**BÁO CÁO**

**Họ tên:Nguyễn Văn Nguyên**

**Mã sinh viên:2020605636**

**Lớp:HTTT02**

**Khóa:K15**

* **Java Tutorial**
* **Java là gì?**

**-**Khái niệm của java: Java là một ngôn ngữ lập trình và là một nền tảng. Java là một ngôn ngữ bậc cao, mạnh mẽ, hướng đối tượng và an toàn.

**-**Ứng dụng: theo sun, 3 tỷ ứng dụng chạy trên java. Có nhiều thiết bị mà Java hiện đang được sử dụng. Một số trong số họ như sau:

+Desktop Applications such as acrobat reader, media player, antivirus, etc.

+Web Applications such as irctc.co.in, javatpoint.com, etc.

+Enterprise Applications such as banking applications.

+Mobile

+Embedded System

+Smart Card

+Robotics

+Games, etc.

**-**Các loại ứng dụng java:

+Standalone Application

+Web application

+ Enterprise Application

+Mobile Application

           -Java Platforms / Editions

+Java SE (Java Standard Edition)

+Java EE (Java Enterprise Edition)

**+**Java ME (Java Micro Edition)

+JavaFX

* **Lịch sử của java**

Các phiên bản của java:

* JDK Alpha and Beta (1995)
* JDK 1.0 (23rd Jan 1996)
* JDK 1.1 (19th Feb 1997)
* J2SE 1.2 (8th Dec 1998)
* J2SE 1.3 (8th May 2000)
* J2SE 1.4 (6th Feb 2002)
* J2SE 5.0 (30th Sep 2004)
* Java SE 6 (11th Dec 2006)
* Java SE 7 (28th July 2011)
* Java SE 8 (18th Mar 2014)
* Java SE 9 (21st Sep 2017)
* Java SE 10 (20th Mar 2018)
* Java SE 11 (September 2018)
* Java SE 12 (March 2019)
* Java SE 13 (September 2019)
* Java SE 14 (Mar 2020)
* Java SE 15 (September 2020)
* Java SE 16 (Mar 2021)
* Java SE 17 (September 2021)
* Java SE 18 (to be
* **Features of java**

1. [Simple](https://www.javatpoint.com/features-of-java#Simple)
2. [Object-Oriented](https://www.javatpoint.com/features-of-java#Object-Oriented)
3. [Portable](https://www.javatpoint.com/features-of-java#Portable)
4. [Platform independent](https://www.javatpoint.com/features-of-java#Platform-independent)
5. [Secured](https://www.javatpoint.com/features-of-java#Secured)
6. [Robust](https://www.javatpoint.com/features-of-java#Robust)
7. [Architecture neutral](https://www.javatpoint.com/features-of-java#Architecture-neutral)
8. [Interpreted](https://www.javatpoint.com/features-of-java#Interpreted)
9. [High Performance](https://www.javatpoint.com/features-of-java#High-Performance)
10. [Multithreaded](https://www.javatpoint.com/features-of-java#Multithreaded)
11. [Distributed](https://www.javatpoint.com/features-of-java#Distributed)
12. [Dynamic](https://www.javatpoint.com/features-of-java#Dynamic)

* **C++ vs Java**

Có nhiều điểm khác nhau và tương đồng giữa java và c++.

* **Hello java problem**

Các tham số được sử dụng trong chương trình đầu tiên của java:

* Class
* Public
* Static
* Void
* Main
* String[]  args  or String args[]
* System.out.println()
* **Program internal**

Trong thời gian chạy các bước sau thực hiện:

* Classloader: Đó là hệ thống con của JVM được sử dụng để tải các tệp lớn.
* Bytecode: Kiểm tra các đoạn mã cho mã bất hợp pháp có thể vi phạm quyền truy cập đối với các đối tượng.
* Interpreter: Đọc luồng bytecode sau đó thực thi các hướng dẫn.
* **Cách đặt đường dẫn trong java**

Có hai cách:

- Tạm thời

- Vĩnh viễn

* **JDK, JRE and JVM**
* JVM: là một máy trừu tượng. Nó được gọi là một máy ảo vid nó không tồn tại về mặt vật lý.
* JRE: là một từ viết tắt của môi trường thời gian chạy Java. Môi trường thời gian chạy Java là một tập các công cụ phần mềm được sử dụng để phát triển các ứng dụng java.
* JDK: là một từ viết tắt của bộ phát triển java. Bộ phát triển java là một môi trường phát triển phần mềm được sử dụng để phát triển các ứng dụng và applet java. Nó tồn tại về thể chất. Nó chứa các công cụ phát triển JRE+.
* **JVM: Java virtual machine**

Kiến trúc JVM:

* Classloade
* Bootstrap classloader
* Extension classloader
* System/application classloader
* Class (method) area
* Heap
* Stack
* Program counter register
* Native method stack
* Execution engine
* A virtual processor
* Interpreter
* Just-in-time compiler
* Java native interface
* **Java variables**

Loại biến:

* Local variable: biến cục bộ, khai báo bên trong phần thân của phương pháp.
* Instance variable: biến thể hiện, được khai báo bên trong lớp nhưng bên ngoài phần thân của phương pháp.
* Static variable: biến tĩnh
* **Kiểu dữ liệu trong java**
* Kiểu nguyên thuỷ:
* Boolean
* Char
* Byte
* Short
* Int
* Long
* Float
* Double
* Kiểu không nguyên thuỷ:
* Classes
* Interface
* Arrays
* **Unicode system**

Là một mã hoá nhân vật tiêu chuẩn quốc tế phổ quát có khả năng đại diện hầu hết các ngôn ngữ viết của thế giới.

* **Operators**

-Operator trong java là một biểu tượng được sử dụng để thực hiện các hoạt động. ví dụ +, -, \*, /, …

-Có nhiều toán tử trong java:

* Toán tử unary
* Toán tử số học
* Toán tử thay đổi
* Toán tử quan hệ
* Toán tử bitwise
* Toán tử logic
* Toán tử ternary
* Toán tử gán
* **Java keywords**

Danh sách từ khoá java: abstract, boolean, break, byte, case, catch, char, class default, do, continue, double, else, enum, extends, final, finally, float, if, import, implements, instanceof, int, interface, long, native, new, null, package, private, protected, public, return, short, static, strictfp, switch, synchronizied, this, throws, transient, try, void, volatile, while.

* **Control statements**
* **Java control statements**

Java cung cấp ba loại câu lệnh điều khiển luồng điều khiển:

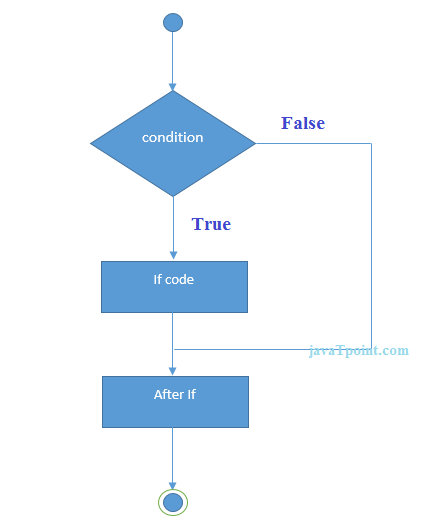
* Câu lệnh điều khiển:
* Câu lệnh if
* Câu lệnh if đơn giản
* Câu lệnh if – else
* Câu lệnh if-else-if
* Câu lệnh if lồng nhau
* Câu lệnh switch
* Câu lệnh lặp:
* Vòng lặp do while
* Vòng lặp while
* Vòng lặp for
* Vòng lặp for-each
* Câu lệnh nhảy
* Câu lệnh break
* Câu lệnh continue
* **Java if- else**

Câu lệnh java if được sử dụng để kiểm tra điều kiện. Nó kiểm tra điều kiện: đúng hoặc sai. Các loại câu lệnh if khác nhau trong java:

* Câu lệnh if
* Syntax:

if(condition) {

statement 1; //executes when condition is true



}

Hình 1.1: câu lệnh if

* Ví dụ

package baitapbuoi1;

import java.util.Scanner;

public class vidu {

public static void main(String[] args) {

int n;

Scanner ss= new Scanner(System.in);

n = ss.nextInt();

if(n % 100 == 0)

{

System.out.println("n là bội của 100");

