CHƯƠNG II - NỀN TẢNG CỦA JAVA

I - GIỚI THIỆU VỀ JAVA

✓Java là một ngôn ngữ lập trình được Sun Microsystems giới thiệu vào tháng 6 năm 1995, sử dụng các cú pháp của C và các đặc trưng hướng đối tượng của C++.

✓ Nhu cầu thực tế đòi hỏi một ngôn ngữ chạy nhanh, gọn, hiệu quả và độc lập thiết bị tức là có thể chạy trên nhiều loại CPU khác nhau, dưới các môi trường khác nhau. “Oak” đã ra đời và vào năm 1995 được đổi tên thành Java.

✓Java là ngôn ngữ lập trình hướng đối tượng, có thể giải quyết hầu hết các công việc mà các ngôn ngữ khác có thể làm được.

✓ Java vừa biên dịch vừa thông dịch. Mục tiêu là cho phép viết chương trình một lần nhưng có thể chạy trên bất cứ phần cứng cụ thể, độc lập thiết bị, không phụ thuộc vào hệ điều hành và để viết chương trình chạy trên Internet.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

II - MÁY ẢO JAVA

✓ Máy ảo Java là trái tim của ngôn ngữ Java. Môi trường Java bao gồm năm phần tử sau:

- Ngôn ngữ
- Định nghĩa Bytecode
- Các thư viện lớp Java/Sun
- Máy ảo Java (JVM)
- Cấu trúc của file .class

✓ Các phần tử tạo cho Java thành công là

- Định nghĩa Bytecode
- Cấu trúc của file .class
- Máy ảo Java (JVM)

✓ Khả năng cơ động của file .class cho phép các chương trình Java viết một lần nhưng chạy ở bất kỳ đâu. Khả năng này có được nhờ sự giúp đỡ của máy ảo Java.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

1 - Khái niệm về máy ảo JAVA

✓ Máy ảo là một phần mềm dựa trên cơ sở máy tính ảo. Nó có tập hợp các lệnh logic để xác định các hoạt động của máy tính. Có thể xem nó như một hệ điều hành thu nhỏ.

✓ Trình biên dịch chuyển mã nguồn thành tập các lệnh của máy ảo mà không phụ thuộc vào phần cứng cụ thể. Trình thông dịch trên mỗi máy sẽ chuyển tập lệnh này thành chương trình thực thi. Máy ảo tạo ra một môi trường bên trong để thực thi các lệnh bằng cách:

- Nạp các file .class
- Quản lý bộ nhớ
- Dọn "rác"

✓ Một khái niệm thông dụng khác trong Java là trình biên dịch "**Just In Time** - JIT". Mục đích chính của JIT là chuyển tập lệnh bytecode thành mã máy cụ thể cho từng loại CPU. Các lệnh này sẽ được lưu trữ và sử dụng mỗi khi gọi đến.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

2 - Bộ công cụ phát triển JDK (Java Development Kit)

Sun Microsystem đưa ra ngôn ngữ lập trình Java qua sản phẩm có tên là Java Development Kit (JDK). Ba phiên bản chính là:
Java 1.0 - Sử dụng lần đầu vào năm 1995
Java 1.1 – Đưa ra năm 1997 vớI nhiều ưu điểm hơn phiên bản trước.
Java 2 –
JDK chứa các công cụ sau:

2.1 Trình biên dịch, 'javac'

2.2 Trình thông dịch, 'java'

2.3 Trình dịch ngược, 'javap'

javap dịch ngược bytecode và in ra thông tin về các thuộc tính (các trường), các phương thức của một lớp.

2.4 Công cụ sinh tài liệu, 'javadoc'

Tiện ích này cho phép ta tạo ra tệp HTML dựa trên các lời giải thích trong mã chương trình (phần nằm trong cặp dấu /*.... */).

2.5 Chương trình tìm lỗi - Debug, 'jdb'

2.6 Chương trình xem Applet , 'appletviewer'

CHƯƠNG II - NỀN TẢNG CỦA JAVA

II - CẤU TRÚC MỘT CHƯƠNG TRÌNH JAVA

✓ Phần đầu của một chương trình Java xác định thông tin môi trường, chương trình được chia thành các lớp hoặc các gói riêng biệt.

✓Sử dụng lệnh "import" để nhập các gói.

✓Mỗi chương trình có thể có nhiều hơn một lệnh nhập. VD:

import java. awt.*;

✓Lệnh này nhập gói 'awt'. Gói này dùng để tạo các đối tượng GUI. Java là tên của thư mục chứa gói 'awt'. Ký hiêu "*" chỉ tất cả các lớp thuộc gói này.

✓Trong java, tất cả các mã, bao gồm các biến và các khai báo nên được thực hiện trong phạm vi một lớp. Bởi vậy, từng khai báo lớp được tiến hành sau lệnh nhập.

✓Một chương trình đơn giản có thể chỉ có một vài lớp. Những lớp này có thể mở rộng thành các lớp khác.

✓Mỗi lệnh đều được kết thúc bởi dấu chấm phảy ";".

✓Chương trình còn có thể bao gồm các ghi chú, chỉ dẫn. Khi dịch, chương trình dịch sẽ tự loại bỏ các ghi chú này.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

II - CẤU TRÚC MỘT CHƯƠNG TRÌNH JAVA

Dạng cơ bản của một lớp được xác định như sau :

```
class classname
{
/* Đây là dòng ghi chú*/
int num1,num2;   // Khai báo biến với các dấu phảy giữa các biến
        Show()
        {
// Method body
statement (s); // Kết thúc bởi dấu chấm phảy
}
}
```

1 - Viết chương trình đầu tiên

```
// This is a simple program called "First.java"
class First
{
        public static void main(String args[])
        {
System.out.println("My first program in Java");
}
}
```

CHƯƠNG II - NỀN TẢNG CỦA JAVA

1- Một vài quy định:

✓Tên lớp và tên file phải trùng nhau.

✓Java phân biệt chữ hoa và chữ thường. Ví dụ tên file 'First' # 'first'

✓Để biên dịch mã nguồn, ta sử dụng trình biên dịch java.

```
C:\jdk1.2.1\bin>javac First.java
```

✓Trình dịch java tạo ra file First.class chứa các mã "bytecodes". Để chương trình thực thi được ta cần dùng "java interpreter"

```
C:\jdk1.2.1\bin>java First
```

✓Kết quả sẽ là:

```
My first program in Java
```

2- Giải thích chương trình:

✓Ký hiệu "// " dùng để thuyết minh dòng lệnh. Trình biên dịch sẽ bỏ qua dòng thuyết minh này (có thể bắt đầu với /* và kết thúc với */)

✓**class First:** Khai báo lớp có tên 'First' (cũng chính là tên file).

✓Tên lớp nói chung nên bắt đầu bằng chữ in hoa.

✓Từ khoá 'class' khai báo định nghĩa lớp được đặt giữa hai dấu ({) và (})

CHƯƠNG II - NỀN TẢNG CỦA JAVA

✓**public static void main(String args[ ])**

- Đây là phương thức chính để chương trình bắt đầu việc thực thi của mình.
- 'public' là một chỉ định truy xuất. Nó cho biết thành viên của lớp có thể được truy xuất từ bất cứ đâu trong chương trình. Phương thức **"main"** được khai báo 'public', bởi vậy JVM có thể truy xuất phương thức này.

✓ **'static'** cho phép **main** được gọi tới mà không cần tạo ra một thể hiện (instance) của lớp. Điều này rất quan trọng vì JVM trước tiên gọi phương thức **main** để thực thi chương trình.Nó không phụ thuộc vào các thể hiện của lớp được tạo ra.

✓ **'void'** thông báo phương thức sẽ không trả lại bất cứ giá trị nào khi thực thi chương trình.

✓Phương thức **'main()'** sẽ thực hiện một số tác vụ nào đó, nó là điểm mốc mà từ đó tất cả các ứng dụng Java được khởi động.

✓ **'String args[]'** là tham số dùng trong phương thức **'main'**. Các biến số trong dấu ngoặc đơn nhận từng thông tin được chuyển vào **'main'**. **'args[]'** là một mảng kiểu "String".

✓***System.out.println("My first program in Java");***

Dòng lệnh này hiển thị chuỗi "My first program in Java" trên màn hình.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

3- Khai báo lớp

Cú pháp:

class classname
{ var_datatype variablename;
:
met_datatype methodname(parameter_list)
:
}

Trong đó:

class - Từ khoá xác định lớp
classname - Tên của lớp
var_datatype - kiểu dữ liệu của biến
variablename - Tên của biến
met_datatype - Kiểu dữ liệu trả về của phương thức
methodname - Tên của phương thức
parameter_list – Các tham số được của phương thức

CHƯƠNG II - NỀN TẢNG CỦA JAVA

III – CÁC KIỂU DỮ LIỆU CỦA JAVA:

1 - Các kiểu dữ liệu nguyên thủy: Tám kiểu

Kiểu dữ liệu	Độ dài theo bit	Phạm vi biểu diễn giá trị	Mô tả
byte	8	-128 đến 127	Sử dụng khi xử lý một file văn bản
char	16	'\u0000' to 'u\ffff '	Ví dụ tên người
boolean	1	"True" hoặc "False"	Dùng để lưu các giá trị "Đúng" hoặc "sai"
short	16	-32768 đến 32767	Dùng để lưu các số có giá trị nhỏ dưới 32767.
Int	32	-2,147,483,648 đến +2,147,483,648	Dùng để lưu một số có giá trị lớn đến 2,147,483,648.
long	64	-9,223,372,036'854,775,808 đến +9,223,372,036'854,775,808	Ví dụ dân số của một nước
float	32	-3.40292347E+38 đến +3.40292347E+38	Ví dụ : giá thành sản phẩm
double	64	-1,79769313486231570E+308 đến +1,79769313486231570E+308	Ví dụ giá trị tín dụng của ngân hàng nhà nước.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

III – CÁC KIỂU DỮ LIỆU CỦA JAVA:

2 - Các kiểu dữ liệu tham chiếu: Ba kiểu

Kiểu dữ liệu	Mô tả
Mảng (Array)	Tập hợp các dữ liệu cùng kiểu. Ví dụ : tên sinh viên
Lớp (Class)	Tập hợp các biến và các phương thức.Ví dụ : lớp "Sinhviên" chứa toàn bộ các chi tiết của một sinh viên và các phương thức thực thi trên các chi tiết đó.
Giao diện (Interface)	Là một lớp trừu tượng được tạo ra cho phép cài đặt đa thừa kế trong Java.

3 – Ép kiểu :

Là t.hợp một kiểu dữ liệu sẽ chuyển đổi sang kiểu khác. VD:

float c=34.896751f;

Int b = (int)c +10;

Đầu tiên giá trị dấu phảy động **c** được đổi thành giá trị nguyên 34. Sau đó nó được cộng với 10 và kết quả là giá trị 44 được lưu vào **b.**

CHƯƠNG II - NỀN TẢNG CỦA JAVA

III – CÁC KIỂU DỮ LIỆU CỦA JAVA:

4 – Khai báo biến

- ✓ Gồm 3 thành phần: kiểu biến, tên của nó và giá trị ban đầu được gán cho biến (không bắt buộc).
- ✓ Sử dụng dấu phẩy để phân cách các biến.
- ✓ Java phân biệt chữ thường và chữ in hoa (case -sensitive).

Cú pháp:

Datatype indentifier [=value] [, indentifier[=value]...];

VD: int counter = 1;

5 – Khai báo mảng:

Mảng có thể được khai báo bằng ba cách :

CHƯƠNG II - NỀN TẢNG CỦA JAVA

Cách khai báo	Mô tả	Cú pháp	Ví dụ
Chỉ đơn thuần khai báo	Chỉ đơn thuần khai báo mảng	Datatype identifier[]	**char ch[ ];**khai báo mảng ký tự có tên **ch**
Khai báo và tạo mảng	Khai báo và cấp phát bộ nhớ cho các phần tử mảng sử dụng toán tử "new'	Datatype identifier[] =new datatype [size]	**char ch[] = new char [10];**Khai b áo một mảng **ch** và lưu trữ 10 ký tự
Khai báo,kiến tạo và khởi tạo	Khai báo mảng,cấp phát bộ nhớ cho nó và gán các giá trị ban đầu cho các phần tử của mảng	Datatype identifier[] = {value1,value2…valueN };	**char ch []** = {'A','B','C','D' }; khai báo mảng **ch** và lưu 4 chữ cái kiểu ký tự

CHƯƠNG II - NỀN TẢNG CỦA JAVA

III – CÁC KIỂU DỮ LIỆU CỦA JAVA:

6- Phương thức trong một lớp:

Phương thức được định nghĩa như một hành động hoặc một tác vụ thật sự của đối tượng. Nó còn được định nghĩa như một hành vi mà trên đó các thao tác cần thiết được thực thi.

Cú pháp

```
access_specifier modifier datatype method_name(parameter_list)
{   //body of method
}
```

Trong đó:

access_specifier: Chỉ định truy cập vào phương thức.

modifier: Cho phép bạn đặt thuộc tính cho phương thức.

datatype: Kiểu dữ liệu mà phương thức trả về. Nếu không có một giá trị nào được trả về, kiểu dữ liệu có thể là **void.**

method_name: Tên của phương thức

parameter_list: Chứa tên của tham số được sử dụng trong phương thức và kiểu dữ liệu. Dấu phẩy được dùng để phân cách các tham số.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

III – CÁC KIỂU DỮ LIỆU CỦA JAVA:

VD: khai báo 2 phương thức là Show() và Main()

```
class Temp
{   static int x=10;//variable
public static void show()//method
{ System.out.println(x);}
public static void main(String args[])
{  Temp t = new Temp();// object 1
t.show();//method call
Temp t1=new Temp();// object 2
t1.x=20;
t1.show();
}
}
```

CHƯƠNG II - NỀN TẢNG CỦA JAVA

III – CÁC KIỂU DỮ LIỆU CỦA JAVA:

7- Các chỉ định truy xuất của phương thức

Các chỉ định truy xuất dùng để giới hạn khả năng truy nhập vào một phương thức. Java cung cấp các chỉ định truy xuất sau đây:

- **Công cộng (Public)**: Phương thức có chỉ định truy xuất public có thể được nhìn thấy từ mọi gói hoặc mọi lớp.
- **Bảo vệ (Protected):** Các lớp mở rộng từ lớp hiện hành trong cùng một gói, hoặc tại các gói khác nhau có thể truy cập các phương thức loại này.
- **Riêng tư (Private):** Phương thức riêng tư chỉ có thể được truy cập nhờ phương thức công cộng trong cùng một lớp.

CHƯƠNG II - NỀN TẢNG CỦA JAVA

8 - Các toán tử

Java cung cấp nhiều dạng toán tử sau:

- Toán tử số học
- Toán tử bit
- Toán tử quan hệ
- Toán tử logic
- Toán tử điều kiện
- Toán tử gán

a) Các toán tử số học

CHƯƠNG II - NỀN TẢNG CỦA JAVA

+	Cộng: Trả về giá trị tổng hai toán hạngVí dụ 5+3 trả về kết quả là 8
-	Trừ: Trả về giá trị khác nhau giữa hai toán hạng hoặc giá trị phủ định của toán hạng. Ví dụ 5-3 kết quả là 2 và –10 trả về giá trị âm của 10
*	Nhân: Trả về giá trị là tích hai toán hạng. Ví dụ 5*3 kết quả là 15
/	Chia: Trả về giá trị là thương của phép chia
%	Phép lấy modulo: Giá trị trả về là phần dư của phép chia
++	Tăng dần: Tăng giá trị của biến lên 1. Ví dụ a++ tương đương với a= a+1
- -	Giảm dần: Giảm giá trị của biến 1 đơn vị. Ví dụ a-- tương đương với a=a-1
+–	Cộng và gán giá trị: Cộng các giá trị của toán hạng bên trái vào toán hạng bên phải và gán giá trị trả về vào

CHƯƠNG II - NỀN TẢNG CỦA JAVA

-=	Trừ và gán giá trị : Trừ các giá trị của toán hạng bên trái vào toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái.Ví dụ c-= a tương đương vớl c=c-a
*=	Nhân và gán: Nhân các giá trị của toán hạng bên trái với toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái.Ví dụ c *= a tương đương với c=c*a
/=	Chia và gán: Chia giá trị của toán hạng bên trái cho toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c /= a tương đương với c=c/a
%=	Lấy số dư và gán: Chia giá trị của toán hạng bên trái cho toán toán hạng bên phải và gán giá trị **số dư** vào toán hạng bên trái. Ví dụ c%=a tương đương với c=c%a

CHƯƠNG II - NỀN TẢNG CỦA JAVA

b) Các toán tử quan hệ

>	Lớn hơn: Kiểm tra giá trị của toán hạng bên phải lớn hơn toán hạng bên trái hay không Ví du if(a>b) . Trả về giá trị “true” nếu a lớn hơn b,ngược lai (nhỏ hơn hoặc bằng), trả về ‘False’
<	Nhỏ hơn: Kiểm tra giá trị của toán hạng bên phải có nhỏ hơn toán hạng bên trái hay không. Ví du if(a<b) . Trả về giá trị “true” nếu a nhỏ hơn b , ngược lại (lớn hơn hoặc bằng trả về ‘False’
>=	Lớn hơn hoặc bằng: Kiểm tra giá trị của toán hạng bên phải có lớn hơn hoặc bằng toán hạng bên trái hay không. Ví du if(a>=b) . Trả về giá trị “true” nếu a lớn hơn hoặc bằng b , ngược lại (nhỏ hơn trả về ‘False’
<=	Nhỏ hơn hoặc bằng: Kiểm tra giá trị của toán hạng bên phải có nhỏ hơn hoặc bằng toán hạng bên trái hay không. Ví du if(a<=b) . Trả về giá trị “true” nếu a nhỏ hơn hoặc bằng b , ngược lại (lớn hơn trả về ‘false’)

CHƯƠNG II - NỀN TẢNG CỦA JAVA

==	So sánh bằng: Toán tử này kiểm tra sự tương đương của hai toán hạng Ví dụ **if (a= =b)** trả về giá tri "True" nếu giá trị của a và b như nhauSo sánh khác: Kiểm tra sự khác nhau của hai toán hạngVí dụ **if(a!=b)** Trả về giá trị "true" nếu a khác b
!=	So sánh khác: Kiểm tra sự khác nhau của hai toán hạng Ví dụ **if(a!=b)** Trả về giá trị "true" nếu a khác b

CHƯƠNG II - NỀN TẢNG CỦA JAVA

IV – ĐIỀU KHIỂN LUỒNG:

Điều khiển luồng cho phép người phát triển phần mềm kiểm tra sự tồn tại của một điều kiện nào đó và ra quyết định phù hợp với điều kiện đó.

Điều khiển rẽ nhánh

- ✓ Mệnh đề if-else
- ✓ Mệnh đề swich-case

Vòng lặp (Loops)

- ✓ Vòng lặp while
- ✓ Vòng lặp do-while
- ✓ Vòng lặp for

CHƯƠNG II - NỀN TẢNG CỦA JAVA

IV – ĐIỀU KHIỂN LUỒNG:

1- Câu lệnh if - else

Cú pháp

If (conditon)
{ action 1 statements; }
else
{ action 2 statements; }

***Condition**: Biểu thức Boolean như toán tử so sánh. Biểu thức này trả về giá trị True hoặc False*

***action 1**: Các dòng lệnh được thực thi khi giá trị trả về là True*

***else**: Từ khoá xác định các câu lệnh tiếp sau được thực hiện nếu điều kiện trả về giá trị False*

***action 2:** Các câu lệnh được thực thi nếu điều kiện trả về giá trị False*

CHƯƠNG II - NỀN TẢNG CỦA JAVA

IV – ĐIỀU KHIỂN LUỒNG:

1- Câu lệnh if – else

Ví dụ:

```
class CheckNumber
{
public static void main(String args[]
{
int num =10;
if(num %2 == 0)
System.out.println (num+ "is an even
   number");
else
System.out.println (num +"is an odd number");
}
}
```

CHƯƠNG II - NỀN TẢNG CỦA JAVA

2- Câu lệnh switch - case

switch-case được sử dụng trong tình huống một biểu thức cho ra nhiều kết quả. Việc sử dụng câu lệnh switch-case cho phép việc lập trình dễ dàng và đơn giản hơn.

Cú pháp

```
swich (expression)
{
case ‘value’:action 1 statement;
break;
case ‘value’:action 2 statement;
break;
:
:
case ‘valueN’: actionN statement; break;
default: default_action statement;
}
```

CHƯƠNG II - NỀN TẢNG CỦA JAVA

2- Câu lệnh switch - case

- expession - Biến chứa một giá trị xác định
- value1,value 2,….valueN: Các giá trị hằng số phù hợp với giá trị trên biến expression .
- action1,action2…actionN: Các phát biểu được thực thi khi một trường hợp tương ứng có giá trị True
- break: Từ khoá được sử dụng để bỏ qua tất cả các câu lệnh sau đó và giành quyền điều khiển cho cấu trúc bên ngoài switch
- default: Từ khóa tuỳ chọn được sử dụng để chỉ rõ các câu lệnh nào được thực hiện chỉ khi tất cả các trường hợp nhận giá trị False
- default - action: Các câu lệnh được thực hiện chỉ khi tất cả các trường hợp nhận giá trị False

CHƯƠNG II - NỀN TẢNG CỦA JAVA

Ví dụ lệnh switch – case:

```
class SwitchDemo
{
public static void main(String agrs[])
 {
int day =4;
switch(day)
    {
case 0 :        System.out.println("Sunday");
break;
case 1 :        System.out.println("Monday");
break;
case 2 :        System.out.println("Tuesday");
break;
case 3 :        System.out.println("Wednesday");
break;
case 4 :        System.out.println("Thursday");
break;
case 5:         System.out.println("Friday");
break;
```

```
case 6 :        System.out.println("Satuday");
break;
default:        System.out.println("Invalid day of week");
          }
        }
}
```

Nếu giá trị của biến day là 4 ,chương trình sẽ hiển thị Thursday và cứ tiếp như vậy .

CHƯƠNG II - NỀN TẢNG CỦA JAVA

3-Vòng lặp While

Vòng lặp while được sử dụng khi vòng lặp được thực hiện mãi cho đến khi điều kiện thực thi vẫn là True. Số lượng lần lặp không được xác định trước song nó sẽ phụ thuộc vào từng điều kiện.

Cú pháp

```
while(condition)
{
action statement;
:
;
}
```

- condition: Biểu thức Boolean, nó trả về giá trị True hoặc False. Vòng lặp sẽ tiếp tục cho đến khi nào giá trị True được trả về.
- action statement: Các câu lệnh được thực hiện nếu condition nhận giá trị True

CHƯƠNG II - NỀN TẢNG CỦA JAVA

*Ví dụ v*òng lặp While

```
class WhileDemo
{
public static void main(String args[])
{
int a = 5,fact = 1;
while  (a >= 1)
      {
                fact *=a;
a--;
      }
System.out.println(The Factorial of 5 is "+fact);
   }
}
```

CHƯƠNG II - NỀN TẢNG CỦA JAVA

*Ví dụ v*òng lặp for

Đoạn chương trình sau hiển thi tổng của 5 số chẵn đầu tiên

```
class ForDemo
{
public static void main(String args[])
{
        int i=1,sum=0;
for (i=1;i<=10;i+=2)
sum+=i;
System.out.println ("sum of first five old numbers is "+sum);
}
}
```

Thông báo: **Sum of first five odd numbers is 25** được hiển thị.