Variables y operadores en Java

Profesor: Germán Ernesto Zapata Ledesma
Ayudante: Dicter Tadeo García Rosas

Junio 2023

0.1 Instrucciones

Lee cuidadosamente la información presentada en este documento y realiza el cuestionario o práctica que se adjunta.

0.2 Introducción

Anteriormente vimos los tipos de datos existentes en java y un par de ejemplos sobre como se declaran las variables que utilizamos en nuestros programas, sin embargo es necesario ahondar más en la declaración de las mismas ya que representan una parte importante del desarrollo de un programa.

Recordando lo que es una variable "Las variables Java son un espacio de memoria en el que guardamos un determinado valor (o dato).", con esto en mente veamos como se define una variable y cual es la estructura que siguen:

[Modificador de acceso] [Tipo de variable] [Identificador];

0.3 Identificadores

Debido a que ya hemos visto los tipos de variables, podemos explorar más acerca de los identificadores y por el momento dejaremos para después el modificador de acceso. Los identificadores son secuencias de caracteres alfanuméricos que pueden iniciar con una letra o con u guión bajo, por ejemplo:

$$cuadrado \qquad x83 \qquad \_variable$$

Estos ejemplos anteriores son completamente válidos dentro de las convenciones para declarar variables, pero ¿qué pasa cuando nuestra variable necesita ser más descriptiva y por lo tanto el identificador más largo?, java sigue la sintaxis Camel Case la cual nos indica que si el identificador une dos o mas palabras la unión a partir de la segunda palabra se hará poniendo en mayúsculas la primer letra de dicha palabra, puede sonar rebuscado pero los siguientes ejemplos pretenden ayudar a clarificar este concepto:

$$calculaArea \qquad miVariable \qquad diaDelMes$$

Aunque es importante señalar que a pesar que no hay alguna restricción sobre el uso de diéresis o acentos en la creación de identificadores se prefiere no usar a fin de que cualquier persona pueda ejecutar dicho programa sin depender de la configuración del sistema en el que se haya hecho.

De la misma manera que las variables tienen identificadores los archivos en los cuales escribimos nuestros programas java y las clases también los tienen, para este tipo de identificadores se utiliza la convención de la primer letra mayuscula y camel case, es decir:

$$Persona \qquad FiguraGeometrica$$

Son nombres de archivo y clases válidos, debemos notar que el nombre del archivo y la clase debe ser el mismo ya que al momento de compilarlo la JVM nos devolverá un error ya que no encuentra dicha clase a compilar.

Así como nosotros podemos crear variables con el identificador que deseemos, en java existen palabras reservadas del lenguaje las cuales no pueden ser utilizadas como identificador ya que tienen un significado particular para el lenguaje y en caso de usarse el compilador lanzaría un error. A continuación se muestra una tabla con las palabras reservadas de java:

abstract	continue	finally	int	public	throw
assert	default	float	interface	return	throws
boolean	do	for	long	short	transient
break	double	goto	native	static	true
byte	else	if	new	strictfp	try
case	enum	implements	null	super	void
catch	extends	import	package	switch	volatile
class	false	inner	private	synchronized	
const	final	instanceof	protected	this	while

Hasta ahora hemos visto los tipos de datos en Java y los identificadores, por lo que nuestras variables validas se ven de la siguiente forma:

$int\ total$; $String\ miNombre$; $boolean\ valordeVerdad$;
$double\ pi$; $char\ caracter$;

Todas estas declaraciones son válidas y hasta el momento su valor es "desconocido" aunque sabemos que tiene uno asignado por defecto, si queremos asignar valores particulares a dichas declaraciones podriamos hacerlo de la siguiente manera:

$int\ total = 12000$; $String\ miNombre = "Juan"$
$boolean\ valordeVerdad = true$; $double\ pi = 3.14$;
$char\ caracter = \ 'a'$;

0.4 Operadores

El tipo de dato asociado a nuestras variables determina las operaciones que podemos realizar con ellos, y dado que los tipos de datos primitivos no son objetos no existen métodos asociados a ellos, por lo cual la manera de trabajar con estos tipos de datos son mediante los operadores existentes para cada uno de ellos.

Un operador y sus operandos constituyen una expresión, el propósito de una expresión es calcular un valor, así mismo dicho resultado puede ser utilizado como operando de una nueva expresión sucesivamente, siempre y cuando el valor devuelto sea el esperado por el operando. Para cada tipo de dato existen operandos particulares los cuales encapsulan las operaciones realizadas evitando calculos extraños como podría ser: true + 8, 5.3 + 'a', etc. Dentro de los operadores podemos clasificarlos de acuerdo a la función que realizan, por ejemplo, operadores lógicos, operadores aritmeticos, operador de asignación, operadores de relación, operadores de asignación compuesta.

0.4.1 Operadores aritméticos

Este tipo de operadores tiene como objetivo realizar calculos con los tipos de datos numéricos como lo son int, double, float, long. También son conocidos

como operadores binarios, estos son:

Java Operator	**Descripción**
+	En la suma, los valores de dos operandos se suman.
-	En la resta, el valor del segundo operando se resta del valor del primer operando.
*	En la multiplicación, se multiplican dos operandos.
/	En la división, el primer operando se divide por el valor del segundo operando.
%	El operando mod calcula el valor restante de una división.
+	El más también puede utilizarse como signo positivo. Sin embargo, en la mayoría de los casos, no se requiere este operador Java.
-	El menos también se puede utilizar como signo negativo.

