PROJECT 1
LẬP TRÌNH AN TOÀN TRONG JAVA

Bùi Trọng Tùng
Khoa kỹ thuật máy tính
Trường CNTT – TT, Đại học BKHN

1

1

Đội ngũ giảng dạy

- Giảng viên:
  - TS. Tống Văn Vạn (vantv@soict.hust.edu.vn)
  - ThS. Bùi Trọng Tùng (tungbt@soict.hust.edu.vn)
  - Khoa Kỹ thuật máy tính, SoICT, HUST
- Trợ giảng:
  - ThS. Nguyễn Quốc Khánh
  - ThS. Lê Văn Đồng
  - Trung tâm An toàn thông tin Đại học BKHN

2

2

Kế hoạch học tập

- Lý thuyết về lập trình an toàn trong Java
- Thực hiện project theo mô hình Pair Programming:
  - Mỗi nhóm gồm 2 sinh viên
  - Nhận đề tài và yêu cầu
  - Lập trình phát triển ứng dụng
- Đánh giá chéo giữa các nhóm
  - Các nhóm bốc thăm ngẫu nhiên đánh giá chéo về tuân thủ các quy tắc lập trình an toàn
- Báo cáo kết quả

3

3

BÀI 1.
CƠ BẢN VỀ LẬP TRÌNH JAVA

Giới thiệu chung về Java
Sử dụng trình biên dịch Java và máy ảo Java
Các phần tử cơ bản trong Java
Toán tử và biểu thức
Một số lệnh vào ra cơ bản

4

4

1. GIỚI THIỆU CHUNG VỀ JAVA

5

5

Ngôn ngữ lập trình Java

- Đơn giản
  - Loại bỏ con trỏ
  - Không có goto, file header
  - Loại bỏ struct và union
- Hướng đối tượng
  - Java được thiết kế xoay quanh mô hình hướng đối tượng.
- Mạnh
  - Chặt chẽ → Loại bỏ các kiểu dữ liệu dễ gây lỗi
- Độc lập phần cứng
  - Viết một lần, chạy nhiều nơi (chỉ khác nhau trình thông dịch – máy ảo Java)

6

6

Ngôn ngữ lập trình Java

- Ngày nay, nhắc đến Java, không còn nhắc đến như một ngôn ngữ mà còn là một công nghệ, một nền tảng phát triển.
  - Java có một cộng đồng phát triển mạnh mẽ
  - Một tập hợp các thư viện với số lượng lớn (từ Sun và các nguồn khác)
- Java là ngôn ngữ vừa biên dịch vừa thông dịch
  - Biên dịch: mã nguồn được biên dịch bằng công cụ JAVAC để chuyển thành dạng ByteCode
  - Thông dịch: Bytecode thực thi trên từng loại máy cụ thể nhờ chương trình thông dịch (nằm trong máy ảo Java)
  - Nhằm mục đích viết một lần, chạy nhiều nơi

7

7

Ngôn ngữ lập trình Java

- J2SE (Java 2 Platform Standard Edition)
  - Cung cấp các thành phần cốt lõi để xây dựng ứng dụng desktop-based
  - JRE: Java Runtime Environment: môi trường thực thi để chạy các ứng dụng Java
- J2EE (Java 2 Platform Enterprise Edition)
  - Xây dựng các ứng dụng hướng dịch vụ (service-oriented)
  - Web service
  - Ứng dụng doanh nghiệp
- J2ME (Java 2 Platform Mobile Edition): xây dựng ứng dụng di động

8

8

Ngôn ngữ lập trình hướng đối tượng

- Chương trình gồm các đối tượng và tương tác giữa chúng

  Chương trình = Đối tượng + Thông điệp

- Lớp(Class): mô hình hóa đối tượng thực
  - Dùng ngôn ngữ lập trình để mô tả đối tượng thực
- Đối tượng(Object): một thể hiện của lớp

Máy ATM – Đối tượng thực*	Mô hình hóa – Lớp (class) ATM
Thuộc tính: Ngân hàng, Vị trí	**Thuộc tính**: bank, location
Hành động: Kiểm tra mã PIN, Rút tiền, Chuyển khoản	**Hàng động**: userAuthenticated(), withdraw(), tranfer()

