**Cú pháp Java cơ bản**

Java là ngôn ngữ lập trình hướng đối tượng, có tính đa nền tảng nhờ chạy trên JVM ( Java Virtual Machine )

+ Hướng đối tương : Dựa trên các khái niệm như lớp, đối tượng, kê thừa, đa hình.

+ Tính độc lập nền tảng : Java viết một lần, chạy mọi nơi.

+ Quản lí bộ nhớ tự động : Hệ thống dọn dẹp rác ( Garbage Collector ).

Cấu trúc cơ bản của chương trình Java :



1. **Kiểu dữ liệu và biến, Hằng số:**

|  |  |  |
| --- | --- | --- |
| Kiểu | Mô tả | Ví dụ |
| int | Số nguyên | int x = 10; |
| float | Số thực chính xác thấp | float y = 31.4; |
| char | Ký tự Unicode | char c = ‘A’ |
| boolean | True/False logic | boolean isOK = true |

* 1. **Hằng số**

Để khai báo hằng số ta sử dụng từ khóa static final đặt trước tên hằng số:

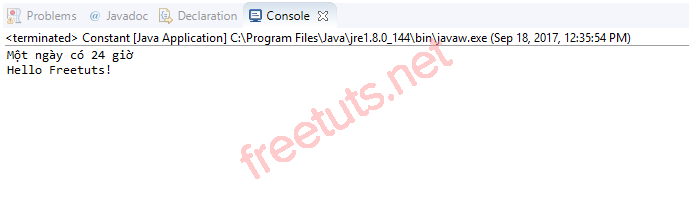
 [Phạm vi truy cập] static final [kiểu dữ liệu] [tên hằng số] = [giá trị];

Dưới đây là chương trình minh họa cách sử dụng hằng số.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | package constant;    public class Constant {      // Khai báo hằng số      public static final int HOUR\_OF\_DAY = 24;      public static final String CHUOI = "Hello Freetuts!";        public static void main(String[] args) {          System.out.println("Một ngày có " + HOUR\_OF\_DAY + " giờ");          System.out.println(CHUOI);      }    } |

**Kết quả sau khi thực thi chương trình:**



****2. Xuất (output) trong Java****

Trong Java, chúng ta có thể sử dụng các hàm bên dưới để xuất dữ liệu ra màn hình.

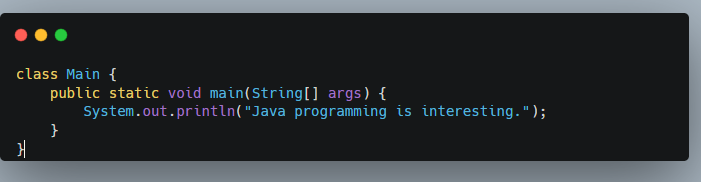
* **System.out** là chuẩn đầu ra, **System.err** dùng để hiển thị thông báo lỗi trong Java.

System.out.println();

hoặcSystem.out.print();

hoặcSystem.out.printf();

#### ****Ví dụ:****



##### ****Kết quả****

Java programming is interesting.

#### ****Sự khác nhau giữa các hàm println(), print() và printf()****

* print(): in ra một chuỗi bên trong dấu ngoặc kép “”
* println(): in ra một chuỗi giống như hàm print(). Sau đó, tự động xuống dòng mới.
* printf(): in ra chuỗi với việc cung cấp các định dạng khác nhau giống như [hàm printf trong C/C++](https://www.programiz.com/cpp-programming/library-function/cstdio/printf).

#### ****Ví dụ hàm print() và println()****

class Main {

public static void main(String[] args) {

System.out.println("1. println ");

System.out.println("2. println ");

System.out.print("1. print ");

System.out.print("2. print");

}}

##### ****Kết quả****

1. println

2. println

1. print 2. Print

****3. Nhập (input) trong Java****

Trong Java, chúng ta có thể sử dụng đối tượng của lớp ****Scanner****để lấy giá trị được nhập vào bởi người dùng.

Đầu tiên, cần ****import****gói java.util.Scanner.

**System.in** là chuẩn đầu vào trong Java

import java.util.Scanner;

Sau đó, cần tạo một đối tượng của lớp ****Scanner****. Đối tượng này sẽ giúp chúng ta lấy giá trị nhập vào của người dùng.