}

}

}

* Câu lệnh if-else
* Syntax

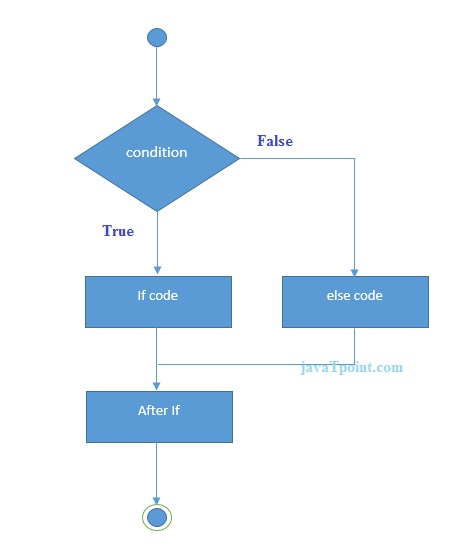
**if**(condition){

//code if condition is true

}**else**{

//code if condition is false

}



Hình 1.2: if- else

* Ví dụ

package baitapbuoi1;

import java.util.Scanner;

public class vidu {

public static void main(String[] args) {

int n;

Scanner ss= new Scanner(System.in);

n = ss.nextInt();

if(n % 2 == 0)

{

System.out.println("n là số chẵn");

}

else

{

System.out.println("n là số lẻ");

}

}

}

* Câu lệnh if-else-if ladder

**if**(condition1){

//code to be executed if condition1 is true

}**else** **if**(condition2){

//code to be executed if condition2 is true

}

**else** **if**(condition3){

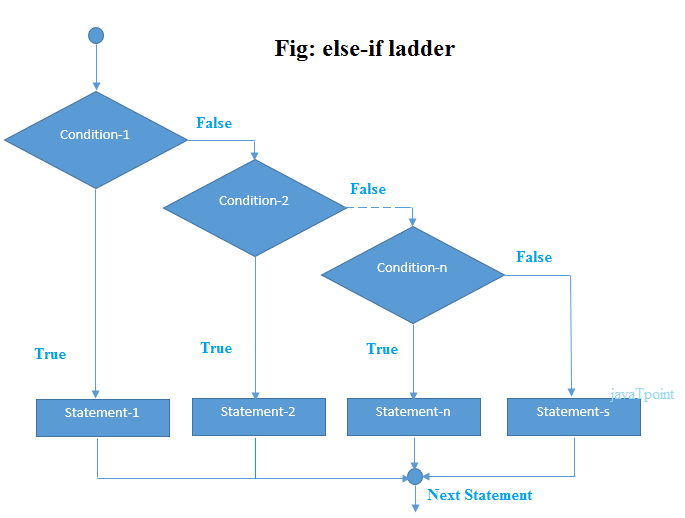
//code to be executed if condition3 is true

}

...  **else**{

//code to be executed if all the conditions are false

}



Hình 1.3: else- if ladder

* Ví dụ minh hoạ

package baitapbuoi1;

import java.util.Scanner;

public class vidu {

public static void main(String[] args) {

int diemtb;

Scanner ss= new Scanner(System.in);

diemtb = ss.nextInt();

if(diemtb < 5)

{

System.out.println("loại yếu");

}

else if( diemtb == 5)

{

System.out.println("loại trung bình");

}

else

{

System.out.println("loại giỏi");

}

}

}

* Câu lệnh if lồng nhau

**if**(condition){

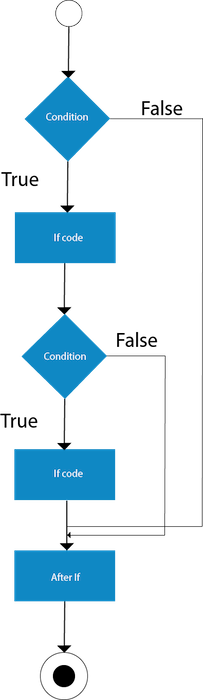
//code to be executed

**if**(condition){

//code to be executed

}

}



Hình 1.4: câu lệnh if lồng nhau

* Ví dụ minh hoạ

public class JavaNestedIfExample {

public static void main(String[] args) {

    //Creating two variables for age and weight

    int age=20;

    int weight=80;

    //applying condition on age and weight

    if(age>=18){

        if(weight>50){

            System.out.println("You are eligible to donate blood");

        }

    }

}}

* **Java switch**

Câu lệnh switch trong java thưc thi một lệnh từ nhiều điều kiện. Nó hoạt động viws các kiểu dữ liệu byte, short, int, long, enum, string và một số kiểu wrpaper như Byte, Short, Int và Long.