9

9

Cài đặt môi trường phát triển Eclipse

- Eclipse là một trong những IDE thông dụng nhất
- Phiên bản được sử dụng trong học phần: v12-2021
- Địa chỉ download:

https://www.eclipse.org/downloads/packages/release/2021-12/r

Gói cài đặt: Eclipse IDE for Java Developers

- Trên cửa sổ Command Line lần lượt gõ 2 lệnh:

> javac –version

➢ java –version

```
C:\Users\TungBT>javac -version
javac 19.0.1

C:\Users\TungBT>java -version
java version "19.0.1" 2022-10-18
Java(TM) SE Runtime Environment (build 19.0.1+10-21)
Java HotSpot(TM) 64-Bit Server VM (build 19.0.1+10-21, mixed mode, sha
```

Nếu hai lệnh được thực hiện thành công, thông tin phiên bản Java sẽ xuất hiện → Eclipse được cài đặt đúng

10

10

Chương trình Java đầu tiên

```
// The first Java program

public class HelloWorld {
    /*Phương thức main, được gọi đầu tiên khi chạy bất cứ
    ứng dụng Java nào*/
    public static void main (String[] args) {
       System.out.println("Hello Java!");
    } //Kết thúc phương thức main
} //Kết thúc lớp HelloWorld
```


11

Khai báo lớp

```
// The first Java program

public class HelloWorld {
    /*Phương thức main, được gọi đầu tiên khi chạy bất cứ
    ứng dụng Java nào*/
    public static void main (String[] args) {
       System.out.println("Hello Java!");
    } //Kết thúc phương thức main
} //Kết thúc lớp HelloWorld
```

Khai báo lớp có tên là HelloWorld

Khai báo một phương thức của lớp HelloWorld

- Tạm thời, chúng ta chưa phân tích kỹ ý nghĩa của các khai báo này. Hãy để dành nó cho các bài sau.

12

Câu lệnh

```
// The first Java program

public class HelloWorld {
    //Phương thức main, được gọi đầu tiên khi chạy bất cứ
    ứng dụng Java nào*/
    public static void main (String[] args) {
        System.out.println("Hello Java!");
    } //Kết thúc phương thức main
} //Kết thúc lớp HelloWorld
```

- *Kết thúc bằng dấu ;*
- Một câu lệnh có thể viết trên 1 hoặc nhiều dòng
- Ký hiệu để bao khối lệnh { }

13

13

Chú thích

```
// The first Java program

public class HelloWorld {
    /*Phương thức main, được gọi đầu tiên khi chạy bất cứ
    ứng dụng Java nào*/
    public static void main (String[] args) {
        System.out.println("Hello Java!");
    } //Kết thúc phương thức main
} //Kết thúc lớp HelloWorld
```

- Giải thích mã nguồn, các câu lệnh, các bước xử lý phức tạp trong chương trình:

// chú thích trên một dòng

/ chú thích trên một đoạn*/*

- Không có ý nghĩa thực thi

14

14

Dịch chương trình

- Sử dụng chương trình Notepad gõ lại đoạn lệnh trên
  - Bắt đầu tập thói quen lùi dòng cho các khối lệnh bao nhau
- Lưu file mã nguồn có tên Test.java vào thư mục D:\Java\W1
- Dịch: mở cửa sổ Command Line và gõ lệnh sau để dịch

>javac D:\Java\W1\Test.java

- Lỗi cú pháp: *"class HelloWorld is public, should be declared in a file named HelloWorld.java"*
  - Nguyên nhân: Tên file mã nguồn khác tên lớp đã khai báo
  - Sửa: đặt lại tên file
- Dịch thành công: dấu nhắc lệnh > xuất hiện trở lại. File mã nguồn được dịch thành file Byte Code có cùng tên và đuôi .class, cùng thư mục với file mã nguồn

Sử dụng Eclipse giúp ta tránh được các lỗi cú pháp

15

15

Chạy chương trình

- Thực hiện lệnh:

> java D:\Java\W1\HelloWorld

Chú ý: Có thể sẽ gặp thông báo lỗi "Could not find or load main class..."

Khắc phục: chuyển vào thư mục chứa file .class

>D:

>cd .\Java\W1

>java HelloWorld

- Kết quả thực hiện

```
D:\Java\bin>java HelloWorld
Hello World!
```

System.out.println(): Hiển thị thông báo trên cửa sổ Console

16

16

Sửa lại phương thức main

```
public static void main (String[] args) {
        myName = "Tung";
        System.out.println("Hello Java! I am " + myName);
    } //Kết thúc phương thức main
} //Kết thúc lớp HelloWorld
```

- Lỗi cú pháp:

```
HelloWorld.java:6: error: cannot find symbol
                myName = "Tung";
                ^
  symbol:   variable myName
  location: class HelloWorld
HelloWorld.java:7: error: cannot find symbol
                System.out.println("Hello World! I am " + myName);
                                                        ^
  symbol:   variable myName
  location: class HelloWorld
2 errors
```

- Nguyên nhân: chưa khai báo *myName*

System.out.println(): Hiển thị thông báo trên cửa sổ Console

17

2. CÁC PHẦN TỬ CƠ BẢN CỦA JAVA

18

Từ khóa và định danh

- Là những từ được Java quy định ý nghĩa và cách sử dụng
- Định danh: xâu ký tự, xác định duy nhất một phần tử trong chương trình
- Quy định với định danh:
  - Không đặt trùng với từ khóa
  - Không bắt đầu bằng chữ số
  - Ký tự được phép sử dụng: chữ cái, chữ số, $, _
  - Phân biệt chữ hoa, chữ thường

19

19

Quy tắc "con lạc đà"

- Biến số: bắt đầu bằng chữ thường, viết hoa chữ cái đầu tiên các từ còn lại
- Hằng số: Toàn bộ bằng chữ hoa, dùng dấu _ ngăn cách từ
- Lớp: viết hoa chữ cái đầu tiên các từ
- Thuộc tính, phương thức: bắt đầu bằng chữ thường, viết hoa chữ cái đầu tiên các từ còn lại
- Gói: sử dụng chữ thường

Biến số	myName, numberOfStudent
Hằng số	MAX_LINE, USER_PARAMETER
Lớp	HelloWorld, Student
Thuộc tính	studentID, mark
Phương thức	setValue(), getValue(),sortByName()
Gói	sis.subject, sis.student

20

20

Các kiểu dữ liệu nguyên thủy

Kiểu dữ liệu	Kích thước	Giá trị mặc định	Giá trị nhỏ nhất	Giá trị lớn nhất
byte	8	0	-128	127
short	16	0	-32768	32767
int	32	0	-2147483648	2147483647
long	64	0L	-2^63	(2^63) – 1
float	32	0.0f		
double	64	0.0d		
boolean	Không xác định	false	NA	NA
char	16	\u0000	NA	NA

21

21

Toán tử số học

Toán tử	Ý nghĩa	Kiểu dữ liệu của toán hạng	Ví dụ
-	Phép đổi dấu	Số thực hoặc số nguyên	`int a, b;` `-12; -a; -25.6;`

+	Phép toán cộng	Số thực hoặc số nguyên	`float x, y;`
			`5 + 8; a + x;`
			`3.6 + 2.9;`

-	Phép toán trừ	Số thực hoặc số nguyên	`3 - 1.6; a - 5;`
*	Phép toán nhân	Số thực hoặc số nguyên	`a * b; b * y;` `2.6 * 1.7;`
/	Phép toán chia	Số thực hoặc số nguyên	`10.0/3.0;` (bằng 3.33…) `10/3.0;` (bằng 3.33…) `10.0/3;` (bằng 3.33…)
/	Phép chia lấy phần nguyên	Giữa 2 số nguyên	`10/3;` (bằng 3)
%	Phép chia lấy phần dư	Giữa 2 số nguyên	`10%3;` (bằng 1)

22

Toán tử nhị phân

Toán tử	Ý nghĩa	Kiểu dữ liệu của toán hạng	Ví dụ	
&	Phép VÀ nhị phân	2 số nhị phân	`0 & 0`	(có giá trị 0)
			`0 & 1`	(có giá trị 0)
			`1 & 0`	(có giá trị 0)
			`1 & 1`	(có giá trị 1)
			`101 & 110`	(có giá trị 100)
|	Phép HOẶC nhị phân	2 số nhị phân	`0 | 0`	(có giá trị 0)
			`0 | 1`	(có giá trị 0)
			`1 | 0`	(có giá trị 0)
			`1 | 1`	(có giá trị 1)
			`101 | 110`	(có giá trị 111)

23

23

Toán tử nhị phân

^	Phép HOẶC CÓ LOẠI TRỪ nhị phân	2 số nhị phân	`0 ^ 0`	(có giá trị 0)
			`0 ^1`	(có giá trị 1)
			`1 ^ 0`	(có giá trị 1)
			`1 ^ 1`	(có giá trị 0)
			`101 ^ 110`	(có giá trị 011)
<<	Phép DỊCH TRÁI nhị phân	Số nhị phân	`a << n`	(có giá trị $a*2^n$)
			`101 << 2`	(có giá trị 10100)
>>	Phép DỊCH PHẢI nhị phân	Số nhị phân	`a >> n`	(có giá trị $a/2^n$)
			`101 >> 2`	(có giá trị 1)
~	Phép ĐẢO BIT nhị phân (lấy Bù 1)	Số nhị phân	`~ 0`	(có giá trị 1)
			`~ 1`	(có giá trị 0)
			`~ 110`	(có giá trị 001)

24

24

Toán tử quan hệ

Toán tử	Ý nghĩa	Ví dụ
>	So sánh lớn hơn giữa 2 số nguyên hoặc thực.	2 > 3 (có giá trị 0) 6 > 4 (có giá trị 1) a > b
>=	So sánh lớn hơn hoặc bằng giữa 2 số nguyên hoặc thực.	6 >= 4 (có giá trị 1) x >= a
<	So sánh nhỏ hơn giữa 2 số nguyên hoặc thực.	5 < 3 (có giá trị 0),
<=	So sánh nhỏ hơn hoặc bằng giữa 2 số nguyên hoặc thực.	5 <= 5 (có giá trị 1) 2 <= 9 (có giá trị 1)
==	So sánh bằng nhau giữa 2 số nguyên hoặc thực.	3 == 4 (có giá trị 0) a == b
!=	So sánh không bằng (so sánh khác) giữa 2 số nguyên hoặc thực.	5 != 6 (có giá trị 1) 6 != 6 (có giá trị 0)

25

25

Toán tử logic

Toán tử	Ý nghĩa	Kiểu dữ liệu của toán hạng	Ví dụ
&&	Phép VÀ LOGIC. Biểu thức VÀ LOGIC bằng 1 khi và chỉ khi cả 2 toán hạng đều bằng 1	Hai biểu thức logic	3<5 && 4<6 (có giá trị 1) 2<1 && 2<3 (có giá trị 0) a > b && c < d
||	Phép HOẶC LOGIC. Biểu thức HOẶC LOGIC bằng 0 khi và chỉ khi cả 2 toán hạng bằng 0.	Hai biểu thức logic	6 || 0 (có giá trị 1) 3<2 || 3<3 (có giá trị 0) x >= a || x == 0
!	Phép PHỦ ĐỊNH LOGIC một ngôi. Biểu thức PHỦ ĐỊNH LOGIC có giá trị bằng 1 nếu toán hạng bằng 0 và có giá trị bằng 0 nếu toán hạng bằng 1	Biểu thức logic	!3 (có giá trị 0) !(2>5) (có giá trị 1)

26

26

Các toán tử khác

- Toán tử rút gọn: +=, −=, *=, /= ...
- Toán tử tăng 1 đơn vị: ++
- Toán tử giảm 1 đơn vị: − −
- Toán tử điều kiện: ?:

(boolean_expression)?true_expression:false_expression

  - Nếu *boolean_expression* đúng, tính giá trị *true-expression*
  - Nếu *boolean_expression* sai, tính giá trị *false-expression*

27

27

Hằng số

- Phần tử trong chương trình không thể thay đổi giá trị
- Cú pháp:

```
final DataType CONSTANT_NAME =  Literal;
```