// create an object of ScannerScanner input = new Scanner(System.in);

// take input from the userint number = input.nextInt();

#### ****Các phương thức của lớp Scanner****

|  |  |
| --- | --- |
| ****Phương thức**** | ****Mô tả**** |
| nextByte() | *Đọc một số nguyên kiểu byte* |
| nextShort() | *Đọc một số nguyên kiểu short* |
| nextInt() | *Đọc một số nguyên kiểu int* |
| nextLong() | *Đọc một số nguyên kiểu long* |
| nextFloat() | *Đọc một số kiểu float* |
| nextDouble() | *Đọc một số kiểu double* |
| next() | *Đọc một string kết thức trước một ký tự trắng* |
| nextLine() | *Đọc một line of text (kết thúc bằng phím Enter)* |

Ví dụ:



##### ****Kết quả****

Enter an integer: 21

You entered 21

Enter float: 2.3

Float entered = 2.3

Enter double: 5.5

Double entered = 5.5

Enter text: HelloJava!

Text entered = HelloJava!

**Lưu ý**: Chúng ta nên gọi hàm close() để đóng đối tượng ****Scanner****khi không sử dụng nữa.

1. **Toán tử** 
   1. **Toán tử số học**

Các toán hạng của các toán tử số học phải ở dạng số. Các toán hạng kiểu boolean không sử dụng được, các toán hạng ký tự cho phép sử dụng loại toán tử này. Một vài kiểu toán tử được liệt kê trong bảng dưới đây.

Giả sử chúng ta có biến số nguyên a = 10 và b = 20.

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Mô tả** | **Ví dụ** |
| + | Cộng Trả về giá trị là tổng của hai toán hạng | a + b sẽ là 30 |
| - | Trừ Trả về kết quả là hiệu của hai toán hạng. | a + b sẽ là -1 |
| \* | Nhân Trả về giá trị là tích của hai toán hạng. | a + b sẽ là 200 |
| / | Chia Trả về giá trị là thương của phép chia. | b / a sẽ là 2 |
| % | Phép lấy modul Giá trị trả về là phần dư của phép chia | b % a sẽ là 0 |
| ++ | Tăng dần Tăng giá trị của biến lên 1. Ví dụ a++ tương đương với a = a + 1 | a++ sẽ là 11 |
| -- | Giảm dần Giảm giá trị của biến 1 đơn vị. Ví dụ a-- tương đương với a = a - 1 | a-- sẽ là 9 |
| += | Cộng và gán giá trị Cộng các giá trị của toán hạng bên trái vào toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c += a tương đương c = c + a | a += 2 sẽ là 12 |
| -= | Trừ và gán giá trị Trừ các giá trị của toán hạng bên trái vào toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c -= a tương đương với c = c - a | a -= 2 sẽ là 8 |
| \*= | Nhân và gán Nhân các giá trị của toán hạng bên trái với toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c \*= a tương đương với c = c\*a | a \*= 2 sẽ là 20 |
| /= | Chia và gán Chia giá trị của toán hạng bên trái cho toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c /= a tương đương với c = c/a | a /= 2 sẽ là 5 |
| %= | Lấy số dư và gán Chia giá trị của toán hạng bên trái cho toán toán hạng bên phải và gán giá trị số dư vào toán hạng bên trái. Ví dụ c %= a tương đương với c = c%a | a /= 8 sẽ là 2 |

* 1. **Toán tử Bit**

Các toán tử dạng bit cho phép chúng ta thao tác trên từng bit riêng biệt trong các kiểu dữ liệu nguyên thuỷ.

|  |  |
| --- | --- |
| **Toán tử** | **Mô tả** |
| ~ | Phủ định NOT Trả về giá trị phủ định của một bít. |
| & | Toán tử AND Trả về giá trị là 1 nếu các toán hạng là 1 và 0 trong các trường hợp khác |
| | | Toán tử OR Trả về giá trị là 1 nếu một trong các toán hạng là 1 và 0 trong các trường hợp khác. |
| ^ | Toán tử Exclusive OR Trả về giá trị là 1 nếu chỉ một trong các toán hạng là 1 và trả về 0 trong các trường hợp khác. |
| >> | Dịch phải Chuyển toàn bộ các bít cuả một số sang phải một vị trí, giữ nguyên dấu của số âm. Toán hạng bên trái là số bị dịch còn số bên phải chỉ số vị trí mà các bít cần dịch. |
| << | Dịch trái Chuyển toàn bộ các bít cuả một số sang trái một vị trí, giữ nguyên dấu cuả số âm. Toán hạng bên trái là số bị dịch còn số bên phải chỉ số vị trí mà các bít cần dịch. |

* 1. **Các toán tử quan hệ**

Các toán tử quan hệ được sử dụng kiểm tra mối quan hệ giữa hai toán hạng. Kết quả của một biểu thức có dùng các toán tử quan hệ là những giá trị Boolean (logic “true” hoặc “false”). Các toán tử quan hệ được sử dụng trong các cấu trúc điều khiển.

