UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

NGÔN NGỮ LẬP TRÌNH JAVA

Chương 1
GIỚI THIỆU NNLT JAVA (P2)
CÁC KIỂU DỮ LIỆU VÀ CÁC LỆNH CĂN BẢN

GVGD: ThS. Lê Thanh Trọng

NỘI DUNG

1. BIẾN & HẰNG
2. KIỂU DỮ LIỆU (KIỂU CƠ SỞ, KIỂU THAM CHIẾU)
3. TOÁN TỬ, BIỂU THỨC
4. CÁC CẤU TRÚC ĐIỀU KHIỂN (CHỌN, RẼ NHÁNH, LẶP)
5. LỚP BAO KIỂU CƠ SỞ

NỘI DUNG

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

1. **BIẾN & HẰNG**
2. KIỂU DỮ LIỆU (KIỂU CƠ SỞ, KIỂU THAM CHIẾU)
3. TOÁN TỬ, BIỂU THỨC
4. CÁC CẤU TRÚC ĐIỀU KHIỂN (CHỌN, RẼ NHÁNH, LẶP)
5. LỚP BAO KIỂU CƠ SỞ

BIẾN

- Biến là một vùng nhớ lưu các giá trị của chương trình
- Mỗi biến gắn với 1 kiểu dữ liệu và 1 định danh duy nhất là tên biến
- Tên biến phân biệt chữ hoa và chữ thường. Tên biến bắt đầu bằng 1 dấu **_, $,** hay **1 ký tự**, không được bắt đầu bằng 1 ký số.
- **Khai báo**
  - <kiểu dữ liệu> <tên biến>;
  - <kiểu dữ liệu> <tên biến> = <giá trị>;
- **Gán giá trị**

  <tên biến> = <giá trị>;

PHÂN LOẠI BIẾN

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Biến trong Java có 2 loại: **member** variable và **local** variable
- **Member**
  - Không cần khởi tạo giá trị (được tự động gán giá trị mặc định)
  - Được khai báo là thành phần của lớp
- **Local**
  - Bắt buộc phải khởi tạo giá trị trước khi sử dụng (nếu không sẽ tạo ra lỗi khi biên dịch)
  - Được khai báo trong một phương thức
- Viết hoa chữ cái đầu tiên các từ (SoLuongNhanVien, HoTen,...)

HẰNG

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Là một giá trị bất biến trong chương trình
- Tên đặt: Viết hoa tất cả các kí tự (NUMER_OF_MEMBER, PI,…)
- Được khai báo dùng từ khóa final, và thường dùng tiếp vĩ ngữ đối với các hằng số (l, L, d, D, f, F) nhằm chỉ rõ kiểu dữ liệu
- Ví dụ:
  - final int x = 10; // khai báo hằng số nguyên x = 10
  - final long y = 20L; // khai báo hằng số long y = 20
- Hằng ký tự: đặt giữa cặp nháy đơn ‘’
- Hằng chuỗi: là một dãy ký tự đặt giữa cặp nháy đôi “”

HẰNG KÝ TỰ ĐẶC BIỆT

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

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

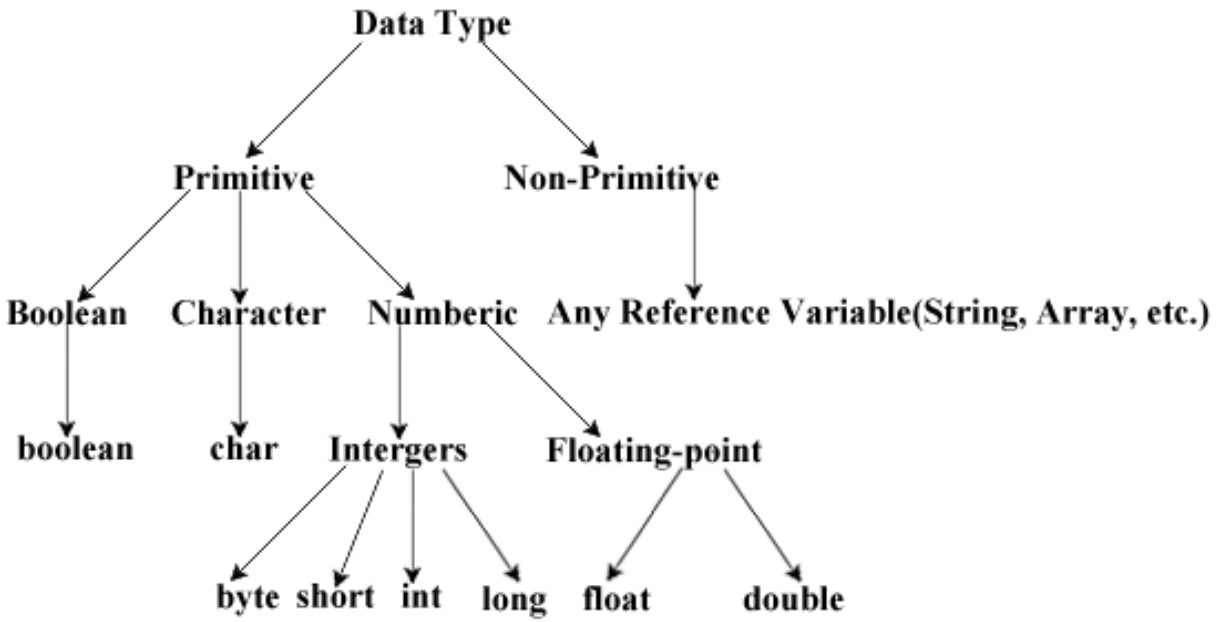
NỘI DUNG

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

1. BIẾN & HẰNG
2. **KIỂU DỮ LIỆU (KIỂU CƠ SỞ, KIỂU THAM CHIẾU)**
3. TOÁN TỬ, BIỂU THỨC
4. CÁC CẤU TRÚC ĐIỀU KHIỂN (CHỌN, RẼ NHÁNH, LẶP)
5. LỚP BAO KIỂU CƠ SỞ

KIỂU DỮ LIỆU

- Kiểu dữ liệu cơ sở (primitive data type)
- Kiểu dữ liệu tham chiếu (reference data type)



KIỂU DỮ LIỆU CƠ SỞ

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

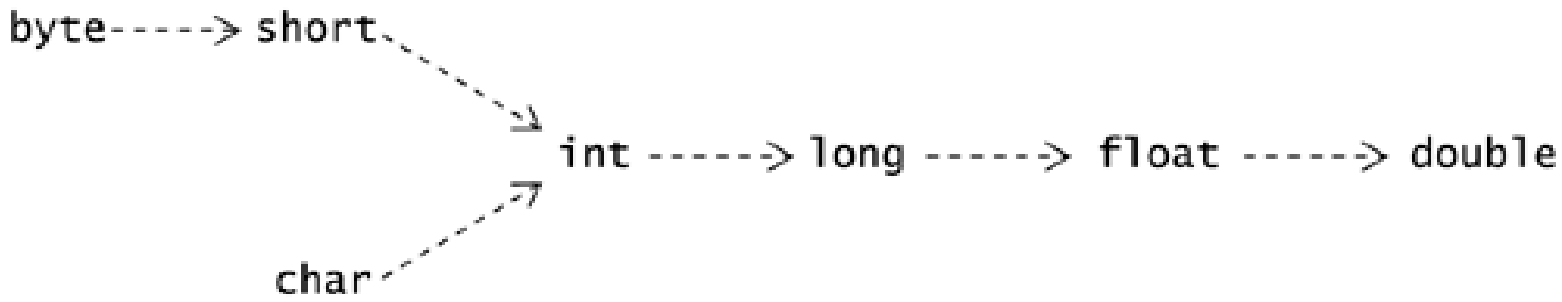
Kiểu	Kích thước (bits)	Giá trị	Giá trị mặc định
boolean	Tùy vào nền tảng	true và false	**false**
char	16	'\u0000' to '\uFFFF' (0 to 65535)	**null**
byte	8	–128 to +127 (-2^7 to $2^7 - 1$)	**0**
short	16	–32,768 to +32,767 (-2^{15} to $2^{15} - 1$)	**0**
int	32	–2,147,483,648 to +2,147,483,647 (-2^{31} to $2^{31} - 1$)	**0**
long	64	–9,223,372,036,854,775,808 to +9,223,372,036,854,775,807 (–263 to 263 – 1)	**0l**
float	32	1.4012984632481707e–45 to 3.4028234663852886E+38	**0.0f**
double	64	4.94065645841246544e–324 to 1.7976931348623157E+308	**0.0d**

KIỂU DỮ LIỆU CƠ SỞ (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Chuyển đổi kiểu dữ liệu: khi có sự không tương thích về kiểu dữ liệu (gán, tính toán biểu thức, truyền đối số gọi phương thức)
- Chuyển kiểu hẹp (lớn ➔ nhỏ): cần ép kiểu