- Trong đó:
  - **`final`**: từ khóa
  - `DataType`: Kiểu dữ liệu
  - `CONSTANT_NAME`: Tên hằng. Tuân thủ quy tắc định danh
  - `Literals` : Giá trị hằng

28

28

Giá trị hằng (Literals)

- Boolean: `true`, `false`
- Số nguyên:
  - Hệ cơ số 8: Bắt đầu bằng chữ số 0
    - Ví dụ: 012 = $001010_{(2)}$ = 8 + 2 = $10_{(10)}$
  - Hệ cơ số 16: Bắt đầu bằng 0x
    - Ví dụ: 0x2A = 00101010 = 2x16 + 10 = 42
  - Kiểu dữ liệu `long`: Kết thúc bằng ký tự `L` hoặc `l`
    - Ví dụ: 10L

29

29

Giá trị hằng

- Số thực:
  - Mặc định có kiểu `double`
  - Kiểu `float`: Kết thúc bằng ký tự `F` hoặc `f`
  - Dạng dấu phảy động: Ký tự `e` (hoặc `E`) kèm theo số mũ
    - Ví dụ: 1.2E7
- Ký tự: Đặt giữa dấu nháy đơn. Ví dụ: `'a'`
- Xâu ký tự: Đặt giữa dấu nháy kép
  - Ví dụ: `"SoICT-HUST"`

30

30

Biến số

- Là phần tử trong chương trình có thể thay đổi giá trị
- Cú pháp:

```
DataType varName1,varName2,...,varNameN;
```

hoặc

```
DataType varName1 =  Literal1,...,varNameN = LiteralN;
```

- Trong đó:
  - `varName` là tên biến, đặt theo quy tắc định danh
  - `Literral` có thể là một biến khác đã được khai báo trước
- Trước khi sử dụng trong biểu thức, biến phải được khởi tạo giá trị

31

31

Toán tử gán

- Cú pháp:

```
variable = expression;
```

- Biến `variable` và biểu thức `expression` nên có cùng kiểu dữ liệu
- Trong thường hợp hai vế có kiểu dữ liệu khác nhau:
  - Vế trái có kiểu dữ liệu "rộng" hơn: ép kiểu tự động
  - Ngược lại: không hợp lệ. Nếu vẫn muốn thực hiện phép gán, cần ép kiểu
- Trong biểu thức có các giá trị khác kiểu, tất cả các giá trị được ép tự động thành kiểu rộng nhất

32

32

Toán tử gán (Ví dụ)

```
long a = 1.2; //không hợp lệ
long b = (long) 1.2; //hợp lệ
int m = b/2; //không hợp lệ
char ch = 'a'; //hợp lệ
int n = ch; //hợp lệ
short k = ch; //không hợp lệ
short p = (short) ch; //hợp lệ
float x = 1.2; //không hợp lệ
float y = 1.2f; //hợp lệ
```

33

33

3. CÁC PHƯƠNG THỨC VÀO RA CƠ BẢN

34

34

Hiển thị dữ liệu

- Phương thức `System.out.println()`: Hiển thị dữ liệu và xuống dòng
- Phương thức `System.out.print()`: Hiển thị dữ liệu
- Phương thức `System.out.printf()`: Hiển thị dữ liệu có định dạng
- Phương thức `System.out.format()`: Hiển thị dữ liệu có định dạng
- Có thể dùng toán tử + để nối các dữ liệu khi hiển thị

35

35

Định dạng dữ liệu khi hiển thị

- Dạng Boolean: %b
- Dạng ký tự: %c
- Dạng số nguyên: %d
- Dạng số thực: %f
- Dạng xâu ký tự: %s

36

36

Nhập dữ liệu từ bàn phím

- Khá phức tạp vì Java coi dữ liệu nhận được từ bàn phím là luồng vào
- Thực hiện
  - Đọc dữ liệu vào bộ đệm:

```
BufferedReader br = new BufferedReader(
    new InputStreamReader(System.in));
```

  - Chuyển dữ liệu từ bộ đệm vào xâu

```
String inValue = br.readLine( );
```

  - Chuyển dữ liệu từ xâu thành giá trị : sử dụng các lớp
    - Giá trị kiểu `int`: `Integer.parseInt(inValue)`
    - Giá trị kiểu long: `Long.parseLong(inValue)`
    - Giá trị kiểu float: `Float.parseFloat(inValue)`
    - Giá trị kiểu double: `Double.parseDouble(inValue)`