Nói cách khác, câu lệnh switch kiểm tra sự bằng nhau của một biến với nhiều giá trị khác nhau.

**switch**(expression){

**case** value1:

 //code to be executed;

**break**;  //optional

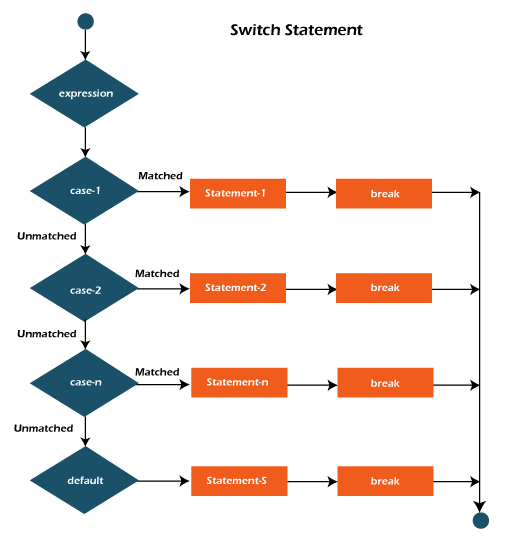
**case** value2:

 //code to be executed;

**break**;  //optional

......

**default**:



Hình 1.5: cấu trúc switch

* **Java for loop**

Vòng lặp for được sử dụng để lặp lại một phần chương trình nhiều lần.

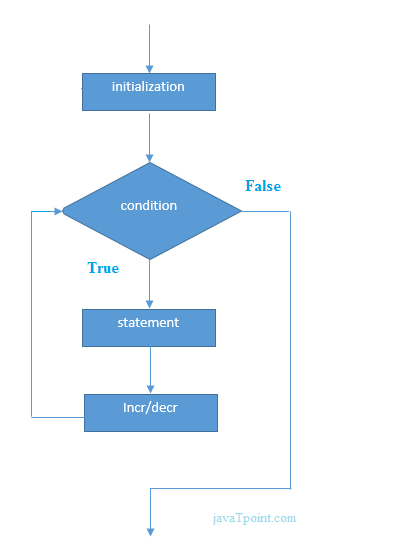
Có ba loại vòng lặp for trong java:

* Simple for loop: vòng lặp for đơn giản. Nó bao gồm bốn phần:
* Initiaization: Đây là điều kiện ban đầu được thực thi một lần khi vòng lặp bắt đầu.
* Condition: đây là điều kiện thứ hai được thực thi mỗi lần để kiểm tra điều kiện của vòng lặp for.
* Increment/ decrement: tăng hoặc giảm giá trị của biến.
* Statement: câu lệnh của vòng lặp được thực thi mỗi lần chạy đến điều kiện thứ hai trở thành sai.

**for**(initialization; condition; increment/decrement){

//statement or code to be executed

}



Hình 1.6: Simple for loop

* Ví dụ minh hoạ

public class ForExample {

public static void main(String[] args) {

    //Code of Java for loop

    for(int i=1;i<=10;i++){

        System.out.println(i);

    }

}

}

* For- each or enhanced for loop: vòng lặp for-each hay enhanced for loop

**for**(data\_type variable : array\_name){

//code to be executed

}

* Ví dụ minh hoạ

public class ForEachExample {

public static void main(String[] args) {

    //Declaring an array

    int arr[]={12,23,44,56,78};

    //Printing array using for-each loop

    for(int i:arr){

        System.out.println(i);

    }

}

}

* Labeled for loop: vòng lặp for được đặt nhãn
* Syntax:

labelname:

**for**(initialization; condition; increment/decrement){

//code to be executed

}

* Ví dụ minh hoạ:

public class LabeledForExample2 {

public static void main(String[] args) {

    aa:

        for(int i=1;i<=3;i++){

            bb:

                for(int j=1;j<=3;j++){

                    if(i==2&&j==2){

                        break bb;