  <tên biến 2> = (kiểu dữ liệu) <tên biến 1>;
- Chuyển kiểu rộng (nhỏ ➔ lớn): tự động chuyển



KIỂU DỮ LIỆU CƠ SỞ (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Không thể chuyển đổi giữa kiểu **boolean** với **int** và ngược lại
- Nếu 1 toán hạng kiểu double thì

  Toán hạng kia chuyển thành double
- Nếu 1 toán hạng kiểu float thì

  Toán hạng kia chuyển thành float
- Nếu 1 toán hạng kiểu long thì

  Toán hạng kia chuyển thành long
- Ngược lại,

  Tất cả chuyển thành int để tính toán

KIỂU DỮ LIỆU CƠ SỞ (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Ví dụ minh họa

```
byte x = 5;

byte y = 10;

byte z = x + y;
```

- Chương trình có lỗi ko???
- Dòng lệnh thứ 3 báo lỗi chuyển kiểu cần sửa lại

➔ byte z = (byte) (x + y);

KIỂU DỮ LIỆU THAM CHIẾU

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Khai báo biến tham chiếu

  <Kiểu đối tượng> <biến ĐT>;

- Khởi tạo đối tượng

  <Kiểu đối tượng> <biến ĐT> = new <Kiểu đối tượng>;

- Truy xuất thành phần đối tượng

  <biến ĐT>.<thuộc tính>

  <biến ĐT>.<phương thức>

KIỂU DỮ LIỆU THAM CHIẾU

- Mảng
- Enum (liệt kê)
- Lớp đối tượng

MẢNG

- Mảng là tập hợp các phần tử có cùng tên và cùng kiểu dữ liệu
- Mỗi phần tử được truy xuất thông qua chỉ số
- Khai báo mảng

<kiểu dữ liệu>[] <tên mảng>; // mảng 1 chiều

<kiểu dữ liệu> <tên mảng>[]; // mảng 1 chiều

<kiểu dữ liệu>[][] <tên mảng>; // mảng 2 chiều

<kiểu dữ liệu> <tên mảng>[][]; // mảng 2 chiều

MẢNG

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Khởi tạo mảng

```
int arrInt[] = {1, 2, 3};
char arrChar[] = {'a', 'b', 'c'};
String  arrString[] = {"ABC", "EFG", "GHI"};
int[][] 2dimensionArrInt =
{
        {16, 3, 2},
        {5, 10, 11, 8},
        {9, 6, 7, 12},
        {4}
};
```

MẢNG

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Cấp phát & truy cập mảng

  int [] arrInt = new int[100];

  int arrInt[100]; // Khai báo này trong Java sẽ bị báo lỗi

  Chỉ số mảng n phần tử: từ 0 đến n-1

- Mảng có số phần từ bằng 0

  int [] arrInt=new int[0];

- In mảng: arrInt.toString();

ARRAYS CLASS

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Package java.util.Arrays
- Một số phương thức
  - public static int binarySearch(Object[] a, Object key)
  - public static boolean equals(long[] a, long[] a2)
  - public static void fill(int[] a, int val)
  - public static void sort(Object[] a)
  - public static void copyOf(Object[] a, int length)

ENUM

```
enum Size { SMALL, MEDIUM, LARGE, EXTRA_LARGE };
Size s = Size.MEDIUM;
switch(s)
{
        case SMALL:
                …
        case MEDIUM:
                …
}
```

STRING

- Chuỗi các kí tự Unicode
- Dạng immutable
- '==' kiểm tra 2 String có cùng địa chỉ

  ➔ str1.equals(str2); //so sánh giá trị
- null khác với length = 0
  - String s1 = null;
  - String s2 = “”
- Chuyển chuỗi thành số

  int intValue = Integer.parseInt(str);
- Lớp tương tự StringBuilder (mutable)

STRING (TT)

```
String s1 = "Hello";
String s2 = "Hello";
String s3 = s1;
String s4 = new String("Hello");
String s5 = new String("Hello");

s1 == s1;       // true
s1 == s2;       // true
s1 == s3;       // true
s1 == s4;       // false
s4 == s5;       // false
```

MODIFIER

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- **Access Control Modifiers**
  - default
  - private
  - public
  - protected
- **Non Access Modifiers**
  - static
  - final
  - abstract
  - synchronized