37

37

Ví dụ

```
/** The Addition class calculates the sum of two numbers */
import java.io.*;
public class Addition {
    /** The main method begins execution of Java application
    *@param args: input parameter
    */
    public static void main (String[] args) throws
            IOException{
       String inputData;
       BufferedReader br = new BufferedReader(new
            InputStreamReader(System.in));
       System.out.print("Enter the 1st number:");
       inputData = br.readLine();
       int number1 = Integer.parseInt(inputData);
```

38

38

Ví dụ (tiếp)

```
        System.out.print("Enter the 2nd number:");
        inputData = br.readLine();
        int number2 = Integer.parseInt(inputData);
        int sum = number1 + number2;
        System.out.println("The sum of two numbers: " + sum);
    }
}
```

39

39

Ví dụ - Giải thích

- Chú thích tạo tài liệu Javadoc:

```
/** Comment something */
```

- Sử dụng các gói thư viện được Java định nghĩa sẵn

```
import somepakage
```

  - `java.io`: Thư viện xuất nhập dữ liệu
- Bỏ qua các ngoại lệ (lỗi) trong khi thực thi chương trình:

```
throws someException
```

  - `IOException`: ngoại lệ xuất hiện khi xuất nhập dữ liệu

40

40

Nhập dữ liệu – Lớp Scanner

- Được cung cấp bởi thư viện java.ulti
- Quét luồng dữ liệu người dùng nhập từ bàn phím và phân tách các giá trị có kiểu dữ liệu nguyên thủy hoặc xâu.
- Rất hữu dụng

```
import java.util.Scanner; //The first line
...
        Scanner inputData = new Scanner(System.in);
        System.out.print("Enter the 1st number:");
        int number1 = inputData.nextInt();
        System.out.print("Enter the 2nd number:");
        int number2 = inputData.nextInt();

        int sum = number1 + number2;
        System.out.println("The sum of two numbers: " + sum);
```

41

41

4. CẤU TRÚC LẬP TRÌNH TRONG JAVA

42

42

Khối lệnh

- Nhóm các lệnh được bao bằng cặp dấu { }
- Thực hiện các lệnh một cách tuần tự
- Phạm vi của biến:
  - Biến chỉ có phạm vi sử dụng trong khối lệnh đã khai báo
  - Với các khối lệnh lồng nhau có khai báo biến trùng tên, biến ở khối lệnh trong được ưu tiên

```
{
   int n = 0;
   n = n + 1; //n =  1
   {
       int n = 10, m = 10;
       n = n + 1; //n = 11
   }
   m = m + 1; //Sai cú pháp
}
```

43

43

Cấu trúc if, if...else

```
if (boolean_expr){   //Nếu boolean_expression là true
       //do something
}
```

```
if (boolean_expr){   //Nếu boolean_expression là true
       //do task1
}
else {               //Nếu boolean_expression là false
       //do task2
}
```

44

44

Cấu trúc if...else lồng nhau

```
if (boolean_expr1){
        //do task1
}
else if (boolean_expr2) {
        //do task2
}
...

else if (boolean_exprN){
        //do taskN
        }
else{
                //do other task
        }
```

45

Cấu trúc switch

- `controlling_expr` phải trả về kiểu dữ liệu `byte, short, char int, String`
- `value`: giá trị có kiểu byte, short, char, int, String
- `break`: thoát khỏi cấu trúc `switch`
- `default`: các giá trị còn lại

```
switch (controlling_expr){
   case value1:      //do task1
                     break;
   case value2:      //do task 2
                     break;
                ...
   case valueN:      //do taskN
                     break;
   default:   //do other task
}
```

46

Ví dụ

```
switch (number){
   case 1:    System.out.print("One");
              break;
   case 2:    System.out.print("Two");
              break;
   case 3:    System.out.print("Three");
              break;
   default:   System.out.print("I don't know.");
}
```

