

Bài 3. Hàm/Phương thức. Các cấu trúc điều khiển

THCS 4: Lập trình cơ bản với Java

Đỗ Thanh Hà, Nguyễn Thị Minh Huyền

Bộ môn Tin học
Khoa Toán - Cơ - Tin học
Trường Đại học Khoa học Tự nhiên

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

1. Định nghĩa hàm/phương thức mới

```
public static void NAME()  
{  
    STATEMENTS;  
}
```

2. Sử dụng hàm/phương thức đã định nghĩa: **NAME();**

```
class NewLine {  
    public static void newLine(){  
        System.out.println("");  
    }  
  
    public static void twoLines(){  
        newLine();  
        newLine();  
    }  
  
    public static void main(String args[]){  
        System.out.println("Line 1");  
        twoLines();  
        System.out.println("Line 2");  
    }  
}
```

Hàm 1 đối/tham biến

```
public static void NAME(TYPE NAME){  
    STATEMENTS;  
}
```

- Gọi hàm: **NAME**(**Arg**)
 - **Arg** là biểu thức có kiểu **TYPE**

Ví dụ (1)

```
class Square {  
    public static void printSquare(int x){  
        System.out.println(x*x);  
    }  
  
    public static void main(String args[]){  
        int value = 2;  
        printSquare(value); // 4  
        printSquare(3); // 9  
        printSquare(value*2); //16  
    }  
}
```

Ví dụ (2)

```
class Square2 {  
    public static void printSquare(int x){  
        System.out.println(x*x);  
    }  
  
    public static void main(String args[]){  
        printSquare(3.2); // error ???  
        printSquare("hello"); //error ???  
    }  
}
```

Hàm nhiều đối

```
public static void NAME(TYPE NAME, TYPE NAME ){  
    STATEMENTS;  
}
```

- Gọi hàm: **NAME**(*arg1*, *arg2*);

```
class Multiply {  
    public static void printProduct(double x, double y){  
        System.out.println(x*y);  
    }  
  
    public static void main(String args[]){  
        printProduct(2,2); // 4.0  
        printProduct(3,4); // 12.0  
    }  
}
```

```
public static TYPE NAME([...]){  
    STATEMENTS;  
    return EXPRESSION;  
}
```

- Không trả lại giá trị: void

Ví dụ (1)

```
class Square2 {  
    public static void printSquare(double x){  
        System.out.println(x*x);  
    }  
  
    public static void main(String args[]){  
        printSquare(5); // 25.0  
    }  
}
```

Ví dụ (2)

```
class Square3 {  
    public static double square(double x){  
        return x*x;  
    }  
  
    public static void main(String args[]){  
        System.out.println(square(5)); // 25.0  
        System.out.println(square(2)); // 4.0  
    }  
}
```

- Phạm vi của một biến là từ vị trí khai báo biến tới cuối khối lệnh (nằm giữa cặp `{}`) chứa định nghĩa biến
- Các đối của hàm có phạm vi là hàm đó

Ví dụ (1)

```
class SquareChange {  
    public static void printSquare(int x){  
        System.out.println("printSquare x = " +x);  
        x = x * x ;  
        System.out.println("printSquare x = " +x);  
    }  
  
    public static void main(String args[]){  
        int x = 5;  
        System.out.println("main x = " +x); // main x = 5  
        printSquare(x);  
        // printSquare x = 5  
        // printSquare x = 25  
        System.out.println("main x = " +x); // main x = 5  
    }  
}
```

Ví dụ (1)

```
class BlockVariable {  
    public static void main(String args[]){  
        int outer = 1;  
        {  
            int inner = 2;  
            System.out.println("inner = " + inner);  
            // inner = 2  
            System.out.println("outer = " + outer);  
            // outer = 1  
        }  
        int inner = 3;  
        System.out.println("inner = " + inner);  
        // inner = 3  
        System.out.println("outer = " + outer);  
        // outer = 1  
    }  
}
```

Ví dụ (2)

```
class Scope{
    public static void main(String args[]){
        int x = 5;
        if (x==5){
            int x = 6; // Error: variable x is already
                       defined
            int y = 72;
            System.out.println("x = "+x + "y="+y);
        }
        System.out.println("x = "+x + "y="+y);
        // Error: cannot find symbol (symbol: variable y)
    }
}
```

Vì sao dùng hàm/phương thức?

