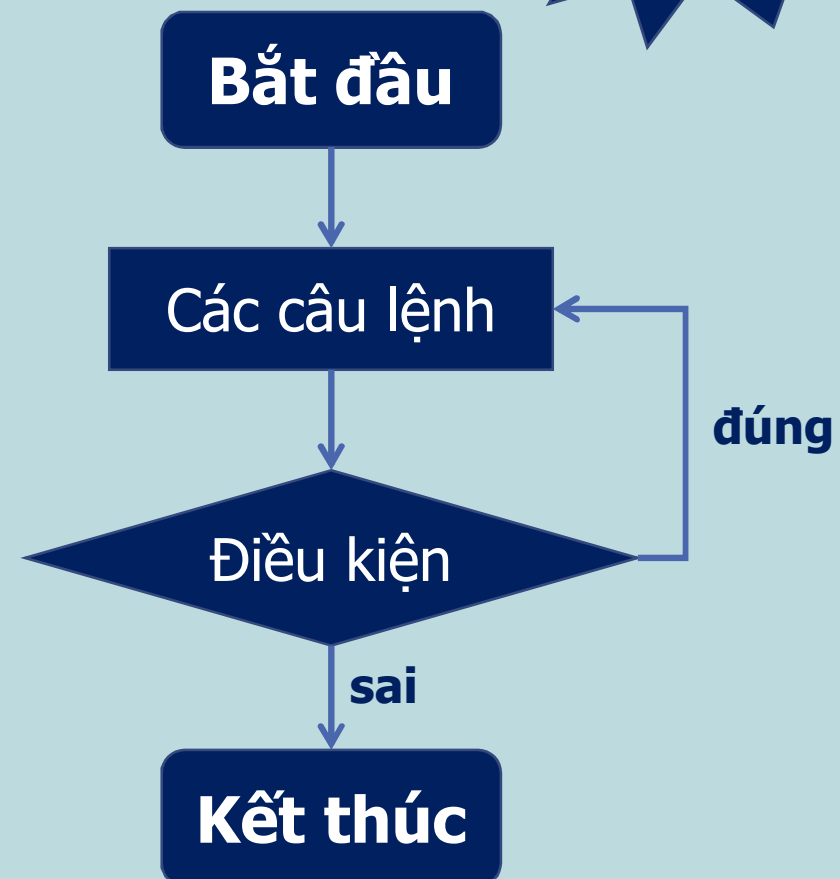
- A. byte**
- B. long**
- C. String**
- D. float**