                    }

                    System.out.println(i+" "+j);

                }

        }

}

}

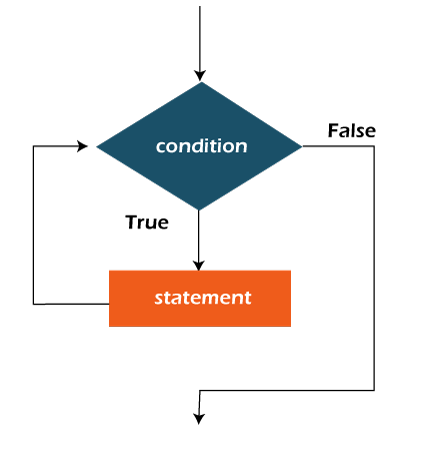
* **Java while loop**

Vòng lặp while trong java đuqocj sử dụng để lặp lại một phần của chương trình cho đến khi điều kiện boolean được chỉ định trở thành true.

While (condition){

While the condition is true

}



Hình 1.7: vòng lặp while

* Ví dụ minh hoạ

public class WhileExample {

public static void main(String[] args) {

    int i=1;

    while(i<=10){

        System.out.println(i);

    i++;

    }

}

}

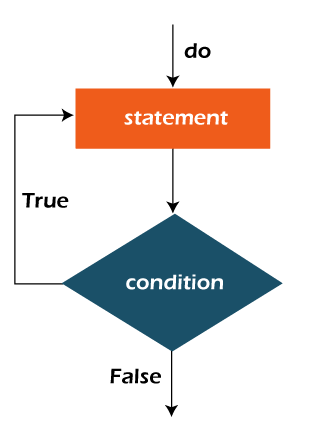
* **Java do-while loop**

Do {

//code to be executed

//update statement

}while (condition)



Hình 1.8: vòng lặp do- while

* Ví dụ minh hoạ

public class DoWhileExample {

public static void main(String[] args) {

    int i=1;

    do{

        System.out.println(i);

    i++;

    }while(i<=10);

}

}

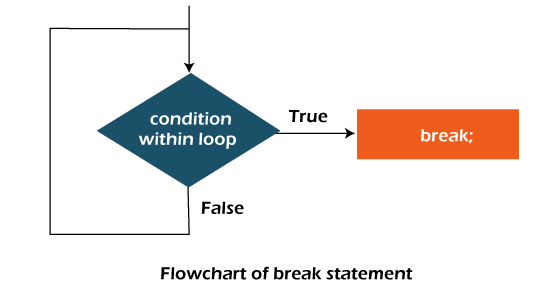
* **Java break**

Chúng ta có thể sử dụng lệnh break trong java cho tất cả các loại vòng lặp như vòng lặp for, vòng lặp while và vòng lặp do-while.

-Syntax:

Jump-statement;

Break;



Hình 1.8: lệnh break

-Ví dụ minh hoạ:

**public** **class** BreakExample {

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

    //using for loop

**for**(**int** i=1;i<=10;i++){

**if**(i==5){

            //breaking the loop

**break**;

  }

        System.out.println(i);

    }

}

}

* **Java continue**

Chúng ta có thể sử dụng lệnh continue trong java cho tất cả các vòng lặp như vòng lặp for, vòng lặp while và vòng lặp do-while.

-Syntax:

Jump-statement;

Continue;

-ví dụ minh hoạ

**public** **class** ContinueExample {

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

   //for loop

**for**(**int** i=1;i<=10;i++){

**if**(i==5){

            //using continue statement

**continue**;//it will skip the rest statement

        }

        System.out.println(i);

    }

}

}

* **Java comments**

Có ba loại comments trong java

* Comment trên một dòng: ‘//’
* Comment trên nhiều dòng: đặt giữa /\* and \*/
* Comment tài liệu: /\*\* and \*/
* **Java programs**

Danh sách các chương trình trong java:

* Java basic programs
* Java number programs
* Java array programs
* Java matrix programs
* Java String programs
* Java searching and sorting programs
* Java conversio programs
* Pattern programs
* Singly linked list programs
* Circular linked list programs
* Doubly linked list programs
* Tree programs

1. Tài liệu tham khảo

<https://www.javatpoint.com/java-tutorial>