- Hàm/phương thức: tổ chức chương trình thành các khối
- Các chương trình lớn được xây dựng nên từ hàm/phương thức nhỏ
- Các phương thức có thể được phát triển, kiểm thử và tái sử dụng một cách riêng biệt
- Người sử dụng phương thức không cần biết nó hoạt động cụ thể như thế nào
 - tính “trừu tượng hoá”

Các cấu trúc điều khiển

- Lệnh điều kiện if ... else
- Lệnh lựa chọn switch
- Lệnh lặp while
- Lệnh lặp do ... while
- Lệnh lặp for

Lệnh điều kiện if ... else

■ Dạng thiếu

```
if(condition){  
    statements  
}
```

■ Dạng đủ

```
if(condition){  
    statements  
} else{  
    statements  
}
```

Lệnh điều kiện if ... else (2)

■ Chuỗi if... else

```
if(condition){  
    statements  
} else if(condition){  
    statements  
} else if(condition){  
    statements  
} else{  
    statements  
}
```

Nhắc lại: các phép toán so sánh

- $x > y$: x lớn hơn y
- $x < y$: x nhỏ hơn y
- $x \geq y$: x lớn hơn hoặc bằng y
- $x \leq y$: x nhỏ hơn hoặc bằng y
- $x == y$: x bằng y
 - bằng: `==`
 - phép gán: `=`

Nhắc lại: các phép toán logic

■ &&: AND

■ ||: OR

```
if (x > 6) {  
    if (x < 9) {  
        ...  
    }  
}  
}
```

→

```
if ( x > 6 && x < 9) {  
    ...  
}
```

Ví dụ (1): Dạng thiếu

```
class ConditionTest{
    public static void test(int x){
        if(x >5){
            System.out.println(x + ">5");
        }
    }
    public static void main(String args[]){
        test(2);
        test(5);
        test(6);
    }
}
```

Ví dụ (2): Dạng đủ

```
class ConditionTest{
    public static void test(int x){
        if(x >5){
            System.out.println(x + ">5");
        } else {
            System.out.println(x + "not > 5");
        }
    }
    public static void main(String args[]){
        test(2);
        test(5);
        test(6);
    }
}
```

Ví dụ (3): Dạng chuỗi

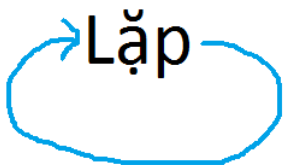
```
class ConditionTest{
    public static void test(int x){
        if(x >5){
            System.out.println(x + " > 5");
        } else if (x == 5){
            System.out.println(x + " = 5");
        } else {
            System.out.println(x + " < 5");
        }
    }
    public static void main(String args[]){
        test(2);
        test(5);
        test(6);
    }
}
```


Lệnh lựa chọn switch

```
switch(variable){  
    case <value_1>: statements_1  
                    break;  
    .....  
    case <value_n>: statements_n  
                    break;  
    default: statement_n+1  
}
```

Ví dụ: SwitchCaseExample.java

Lặp

A blue circular arrow is drawn around the word 'Lặp'. The arrow starts from the right side of the word, goes down, then left, and finally up to point at the beginning of the word, indicating a loop or repetition.

```
static void main (String args[]){  
    System.out.println("Rule #1");  
    System.out.println("Rule #2");  
    System.out.println("Rule #3");  
}
```

- Thực hiện công việc này với 200 luật?

Lệnh lặp

■ lặp while

```
while(condition){  
    statements  
}
```

■ lặp do.. while

```
do{  
    statements  
}while(condition);
```

■ lặp for

```
for (initialization; condition; update){  
    statements  
}
```

■ Điều khiển trong vòng lặp: **continue** và **break**;

Ví dụ (1)

```
int i = 0;
while (i < 3){
    System.out.println("Rule #" + i);
    i = i + 1;
}
```

- Chú ý, chương trình lặp phải kết thúc sau một số hữu hạn lần lặp

Ví dụ (2)

```
int i = 0;
do{
    System.out.println("Rule #" + i);
    i = i + 1;
}while (i<3)
```

Ví dụ (3)

```
for (int i =0; i < 3; i++){  
    System.out.println("Rule #" + i);  
}
```

- Chú ý: $i++$ được thay bằng $i = i + 1$

Ví dụ (4)

```
for (int i =0; i < 100; i++){  
    if (i == 50)  
        break;  
    System.out.println("Rule #" + i);  
}
```

Ví dụ (5)

```
for (int i =0; i < 100; i++){  
    if (i%2 == 0)  
        continue;  
    System.out.println("Rule #" + i);  
}
```

Ví dụ (6)

```
for (int i =0; i < 3; i++){  
    for (int j =0; j < 4; j++){  
        System.out.println(i + " " + j);  
    }  
}
```