Cấu trúc điều khiển

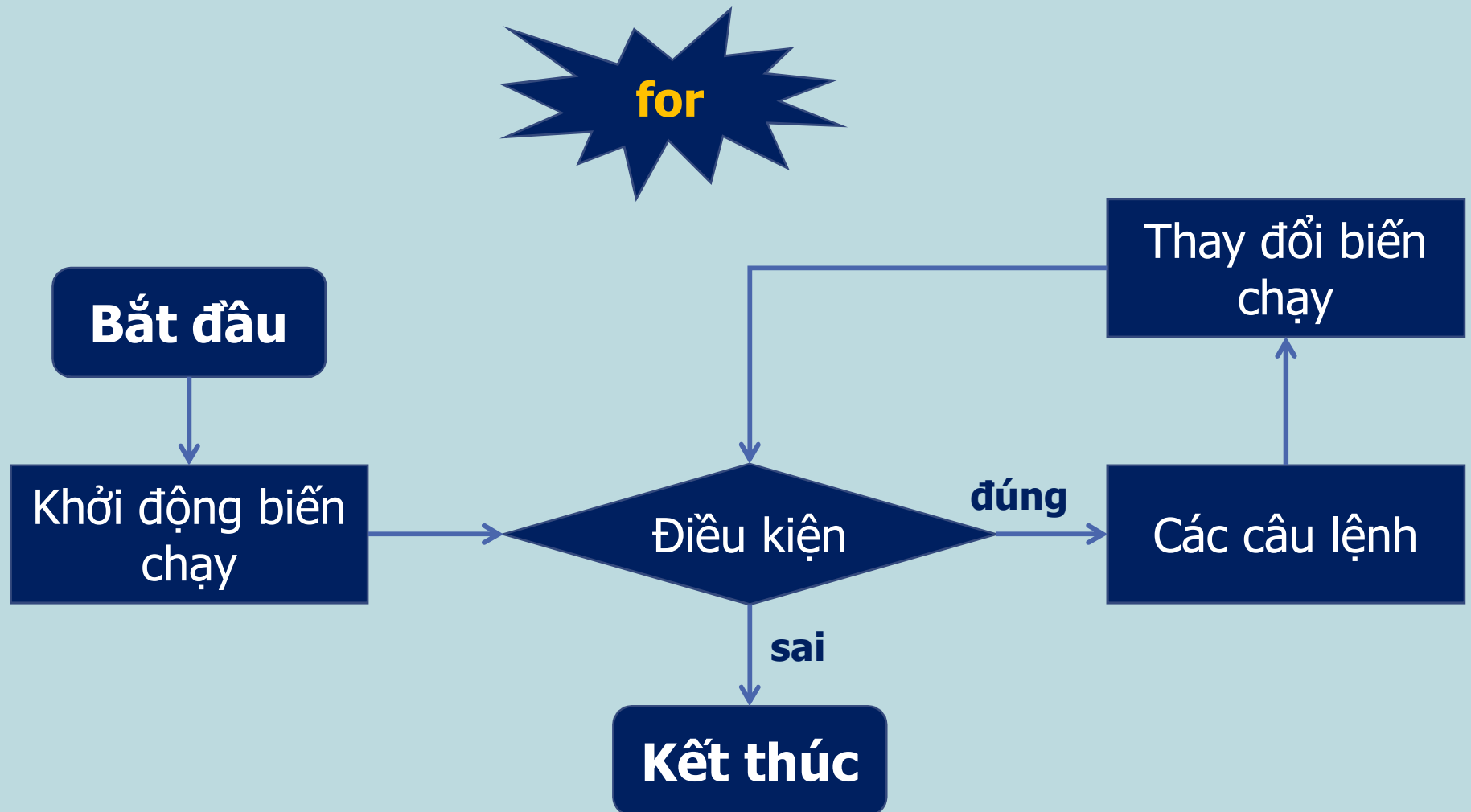
while



do/while




Cấu trúc điều khiển




Cấu trúc điều khiển

- **Lệnh break và continue**

```
...  
for (...; ...; ...) {  
    ...  
    break;  
    ...  
}  
...
```



```
...  
for (...; ...; ...) {  
    ...  
    continue;  
    ...  
}  
...
```



Chuỗi

- Chuỗi trong Java là một dãy các ký tự Unicode.
- Lớp **String** là lớp **immutable** dùng để thao tác với chuỗi trong Java.

```
String s1 = "Hello";  
String s2 = new String("Hello");
```

Một số thao tác trên chuỗi

Lấy chiều dài

length()

Lấy ký tự

charAt(index)

Chuyển đổi chuỗi

toUpperCase()

toLowerCase()

trim()

Một số thao tác trên chuỗi

Nối chuỗi

+

concat

So sánh chuỗi

equals

equalsIgnoreCase

compareTo

compareToIgnoreCase

startsWith

endsWith

contains

Một số thao tác trên chuỗi

Tìm chuỗi

→ `indexOf(char/string [, fromIndex])`

→ `lastIndexOf(char/string [, endIndex])`

Lấy chuỗi con

→ `substring(beginIndex [, endIndex])`

Cho biết kết quả đoạn chương trình?

```
String s = "thanh.dh@ou.edu.vn";  
System.out.println(s.substring(0, s.indexOf("@")));
```

- A. thanh.dh@**
- B. ou.edu.vn**
- C. @ou.edu.vn**
- D. thanh.dh**

Một số thao tác trên chuỗi

Tách chuỗi

→ `split(delimiter: String)`

Thay thế chuỗi

→ `replace(oldChar, newChar)`

→ `replaceFirst(oldString, newString)`

→ `replaceAll(oldString, newString)`

Một số thao tác trên chuỗi

Chuyển kiểu dữ liệu cơ bản thành chuỗi

String.valueOf(...)

Chuyển chuỗi thành kiểu dữ liệu khác

Wrapper class

Integer

Double

...

Một số thao tác trên chuỗi

Định dạng chuỗi

`String.format(ChuỗiĐịnhDạng, v1, v2, ...)`

```
String.format("%7.2f%5d%-3s", 45.3333, 12, "A");
```

		4	5	.	3	3				1	2	A		
--	--	---	---	---	---	---	--	--	--	---	---	---	--	--

StringBuilder

```
StringBuilder builder = new StringBuilder();  
  
// Nối một ký tự  
builder.append('A');  
  
// Nối chuỗi  
builder.append(" great method!");  
  
// Lấy về chuỗi  
String result = builder.toString();
```

Mảng một chiều

- Khai báo mảng

KiểuDữLiệu[] tênMảng = **new** KiểuDữLiệu[KíchThước]

KiểuDữLiệu tênMảng[] = **new** KiểuDữLiệu[KíchThước]

- Khởi gán

```
int[] a = {15, 20, 25};  
String[] b = {"Happy", "New", "Year"};
```


Phát biểu nào sau đây sai?

- A. Các phần tử trong mảng phải có cùng kiểu dữ liệu.
- B. Java luôn sử dụng truyền trị (pass-by-value) để truyền các đối số vào phương thức.
- C. Java sử dụng truyền tham chiếu (pass-by-reference) khi truyền kiểu mảng vào phương thức.
- D. Mảng một khi đã tạo thì kích thước của nó là cố định.

Một số thao tác trên mảng

```
int[] a = {4, 5, 6, 8, 3};  
  
System.out.println("Chiều dài: " + a.length);  
  
int[] b = Arrays.copyOf(a, a.length);  
for (int i: b)  
    System.out.printf("%d ", i);  
System.out.println();  
  
Arrays.sort(a);  
for (int i: a)  
    System.out.printf("%d ", i);
```

Một số thao tác với mảng

Lấy chiều dài mảng

→ `TênMảng.length`

Sao chép mảng

→ `Arrays.copyOf(arr, len);`

→ `System.arraycopy(...)`

Sắp xếp mảng

→ `Arrays.sort()`

Cho biết kết quả đoạn chương trình?

```
int[] number = {1, 2, 3};  
int[] copiedNumber = number;  
copiedNumber[1] = 9;  
System.out.println(number[1]);
```

A. 2

B. 9

C. 1

D. 3

Date

- Để tạo đối tượng ngày tháng sử dụng các phương thức khởi tạo của lớp **Date**.
 - `new Date()`
 - `new Date(milliseconds)`
- Ví dụ

```
Date d1 = new Date();  
// Số mili giây từ 01/01/1970 00:00:00 -> hiện tại  
System.out.println("Số mili giây: " + d1.getTime());
```

```
Date d2 = new Date(System.currentTimeMillis()-500);  
System.out.println(d2.toString());
```

Một số thao tác trên Date

So sánh ngày tháng

before

after

equals

compareTo

Một số thao tác trên Date

- Định dạng date dùng SimpleDateFormat

```
Date d = new Date();  
SimpleDateFormat f  
    = new SimpleDateFormat("dd/mm/yyyy 'at' hh:mm:ss");  
System.out.println(f.format(d));
```

- Chuyển chuỗi thành date

```
SimpleDateFormat f = new  
SimpleDateFormat("dd/mm/yyyy");  
Date d = f.parse("09/04/2019");  
System.out.printf("Số mili giây: %d\n", d.getTime());
```