TOÁN TỬ, BIỂU THỨC

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Toán tử số học

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

TOÁN TỬ, BIỂU THỨC (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Phép toán trên bit

Toán tử	Ý nghĩa
&	AND
|	OR
^	XOR
<<	Dịch trái
>>	Dịch phải
~	Bù bit

TOÁN TỬ, BIỂU THỨC (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- 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, BIỂU THỨC (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Toán tử gán

Toán tử	Ví dụ	Ý nghĩa
=	a = b	gán a = b
+=	a += 5	a = a + 5
-=	b -= 10	b = b – 10
*=	c *= 3	c = c * 3
/=	d /= 2	d = d/2
%=	e %= 4	e = e % 4

TOÁN TỬ, BIỂU THỨC (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Độ ưu tiên

Operators	Associativity
[] . () (method call)	Left to right
! ~ ++ -- + (unary) - (unary) () (cast) new	Right to left
* / %	Left to right
+ -	Left to right
<< >> >>>	Left to right
< <= > >= instanceof	Left to right
== !=	Left to right
&	Left to right
^	Left to right
|	Left to right
&&	Left to right
||	Left to right
?:	Right to left
= += -= *= /= %= &= |= ^= <<= >>= >>>=	Right to left

TOÁN TỬ, BIỂU THỨC (TT)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Toán tử điều kiện
  - Cú pháp: <điều kiện> ? <biểu thức 1> : < biểu thức 2>
  - Ví dụ:

```
int x = 10;
int y = 20;
int Z = (x<y) ? 30 : 40;
// Kết quả z = 30 do biểu thức (x < y) là đúng.
```

CẤU TRÚC ĐIỀU KHIỂN

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Cấu trúc if … else
  - Dạng 1: if (<điều_kiện>) {
    <khối_lệnh>;
    }
  - Dạng 2: if (<điều_kiện>) {
    <khối _lệnh1>;
    }
    else {
    <khối _lệnh2>;
    }

CẤU TRÚC ĐIỀU KHIỂN

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Cấu trúc switch … case

```
switch (<biến>) {
        case <giátrị_1>:
                <khối_lệnh_1>;
                break;
        ….
        case <giátrị_n>:
                <khối_lệnh_n>;
                break;
        default:
                <khối lệnh default>;
}
```

CẤU TRÚC ĐIỀU KHIỂN

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Cấu trúc lặp
  - Dạng 1:

    while (<điều_kiện_lặp>) {…}
  - Dạng 2:

    do {…

    } while (điều_kiện);
  - Dạng 3:

    for (khởi_tạo_biến_đếm;đk_lặp;tăng_biến) {…}
  - Dạng 4 (từ Java 5)

    int arr[]={1, 2, 3};

    for(int i: arr){….}

CẤU TRÚC ĐIỀU KHIỂN

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Cấu trúc lệnh nhảy jump: dùng kết hợp nhãn (label) với từ khóa break và continue để thay thế cho lệnh goto (trong C)
- Ví dụ:

```
label:
for (…) {
  for (…) {
        if (<biểu thức điều kiện>)
              break label;
        else
              continue label;
  }
}
```

INPUT

- Lớp java.util.Scanner

public boolean	**nextBoolean**() Details
public byte	**nextByte**() Details
public byte	**nextByte**(int radix) Details
public double	**nextDouble**() Details
public float	**nextFloat**() Details
public int	**nextInt**() Details
public int	**nextInt**(int radix) Details
public String	**nextLine**() Details
public long	**nextLong**() Details
public long	**nextLong**(int radix) Details
public short	**nextShort**() Details
public short	**nextShort**(int radix) Details

INPUT

```
Scanner in = new Scanner(System.in);

System.out.print("What is your name? ");

String name = in.nextLine();

System.out.print("How old are you? ");

int age = in.nextInt();
```

OUTPUT

System.out.print(…);

System.out.println(…);

System.out.printf(…);

OUTPUT

- printf()
- Cú pháp: **%[flags][width][.precision]conversion-character**
- **[flags]: định dạng hiển thị cho output** (thường cho số nguyên hoặc số thực: thêm "+", thêm khoảng trắng, dấu ngoặc đơn canh trái, thêm dấu phân tách phần ngàn,..)
- **[width]:** xác định độ dài cho output (kích thước ngắn nhất)
- **[.precision]:** xác định số lượng ký số cho phần thập phân (hoặc kích thước của chuỗi con được trích từ một chuỗi)

OUTPUT

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Flag	Purpose	Example
+	Prints sign for positive and negative numbers	+3333.33
space	Adds a space before positive numbers	| 3333.33|
0	Adds leading zeroes	003333.33
-	Left-justifies field	|3333.33 |
(	Encloses negative numbers in parentheses	(3333.33)
,	Adds group separators	3,333.33
# (for f format)	Always includes a decimal point	3,333.
# (for x or o format)	Adds 0x or 0 prefix	0xcafe
$	Specifies the index of the argument to be formatted; for example, %1$d %1$x prints the first argument in decimal and hexadecimal	159 9F
<	Formats the same value as the previous specification; for example, %d %<x prints the same number in decimal and hexadecimal	159 9F

Conversion character	Type	Example
d	Decimal integer	159
x	Hexadecimal integer	9f
o	Octal integer	237
f	Fixed-point floating-point	15.9
e	Exponential floating-point	1.59e+01
g	General floating-point (the shorter of e and f)	—
a	Hexadecimal floating-point	0x1.fccdp3
s	String	Hello
c	Character	H
b	boolean	true
h	Hash code	42628b2
tx	Date and time	See Table 3.7
%	The percent symbol	%
n	The platform-dependent line separator	—

Một số ký pháp

- %n: xuống dòng
- %b hay %B: Boolean
- %s: String
  - printf(“’%-12s' %n", “Java Code"); // ‘Java Code ‘
- %c hay %C: char
- System.out.printf(Locale.US, "%,d %n", 10000); //10,000
- System.out.printf(Locale.ITALY, "%,d %n", 10000); //10.000

Một số ký pháp

- %t[H/M/S]: date [Giờ/Phút/Giây]
  - Date date = new Date();
  - System.out.printf("%tT%n", date);
  - System.out.printf("%tH:%tM:%tS%n", date, date, date);
- System.out.printf("%1$tA, %1$tB %1$tY %n", date);
  - A: thứ (chữ: Monday, Thursday)
  - d: ngày trong tháng với 2 ký số
  - B: tháng (chữ: April, November)
  - m: tháng với 2 ký số
  - Y: năm với 04 ký số
  - y: năm với 02 ký số cuối
  - System.out.printf("%1$td.%1$tm.%1$tY %n", date); // 05.03.2024

LỚP BAO KIỂU DỮ LIỆU

Kiểu cơ sở	Lớp bao tương ứng (java.lang.*)	Ghi chú
boolean	Boolean	- Gói (package): chứa nhóm nhiều class. - Ngoài các Wrapper Class, gói java.lang còn cung cấp các lớp nền tảng cho việc thiết kế ngôn ngữ java như: String, Math, …
byte	Byte	
short	Short	
char	Character	
int	Integer	
long	Long	
float	Float	
double	Double	

LỚP BAO KIỂU DỮ LIỆU

- Chuyển kiểu số (primitive) thành kiểu object hỗ trợ việc truyền tham chiếu cho phương thức
- Các cấu trúc trong Collection (Array, Vector,…) chỉ thao tác trên các biến là đối tượng
- Chỉ các đối tượng mới được hỗ trợ đồng bộ (đa luồng)
- Các lớp Wrapper cũng là các lớp có giá trị không thể thay đổi được (immutable)
- Các lớp Wrapper là các lớp **final**, và vì vậy bạn không thể nào tạo được lớp con từ các lớp Wrapper

Chuyển Kiểu Nguyên Thủy Và Wrapper

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- **Boxing**: nguyên thủy ➔ kiểu Wrapper
  - int a = 500;
  - Integer i = new Integer(a);
  - Float f = new Float(4.5);
  - Double d = new Double(5);
  - Character ch = new Character('a');
  - Boolean b = new Boolean(true);
- **Autoboxing**: gán trực tiếp các giá trị nguyên thủy vào cho các lớp Wrapper
  - int a = 500;
  - Integer i = a;
  - Integer j = 500;
  - Float f = 4.5f;

Chuyển Kiểu Nguyên Thủy Và Wrapper

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- **Unboxing**: chuyển từ một kiểu Wrapper ➔ kiểu nguyên thủy
  - int a = 10;
  - Integer i = a; // Autoboxing
  - int i2 = i.intValue(); // Unboxing
  - Character ch = ‘z'; // Autoboxing
  - char ch2 = ch.charValue(); // Unboxing
- Với kỹ thuật unboxing cũng có thể autoboxing
  - int a = 10;
  - Integer i = a;
  - int i2 = i; // Unboxing

Các Phương Thức Hữu Ích Của Lớp Wrapper

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- **parseXxx()**
  - int i = Integer.**parseInt**(“2");
  - float f = Float.**parseFloat**(“1.5");
  - boolean b = Boolean.**parseBoolean**(“false");
- **toString()**
  - String str = Integer.**toString**(10);
  - System.out.println(str);
- **xxxValue()**
  - Double f = 0.5;
  - int i = f.**intValue**();
  - byte b = f.**byteValue**();

Các Phương Thức Hữu Ích Của Lớp Wrapper

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- **equals()**
  - Integer i1 = 50, i2 = 50;
  - System.out.println("So sánh i1 & i2: " + i1.equals(i2)); //true
  - Float f1 = 5.2f, f2 = 2.4;
  - System.out.println("So sánh f1 & f2: " + f1.equals(f2)); //false
- **compare()**
  - Float f1 = 20.25f, f2 = 2.43f;
  - System.out.println("So sánh f1 & f2: " + Float.**compare**(f1,f2));
- **compareTo()**
  - Integer i = 50, i1 = 50, i2 = 52, i3 = 30;
  - System.out.println("So sánh giữa i & i1= " + i.**compareTo**(i1)); //0
  - System.out.println("So sánh giữa i & i2= " + i.**compareTo**(i2)); //-1
  - System.out.println("So sánh giữa i & i3= " + i.**compareTo**(i3)); //1

Các Phương Thức Hữu Ích Của Lớp Wrapper

- **abs():** Trả về giá trị tuyệt đối của đối số được chỉ định
- **ceil():** Trả về số nguyên nhỏ nhất bằng hoặc lớn hơn đối số đã chỉ định ở định dạng kép
- **floor():** Trả về số nguyên lớn nhất bằng hoặc nhỏ hơn đối số đã chỉ định ở định dạng kép
- **round():** Trả về long hoặc int gần nhất theo kiểu trả về của phương thức
- **min():** Trả về giá trị nhỏ hơn giữa hai đối số
- **max():** Trả về giá trị lớn hơn giữa hai đối số
- **exp:**Trả lại e cho lũy thừa của đối số, tức là cơ số của logarit tự nhiên
- **log():** Trả về logarit tự nhiên của đối số được chỉ định
- **pow():** Trả về kết quả của đối số đầu tiên được nâng lên thành lũy thừa của đối số thứ hai

BÀI TẬP

1. Viết và chạy chương trình xuất ra dòng “Hello World” với Java (bằng notepad và IDE)
2. Viết chương trình giải phương trình bậc hai ax2 + bx + c = 0.
3. Nhập n số nguyên. Hãy sắp xếp giá trị của các số nguyên này theo thứ tự tăng dần.
4. Nhập vào tháng và năm. Cho biết tháng đó có bao nhiêu ngày.
5. Viết chương trình quản lý một dãy số nguyên gồm các tính năng: nhập, xuất dãy; cho phép thêm, xóa, sửa các số trong dãy; sắp xếp dãy số tăng dần; tính giá trị trung bình của dãy và cho biết phần tử nào gần với giá trị trung bình nhất.

BÀI TẬP

6. Viết chương trình xuất ra lịch của một năm (do người dùng nhập vào)
7. Viết chương trình nhập vào mảng phân số(n phần tử) và xuất ra phân số nhỏ nhất, lớn nhất của mảng vừa nhập.
8. Viết hàm nhập, xuất 1 mảng gồm n phần tử (số nguyên) và thực hiện các yêu cầu
   - Đếm số phần tử chẵn và lẻ
   - Tính giá trị trung bình của mảng
   - Tìm phần tử lớn nhất và nhỏ nhất
   - Xuất mảng theo chiều ngược lại
   - Sắp xếp mảng tăng dần và xuất ra kết quả

TÓM TẮT BÀI HỌC

- Biến là một vùng nhớ lưu các giá trị của chương trình và có 2 loại: member và local
- Các kiểu dữ liệu cơ sở: boolean, char, byte, short, int, long, float, double
- Kiểu tham chiếu: mảng, enum, kiểu lớp (String, StringBuffer, Thread, Date, Scanner,...
- Các lệnh cơ bản (toán tử, biểu thức, lệnh điều khiển,...) tương tự C++
- Nhập xuất console
  - Scanner
  - System (out/in)
- Mỗi kiểu cơ sở có lớp bao tương ứng