|  |  |
| --- | --- |
| **Toán tử** | **Mô tả** |
| == | So sánh bằng Toán tử này kiểm tra sự tương đương của hai toán hạng |
| != | So sánh khác Toán tử này kiểm tra sự khác nhau của hai toán hạng |
| > | Lớn hơn Kiểm tra giá trị của toán hạng bên phải lớn hơn toán hạng bên trái hay không |
| < | Nhỏ hơn Kiểm tra giá trị của toán hạng bên phải có nhỏ hơn toán hạng bên trái hay không |
| >= | Lớn hơn hoặc bằng Kiểm tra giá trị của toán hạng bên phải có lớn hơn hoặc bằng toán hạng bên trái hay không |
| <= | Nhỏ hơn hoặc bằng Kiểm tra giá trị của toán hạng bên phải có nhỏ hơn hoặc bằng toán hạng bên trái hay không |

* 1. **Các toán tử logic**

Các toán tử logic làm việc với các toán hạng Boolean. Các toán tử quan hệ được sử dụng trong các cấu trúc điều khiển.

|  |  |
| --- | --- |
| **Toán tử** | **Mô tả** |
| && | Toán tử và (AND) Trả về một giá trị “Đúng” (True) nếu chỉ khi cả hai toán tử có giá trị “True” |
| || | Toán tử hoặc (OR) Trả về giá trị “True” nếu ít nhất một giá trị là True |
| ^ | Toán tử XOR Trả về giá trị True nếu và chỉ nếu chỉ một trong các giá trị là True, các trường hợp còn lại cho giá trị False (sai) |
| ! | Toán tử phủ định (NOT) Toán hạng đơn tử NOT. Chuyển giá trị từ True sang False và ngược lại. |

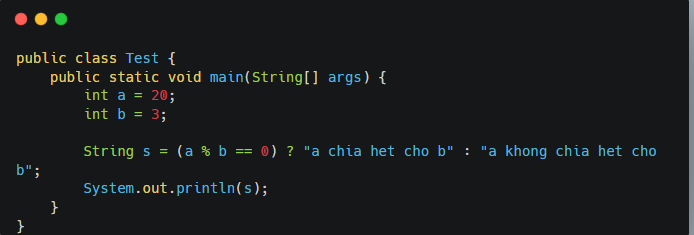
* 1. **Các toán tử điều kiện**

Toán tử điều kiện là một loại toán tử đặc biệt vì nó bao gồm ba thành phần cấu thành biểu thức điều kiện. Cú pháp:

|  |  |
| --- | --- |
| 1 | <biểu thức 1>   ?   <biểu thức 2>   :   <biểu thức 3>; |

* ****biểu thức 1:**** Biểu thức logic. Trả trả về giá trị True hoặc False
* ****biểu thức 2:**** Là giá trị trả về nếu <biểu thức="" 1="" style="box-sizing: border-box;">xác định là True
* ****biểu thức 3:**** Là giá trị trả về nếu <biểu thức="" 1="" style="box-sizing: border-box;">xác định là False

Ví dụ:



Kết quả: a khong chia het cho b

* 1. **Toán tử gán**

Toán tử gán (=) dùng để gán một giá trị vào một biến và có thể gán nhiều giá trị cho nhiều biến cùng một lúc.

Ví dụ:



**5. Cú pháp comment code trong Java.**

Trong Java, chúng ta có 3 cách comment code như sau:

**Cú pháp comment code trong Java**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | /\*   \* Comment nhiều dòng   \* Dòng 1   \* Dòng 2   \* ...   \* Dòng n   \*/    // Comment 1 dòng    /\*\*   \* Comment đặc biệt dùng để tạo Java code documentation ở định dạng HTML (Java Document)  \*/ |

Dưới đây là đoạn chương trình minh họa về comment code.

**Ví dụ.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | package comment\_code;    /\*\*   \*   \* @author keeps   \* Đây là comment dùng để tạo Java document ở định dạng HTML từ source code Java.   \* Đoạn code này sẽ được trình biên dịch bỏ qua khi thực thi.   \*/  public class CommentCodeInJava {        public static void main(String[] args) {          // System.out.println("Đây là comment 1 dòng. Dòng này sẽ được trình biên dịch bỏ qua khi thực thi");            /\*           \* System.out.println("Đây là comment nhiều dòng. Dòng này sẽ được trình biên dịch bỏ qua khi thực thi");           \*/          System.out.println("Đây là ví dụ về 3 cách comment code trong Java. Dòng này sẽ được trình biên dịch thực thi.");      }    } |

Kết quả biên dịch chương trình:

