Ngôn ngữ Java

(Phép toán – kiểu dữ liệu)

Môn học: Cơ sở lập trình [*Buổi 3-4*]

GV: Nguyễn Mai Huy

Cấu trúc chương trình đơn giản

- Bao gồm 2 phần
  - Khai báo thư viện, module:
    import <tên thư viện>;
    package <tên module, gói thư viện>;
  - Class & hàm main

```
public class <tenClass>{
    public static void main(String[] args)
    {
        …. //--- Mã nguồn dưới dạng các lệnh cần thực thi
        ….
    }
}
```

Cấu trúc chương trình

Khai báo class với từ khóa public

```
1  package FirstProgram;
2
3  public class helloWorld {
4
5      public static void main(String[] args) {
6          // TODO Auto-generated method stub
7          System.out.println("
8          System.out.print
9      }
10
11 }
12
```

Khai báo: package, import, …

public class phải có hàm main, được khai báo public static void

Lệnh được thực hiện dựa vào Class, Object & Method. Trong trường hợp trên, ta gọi đối tượng out của lớp System, và thông qua phương thức println để xuất chuỗi ra màn hình

System.out.print

System.out.print là lệnh dùng cho việc in dữ liệu ra màn hình máy tính.

Cú pháp:

System.out.print("*Chuỗi thông báo*"+ tham số+ …);

- "*Chuỗi thông báo*": Thông tin cần xuất ra trên màn hình
- Tham số, … : Các giá trị dữ liệu cần đưa vào, thể hiện trên chuỗi thông báo

System.out.println

Đây là phương thức xuất chuẩn của java, dùng để cho phép in ra thông báo trên màn hình

Cú pháp:

System.out.println(“Chuỗi thông báo”+ tham số + …);

Định dạng	Tham số	Kết quả	Ghi chú
%d	i	“123”	độ rộng 3
%05d	i	“00123”	Thêm 2 số 0
%7o”	i	“ 123”	Hệ 8, canh phải
%-9x	i	“7b “	Hệ 16, canh trái
%c	i	“{“	Ký tự có mã ASCII 123
%-#9x	i	“0x7b “	Hệ 16, canh trái
%10.5f	x	“ 0.12346”	Độ rộng 10, có 5 chữ số thập phân
%-12.5e	x	“1.23457e-01 “	Canh trái, in ra dưới dạng khoa học

Ký tự đặc biệt – Escape characters

Ký tự	Tác dụng	Mã ASCII
\n	Xuống hàng mới	10
\t	Tab	9
\b	Xóa ký tự bên trái	8
\r	Con trỏ trở về đầu hàng	13
\f	Sang trang	12
\a	Phát tiếng còi	7
\\	Xuất dấu chéo ngược	92
\’	Xuất dấu nháy đơn ‘	39
\’’	Xuất dấu nháy kép “	34
\xdd	Xuất ký tự có mã ASCII dạng Hex là dd	
\ddd	Xuất ký tự có mã ASCII dạng Dec là ddd	
\0	Ký tự NULL	0

Kiểu dữ liệu

- Java cung cấp 8 kiểu dữ liệu nguyên thủy "***primitive data***", hay còn gọi là built-in data type. Bao gồm:
  - 4 kiểu dùng cho chứa số nguyên: **byte, short, int, long**
  - 2 kiểu chứa số thực, có dấu chấm động: **float, double**
  - 1 kiểu chứa dữ liệu logic : **boolean**
  - 1 kiểu chứa ký tự: **char**

Primitive data types

Type	Description	Default	Size	Example Literals
boolean	true or false	false	1 bit	`true, false`
byte	twos complement integer	0	8 bits	(none)
char	Unicode character	`\u0000`	16 bits	`'a', '\u0041', '\101', '\\', '\'', '\n', 'ß'`
short	twos complement integer	0	16 bits	(none)
int	twos complement integer	0	32 bits	`-2, -1, 0, 1, 2`
long	twos complement integer	0	64 bits	`-2L, -1L, 0L, 1L, 2L`
float	IEEE 754 floating point	0.0	32 bits	`1.23e100f, -1.23e-100f, .3f, 3.14F`
double	IEEE 754 floating point	0.0	64 bits	`1.23456e300d, -1.23456e-300d, 1e1d`

Range of numeric data types in Java

Type	Size	Range
byte	8 bits	-128 .. 127
short	16 bits	-32,768 .. 32,767
int	32 bits	-2,147,483,648 .. 2,147,483,647
long	64 bits	-9,223,372,036,854,775,808 .. 9,223,372,036,854,775,807
float	32 bits	$3.40282347 \times 10^{38}$, $1.40239846 \times 10^{-45}$
double	64 bits	$1.7976931348623157 \times 10^{308}$, $4.9406564584124654 \times 10^{-324}$

Biến & hằng – (Variable - Constant)

✓ Biến – Variable

Biến được xem là những đối tượng khai báo trong chương trình để lưu trữ dữ liệu thuộc một "*kiểu xác định*" nào đó (*Theo quy ước của người lập trình*). Nội dung của biến có thể bị thay đổi tuỳ thuộc vào thuật toán & thời điểm mà ta truy xuất đến. Cú pháp khai báo biến trong C được mô tả như sau

<**kiểu_dữ_liệu**> <**ten_bien**>;

< **kiểu_dữ_liệu**> <**ten_bien**> = <giá_trị>;

Ví dụ:

```
int n,s;                                    [1]
short bcc = 5;                              [2]
long gt = 0, i, a = 12, b = 7, c = -11;     [3]
```

Ghi chú:

- [1] – Khai báo 2 biến n và s thuộc kiểu nguyên, độ lớn mỗi biến là 2 bytes, có thể chứa giá trị trong khoảng từ -32768 … + 32767
- [2] – Khai b áo biến bcc thuộc kiểu short và khởi gán giá trị ban đầu là 5
- [3] – Khai báo các biến gt, i, a, b, c kiểu long và tiến hành khởi gán giá trị ban đầu cho các biến gt, a, b, c

Biến & hằng – (Variable - Constant)

✓ Hằng – Constant

Hằng là đối tượng được khai báo trong chương trình để lưu trữ một giá trị thuộc một "*kiểu xác định*" nào đó (*kiểu dữ liệu của hằng sẽ được ngôn ngữ xác định tuỳ thuộc vào giá trị khai báo cho nó*). Nội dung của hằng ***không thể*** bị thay đổi trong suốt quá trình chương trình hoạt động.

Cú pháp khai báo hằng trong C có thể thực hiện bằng 1 trong 2 cách

final <kiểu_dữ_liệu> <ten_hằng> = <giá_trị>; [*Khai báo trong chương trình*]

Ví dụ:

final int tyGia = 21500; [2]

final char kt = 'A'; [3]

Ghi chú:

- [2] – Khai báo hằng tyGia chứa giá trị 21500 trong chương trình (*Kiểu nguyên*)
- [3] – Khai báo hằng kt chứa chữ A (*Kiểu char*)

* Định danh – (*Identifier*)

Đây là một khái niệm hết sức quan trọng trong chương trình, để đặt tên cho một thành phần nào đó trong chương trình của mình, lập trình viên cần phải đặt tên cho thành phần tương ứng (*VD: Tên biến, tên hằng, tên hàm, tên kiểu dữ liệu, …*), sau khi đã đặt tên, bạn có thể gọi sử dụng chúng (*Truy xuất*) trong phạm vi chương trình của mình. Tuy nhiên, việc đặt tên cho các thành phần trong chương trình phải thoả mãn các vấn đề sau:

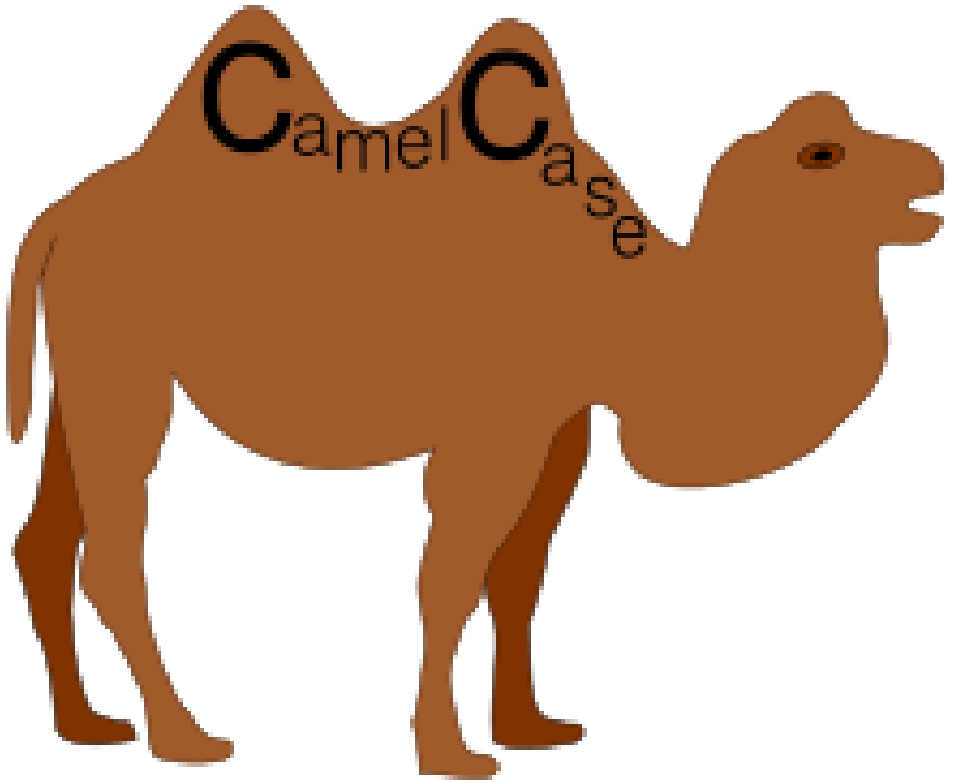
- **Không được trùng tên, phải là duy nhất** (*Nếu trùng thì sẽ: Thiếu tường minh*)
- Không được phép sử dụng các ký tự đặc biệt và khoảng trống, **chỉ được phép sử dụng ký tự trong bảng chữ cái** [a,b,…z;A,B,…Z], **ký số** [0,1,…9] và **dấu gạch dưới** "_" (Underscore)
- **Không được trùng tên với các từ khoá** quy ước trong ngôn ngữ lập trình
- **Luôn phải bắt đầu bởi ký tự trong bẳng chữ cái hoặc dấu gạch dưới**

Variable Name	Valid/Invalid
Employee	Valid
student	Valid
_Name	Valid
Emp_Name	Valid
@goto	Valid
static	Invalid as it is a keyword
4myclass	Invalid as a variable cannot start with a digit
Student&Faculty	Invalid as a variable cannot have the special character &

Quy ước đặt tên

Camel case

- Tuân theo nguyên tắc về đặt tên cho Identified trong lập trình
- Bắt đầu các từ phải là viết chữ hoa, các ký tự còn lại phải là chữ thường, *duy nhất từ đầu tiên phải viết chữ thường toàn bộ*. VD **iPhone**, **eBay**, **fedEx**, **chieuDai**, **tyGia**, …



Đối tượng "scanner"

- **Tạo đối tượng: scanner**
- Phương thức nhận dữ liệu
  - nextBoolean()
  - nextByte()
  - nextDouble()
  - nextFloat()
  - nextInt()
  - nextLine()
  - nextLong()
  - nextShort()
- Đóng Scanner
  - .close()

Các phép toán

Để phục vụ cho việc tính toán (*Xử lý trong chương trình*), ngôn ngữ lập trình cung cấp các phép toán: Số học (Arithmetic), luận lý (*Logic*), Quan hệ (*Relative*), Gán (*Asignment*), … các phép toán trong C thường được chia làm 2 dạng

- Phép toán một ngôi
- Phép toán hai ngôi

Trong đó, các thành phần tên gọi (*biến, hằng, hàm, …*) tham gia trong phép toán được gọi là hạng thức (*hoặc toán hạng*), các kí hiệu dùng để biểu diễn phép toán được gọi là toán tử

Ví dụ:

- **a+b**: là một phép toán số học, trong đó **a**, **b** là các toán hạng và ký hiệu "**+**" là toán tử số học biểu diễn cho ý nghĩa của phép toán cộng

Lưu ý:

- Số ngôi của phép toán chính là số toán hạng tham gia trong phép toán tương ứng. Trong ví dụ trên, ta có phép toán hai ngôi

Phép toán số học – *Arithmetic operators*

Operators	Description	Examples
+ (Addition)	Performs addition. If the two operands are strings, then it functions as a string concatenation operator and adds one string to the end of the other.	`40 + 20`
– (Subtraction)	Performs subtraction. If a greater value is subtracted from a lower value, the resultant output is a negative value.	`100 - 47`
* (Multiplication)	Performs multiplication.	`67 * 46`
/ (Division)	Performs division. The operator divides the first operand by the second operand and gives the quotient as the output.	`12000 / 10`
% (Modulo)	Performs modulo operation. The operator divides the two operands and gives the remainder of the division operation as the output.	`100 % 33`

Phép toán quan hệ - *elational operators*

Relational Operators	Descriptions	Examples
==	Checks whether the two operands are identical.	85 == 95
!=	Checks for inequality between two operands.	35 != 40
>	Checks whether the first value is greater than the second value.	50 > 30
<	Checks whether the first value is lesser than the second value.	20 < 30
>=	Checks whether the first value is greater than or equal to the second value.	100 >= 30
<=	Checks whether the first value is lesser than or equal to the second value.	75 <= 80

Toán tử tăng, giảm – *Increment and Decrement operators*

Expression	Type	Result
`valueTwo = ++ValueOne;`	Pre-Increment	`valueTwo = 6`
`valueTwo = valueOne++;`	Post-Increment	`valueTwo = 5`
`valueTwo = --valueOne;`	Pre-Decrement	`valueTwo = 4`
`valueTwo = valueOne--;`	Post-Decrement	`valueTwo = 5`

Toán tử gán – *Assignment operators*

Expression	Description	Result
`valueOne += 5;`	`valueOne = valueOne + 5`	`valueOne = 15`
`valueOne -= 5;`	`valueOne = valueOne - 5`	`valueOne = 5`
`valueOne *= 5;`	`valueOne = valueOne * 5`	`valueOne = 50`
`valueOne %= 5;`	`valueOne = valueOne % 5`	`valueOne = 0`
`valueOne /= 5;`	`valueOne = valueOne / 5`	`valueOne = 2`

Toán tử logic – *Logical operators*

Toán tử		Ý nghĩa
NOT	!	Phủ định
AND	&&	Giao, và
OR	||	Hội, Hoặc

Chuyển kiểu, "Ép kiểu" – *Casting*

Khi khai báo biến để sử dụng trong chương trình của mình, đồng nghĩa với việc bạn "*Đề nghị được cấp phát một vùng nhớ dùng cho mục đích lưu trữ dữ liệu với không gian ... - tuỳ thuộc vào kiểu đã được khai báo*", và như vậy chương trình của bạn đôi khi lại phải đối mặt với một số vấn đề khi xử lý. Hãy xét đoạn chương trình sau

```
int  a = 7, b = 2;
float c = a / b;
 System.out.printf("%d / %d = %2.1f", a,b,c);
```

Câu hỏi đặt ra là kết quả in ra trên màn hình là gì ?

Câu trả lời là: **7 / 2 = 3.0**

Chắc bạn sẽ rất ngạc nhiên, bởi lẽ ai cũng biết 7 chia 2 phải bằng 3.5 mới đúng !?.

Chuyển kiểu, "Ép kiểu" – *Casting*

Như vậy, để linh hoạt hơn trong những tình huống đặc biệt, ngôn ngữ C (*Hoặc những ngôn ngữ lập trình "**Tựa C**" như Java, C Sharp, ...*) cung cấp một kỹ thuật, được gọi là "***Chuyển kiểu***" hay "**Ép kiểu**" đối với dữ liệu hoặc biến trong chương trình, và đoạn chương trình trên sẽ được viết lại như sau

```
int  a = 7, b = 2;
float c = (float) a / b;
System.out.printf ("%d / %d = %2.1f", a,b,c);
```

Lúc này, kết quả in ra trên màn hình sẽ đúng như bạn muốn

7 / 2 = 3.5

Cú pháp:

(**kiểu_dữ_liệu**) <biến_cần_chuyển_kiểu>;
(**kiểu_dữ_liệu**) <dữ_liệu_cần_chuyển_kiểu>;

Chuyển kiểu, "Ép kiểu" – *Casting*

Trong C, việc ép kiểu được phân ra làm 2 loại

Ép kiểu ngầm định

```
float  v = 23;          //--- Tự động chuyển v = 23.0
int  f = 7.23;          //--- Tự động chuyển f = 7
```

Ép kiểu tường minh

```
int  a=10, b=4;
float  c = (float) a/b; //--- Chuyển giá trị của a thành số thực
                        //--- sau đó thực hiện phép chia a cho b
                        //--- Lúc này, b tạm thời cũng được
                        //--- chuyển thành số thực (một cách tạm thời)
```

Thứ tự ưu tiên của phép toán

Precedence (where 1 is the highest)	Operator	Description	Associativity
1	`()`	Parentheses	Left to Right
2	`++ or --`	Increment or Decrement	Right to Left
3	`*, /, %`	Multiplication, Division, Modulus	Left to Right
4	`+, -`	Addition, Subtraction	Left to Right
5	`<, <=, >, >=`	Less than, Less than or equal to, Greater than, Greater than or equal to	Left to Right
6	`==, !=`	Equal to, Not Equal to	Left to Right
7	`&&`	Conditional AND	Left to Right
8	`||`	Conditional OR	Left to Right
9	`=, +=, -=, *=, /=, %=`	Assignment Operators	Right to Left

Nhớ gì ?!!!

- Primitive data types: **boolean, char, int, float, …**
- **Identifier** là gì ?, các quy ước liên quan đến identifier. **Camel case** ?.
- Đối tượng Scanner dùng để làm gì ?. Nhớ được bao nhiêu phương thức của Scanner object (**nextInt, nextFloat, nextLine, …**)
- Có bao nhiêu loại phép toán, độ ưu tiên của phép toán
- Hiểu thế nào về **casting** ?

- **Primitive Data Types, The Java™ Tutorials, Oracle Document**
  (https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html)
- **Java Scanner Class, Java point**
  (https://www.javatpoint.com/Scanner-class)
- **Class scanner, Oracle document**
  (https://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html)
- **Formatting**
  (https://docs.oracle.com/javase/tutorial/essential/io/formatting.html)