47

47

Cấu trúc switch - Nhóm giá trị

```
switch (month){ }
   case 1:
   case 3:
   case 5:
   case 7:
   case 8:
   case 10:
   case 12:   System.out.print("The month has 31 days");
              break;
   case 4:
   case 6:
   case 9:
   case 11:   System.out.print("The month has 30 days");
              break;
   default:   System.out.print("The month has 28 or 29 days");
}
```

48

48

Cấu trúc while và do...while

- Thực hiện lặp đi lặp lại một công việc khi biểu thức `boolean_expr` còn có giá trị `true`

```
while(boolean_expr){
   //do something
}
```

- Công việc có thể không được thực hiện lần nào

```
do {
   //do something
} while(boolean_expr);
```

- Công việc được thực hiện tối thiểu 1 lần

49

49

Ví dụ

- Sử dụng cấu trúc while

```
int n, i, factorial = 1;
i = 1;
while(i < = n){
   factorial *= i;
   i++;
}
```

- Sử dụng cấu trúc do...while

```
int n, i, factorial = 1;
i = 1;
do{
   factorial *= i;
   i++;
} while(i < = n);
```

50

50

Cấu trúc for

- Cú pháp

```
for (start_expr; loop_condition; loop_change){
        //do something
}
```

- Trong đó:
  - `start_expr` : Biểu thức khởi tạo
  - `loop_condition` : Biểu thức điều kiện thực hiện vòng lặp
  - `loop_change` : Biểu thức thay đổi biến
- Ví dụ

```
int n, factorial = 1;
for (int i = 1; i <= n; i++)
        factorial *= i;
```

51

51

Các lệnh thay đổi cấu trúc vòng lặp

- **`continue`**
  - Bỏ qua việc thực hiện các câu lệnh nằm sau lệnh **continue** trong thân vòng lặp.
  - Chuyển sang thực hiện một vòng lặp mới
- **`break`**
  - Thoát khỏi vòng lặp ngay cả khi biểu thức điều kiện của vòng lặp vẫn còn được thỏa mãn.
- Hai dạng:
  - Không gán nhãn: Chuyển ra ngoài vòng lặp, thực hiện câu lệnh ngay sau vòng lặp
  - Gán nhãn: Chuyển ra ngoài vòng lặp, thực hiện câu lệnh tiếp theo sau vòng lặp được đánh dấu bởi nhãn

52

52

Ví dụ

```
int sum = 0;
outer: for(int i = 0; i < 10; i++){
   inner: for(int j = i; j < 10; j++){
       sum ++;
       if (j == 1) continue;
       if (j == 2) continue outer;
       if (j == 3) break;
       if (j == 4) break outer;
    } // terminate inner
} // terminate outer
System.out.println(" sum = " + sum);
```

53

53

5. MỘT SỐ CẤU TRÚC DỮ LIỆU CƠ BẢN

54

54

Mảng

- Mảng là tập hợp hữu hạn các phần tử cùng kiểu
- Số lượng phần tử xác định khi khai báo, không đổi
- Khai báo

```
DataType[] array = new DataType[size];
DataType array[] = new DataType[size];
DataType[] array = {value1, value2,...,valueN};
```

Trong đó:

`array`: Biến mảng

`size`: Số phần tử trong mảng, có thể sử dụng giá trị, biến, biểu thức

`Value1,... valueN` : các giá trị khởi tạo

55

55

Mảng nhiều chiều

- Được coi là mảng của các mảng:
  - Mảng 2 chiều: mảng các mảng 1 chiều
  - Mảng 3 chiều: mảng của các mảng 2 chiều
- Khai báo mảng 2 chiều:

```
DataType[][] array = new DataType[size1][size2];
DataType array[][] = new DataType[size1][size2];
DataType[][] array = {value11,value12,...,value1N}
                     {value21,value22,...,value2N};
```

- Tương tự cho mảng nhiều chiều khác

56

56

Thao tác với mảng

- Phương thức `length()`: Lấy số phần tử của mảng
- Truy cập vào phần tử của mảng 1 chiều: `array[index]`

`index`: chỉ số, bắt đầu từ 0

- Truy cập vào phần tử của mảng nhiều chiều:
  `array[index1][index2]...[indexN]`

57

57

Ví dụ - Tìm kiếm trên mảng

```
/** The Search class lookup a key in the array . A message
is displayed to notify the index of the key in the array*/
import java.util.Scanner;
public class Search {
    /** The main method begins execution of Java application
    *@param args: input parameter
    */
    public static void main (String[] args) {
       int size = 0;
       Scanner inputData = new Scanner(System.in);
       System.out.print("Enter the size of the array:");
       size = inputData.nextInt();
       int[] intArr = new int[size];
```

58

58

Ví dụ: Tìm kiếm trên mảng(tiếp)

```
        //Enter value for each of element
        System.out.println("Enter value for array:);
        for(int i = 0; i < size; i++){
            System.out.print("intArr[" + i + "] = ");
            intArr = inputData.nextInt();
        }

        //Look up the key
        int key = 0;
        System.out.println("Enter value for key:);
        key = inputData.nextInt();
        boolean found = false;
```

59

59

Ví dụ: Tìm kiếm trên mảng(tiếp)

```
        for(int i = 0; i < size; i++){
           if (intArr[i] == key){
               System.out.println("Found " + key +"at " + i);
               found = true;
               break;
           }
        }
        //Notify if could not find the key
        if(!found)
           System.out.println ("Could not find " + key);
    }
}
```

60

60

String trong Java

- Được xây dựng như là một lớp trong Java
- Khai báo một đối tượng

```
String strObject = new String(literalString);
String strObject = new String(char[] charArr)
```

Hoặc đơn giản hơn:

```
String strObject = literalString;
```

- Trong đó:

`strObject` : đối tượng

`literalString`: Giá trị hằng xâu ký tự

`charArr`: Mảng ký tự

- Ví dụ: `String myUniversity = "HUST";`
- Sử dụng phương thức: `strObject.method()`

61

61

Các phương thức của String

- `char charAt(int index)`: trả về ký tự có chỉ số index
  - Chỉ số bắt đầu từ 0
- `int length()`: trả về kích thước của xâu
- `String toLowerCase()`: chuyển thành chữ thường
- `String toUpperCase()`: chuyển thành chữ hoa
- `int compareTo(String anotherString)`: so sánh hai xâu theo thứ tự từ điển, ưu tiên chữ thường. Trả về:
  - = 0: 2 xâu giống nhau
  - < 0: xâu nhỏ hơn `anotherString`
  - > 0: xâu lớn hơn `anotherString`
- `int compareToIgnoreCase(String str)`: so sánh không kể chữ hoa, chữ thường

62

62

Các phương thức của String (tiếp)

- `boolean equals (Object object)`: so sánh với một đối tượng bất kỳ
  - Trong Java, một đối tượng bất kỳ đều có thể chuyển thành String
- `boolean equalsIgnoreCase(String anotherString)`: so sánh với một đối tượng bất kỳ, không phân biệt chữ hoa chữ thường
- `char[] toCharArray`: chuyển xâu thành mảng ký tự
- `String concat(String str)`: ghép nội dung `str` vào cuối xâu
- `int indexOf(int ch)`: trả về chỉ số của ký tự đầu tiên có giá trị bằng `ch`
- `int indexOf(String str)`: trả về vị trí của `str` trong xâu

63

63

Các phương thức của String (tiếp)

- `String replace (char oldChar, char newChar)`: trả về xâu mà tất cả ký tự `oldChar` trong xâu thành `newChar`
- `String replace (String oldString, String newString)`
- `String replaceFirst(String oldString, String newString)`
- `String[] split (String regex)`: chia xâu thành các xâu con bởi xâu `regex`
- `String trim()`: trả về xâu được loại bỏ dấu cách ở đầu và cuối

64

64

Các phương thức gọi trực tiếp từ lớp String

- `static String copyValueOf(char[] data)`: trả về xâu có chứa các ký tự của mảng `data`
- `static String copyValueOf(char[] data, int offset, int count)`: trả về xâu có chứa count ký tự từ chỉ số offset của mảng `data`
- `static String format()`: trả về xâu hiển thị giá trị của đối tượng bất kỳ theo định dạng
- `static String valueOf(Object)`: trả về xâu chứa nội dung của một đối tượng nào đó

65