Lớp GregorianCalendar

- **Lớp GregorianCalendar là lớp con Calendar**

```
GregorianCalendar c = new GregorianCalendar();
System.out.println(c.get(Calendar.YEAR));
System.out.println(c.get(Calendar.MONTH));
System.out.println(c.get(Calendar.DAY_OF_MONTH));
System.out.println(c.get(Calendar.HOUR));
System.out.println(c.get(Calendar.MINUTE));
System.out.println(c.get(Calendar.SECOND));
System.out.println(c.get(Calendar.AM_PM));

if (c.isLeapYear(2020))
    System.out.println("Năm nhuận");

c.add(Calendar.DAY_OF_MONTH, 1);
System.out.println("Ngày sau: " + c.getTime());
```


Nhập xuất console

- Java dùng `System.out` và `System.in` cho thiết bị nhập/xuất chuẩn.
- Để xuất ra console dùng phương thức `print` hoặc `println` của `System.out`.
- Java không trực tiếp hỗ trợ đọc từ console, nhưng dùng lớp `java.util.Scanner` tạo đối tượng đọc dữ liệu từ `System.in`.

Nhập xuất console

```
System.out.print("Nhap n = ");
Scanner scanner = new Scanner(System.in);
int n = scanner.nextInt();

int dem = 0;
while (n != 0) {
    if ((n % 10) % 2 != 0)
        dem++;
    n /= 10;
}

System.out.println("So chu so le cua n: " + dem);
```

Đọc ghi tập tin văn bản

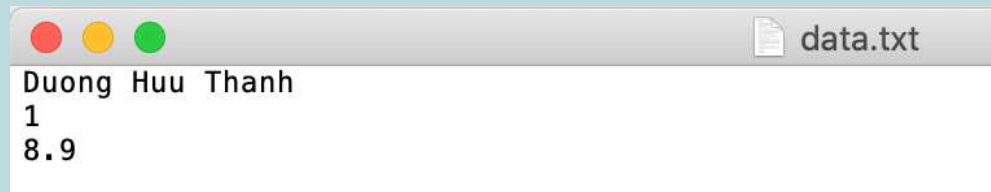
- Sử dụng `java.util.Scanner` để đọc tập tin văn bản.

```
File f = new File(<filename>);  
Scanner scanner = new Scanner(f);
```

- Ngoại lệ `FileNotFoundException` được ném ra nếu tập tin không tồn tại.
- Sử dụng `next`, `nextInt`, `nextDouble` để đọc dữ liệu tương ứng.
- Sử dụng `close` đóng tập tin khi không dùng.

Đọc ghi tập tin văn bản

▪ Ví dụ



Tập tin data.txt

```
File f = new File("data.txt");

try (Scanner scanner = new Scanner(f)) {
    String hoTen = scanner.nextLine();
    int mssv = scanner.nextInt();
    double diemTB = scanner.nextDouble();
    System.out.printf("MSSV: %d\nHo ten: %s\n" +
        "Diem TB: %.2f\n", mssv, hoTen, diemTB);
}
```

Đọc ghi tập tin văn bản

- Sử dụng lớp `java.io.PrintWriter` để ghi tập tin văn bản

```
File f = new File(<filename>);  
PrintWriter p = new PrintWriter(f);
```

- Nếu tập tin không tồn tại thì nó sẽ được tạo ra, ngược lại thì nội dung tập tin sẽ bị huỷ.
- Sử dụng `print`, `println` để ghi dữ liệu.
- Sử dụng `close` đóng tập tin khi không dùng.

Đọc ghi tập tin văn bản

▪ Ví dụ

```
File f = new File("data.txt");

try (PrintWriter writer = new PrintWriter(f)) {
    writer.println("Duong Huu Thanh");
    writer.println(1);
    writer.println(8.9);
}
```

Đọc ghi tập tin văn bản

- Để mở tập tin ghi tiếp sử dụng **FileWriter**

```
File f = new File("data.txt");  
FileWriter w = new FileWriter(f, true);  
  
try (PrintWriter writer = new PrintWriter(w)) {  
    writer.println(Math.random());  
}
```