Veamos entonces un ejemplo de cada operador:

$Suma:$
$int\ a = 10;$
$int\ b = 6;$
$int\ resultado\ =\ a + b$ $//devuelve\ 16$
$Resta:$
$int\ a = 23;$
$int\ b = 18;$
$int\ resultado\ =\ a - b$ $//devuelve\ 5$
$Multiplicacion:$
$int\ a = 8;$
$int\ b = 9;$
$int\ resultado\ =\ a * b$ $//devuelve\ 72$
$Division:$
$int\ a = 100;$
$int\ b = 4;$
$int\ resultado\ =\ a/b$ $//devuelve\ 25$
$Modulo\ o\ Residuo:$
$int\ a = 10;$
$int\ b = 2;$
$int\ resultado\ =\ a + b$ $//devuelve\ 0$

Los siguientes operadores que estudiaremos son los operadores unarios o de autoincremento y autodecremento, dentro de estas dos categorías podemos encontrar variantes de los mismos operadores, pues estas variantes satisfacen casos de uso en los que es necesario aumentar o decrementar la misma variable antes de ser usada, preincremento y postincremento respectivamente, veamos un uso de dichos operadores:

$Autoincremento$
$int\ a\ =\ 7;$
$a++;$ $//devuelve\ 8$
$Autodecremento$
$int\ a\ =\ 7;$
$a--;$ $//devuelve\ 6$

Como mencionamos anteriormente, dichos operadores pueden utilizarse para cambiar el valor de una variable antes o después de ser usada para un cálculo, estos cambios los realizamos mediante la posición en la que ubicamos dicho operador, ya sea de manera prefija o postfija, en el ejemplo anterior vimos que el operador se encuentra de manera postfija:

Cambiando los ejemplos anterior a notación prefija:

$Autoincrementoprefijo$
$int\ a\ =\ 7;$
$++a;$ $//devuelve\ 8$
$Autodecremento$
$int\ a\ =\ 7;$
$--a;$ $//devuelve\ 6$

0.4.2 Operadores de Relación

Estos operadores nos permiten definir comparaciones entre operandos de cualquier tipo (int, float, double, etc) de acuerdo a la relación de orden entre los valores comparados y devuelven un valor booleano (true o false). Se muestran a continuación:

Operador	Descripción
<	Menor que
<=	Menor o igual que
==	Igual que
>=	Mayor o igual que
>	Mayor que
!=	Diferente de

A continuación se muestran ejemplos de dichos operadores:

$<$

```
int a = 5;
int b = 6;
a < b;                    //devuelve true
```

$>$

```
int a = 6;
int b = 10;
a > b;                    //devuelve false
```

$<=$

```
int a = 3;
int b = 9;
a <= b;                   //devuelve true
```

$>=$

```
int a = 10;
int b = 6;
a >= b;                   //devuelve true
```

$==$

```
int a = 18;
int b = 17;
a == b;                   //devuelve false
```

$!=$

$int\ a\ =\ 25;$

$int\ b\ =\ 32;$

$a!=b;$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $//devuelve\ true$

0.4.3 Operadores lógicos

Los operadores lógicos trabajan con datos o expresiones booleanas un valor booleano como resultado (true o false), los operadores pra este tipo de datos se presentan a continuación:

Operador	Descripción
&&	Conjunción
||	Disyunción
!	Negación

&&	true	false
true	true	false
false	false	false

||	true	false
true	true	true
false	true	false

!	
false	true
true	false

A continuación se muestran algunos ejemplos:

&&

$boolean\ a\ =\ false;$

$boolean\ b\ =\ true;$

$a\ \&\&\ b;$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $//devuelve\ false$

||
$boolean\ a\ =\ false;$
$boolean\ b\ =\ true;$
$a\ ||\ b;$ $//devuelve\ true$
!
$boolean\ a\ =\ true;$
$!b;$ $//devuelve\ false$

Es importante resaltar que el operador && detiene su ejecución al encontrar el primer false, pues devuelve falso, mientras que el operador || se detiene cuando encuentra el primer true y lo devuelve.

0.4.4 Operador para cadenas

Como hemos visto anteriormente, las cadenas no son un tipo de dato primitivo dentro del lenguaje java, sin embargo dado el extenso uso que se les dan se ha optimizado para ser usado de manera similar a los datos primitivos, por lo cual también cuenta con su propio operador el cual concatena dos cadenas cualquiera, este operador es la +, por ejemplo:

+
$String\ nombre\ =\ "Juan";$
$String\ apellido\ =\ "Alvarez";$
$String\ nombreCompleto = nombre + apellido;$ $//devuelve\ Juan\ Alvarez$
+
$int\ edad\ =\ 23;$
$String\ miEdad = "Hola,\ tengo"\ +\ edad\ +\ "\ años\ de\ edad\ Edad"$
$//devuelve\ Hola,\ tengo\ 23\ años\ de\ edad$

0.4.5 Operador de asignación compuesta

algunas ocasiones se desea actualizar el valor de una variable a partir del valor que ya tiene asignado pero añadiendo alguna operación dentro de la actual-

ización, comunmente escribiriamos dicha asignación de la siguiente manera $a = a+1$, dicha operación suma 1 al valor de a y lo reasigna a la misma variable, podemos abreviar dicha expresión mediante el operador de asignación compuesta de la siguiente manera:

$+=$
$int\ b\ =\ 5;$
$b+=5;$ $\quad //equivalente\ a\ b\ =\ b\ +\ 1$
$-=$
$int\ c\ =\ 5;$
$c+=2;$ $\quad //equivalente\ a\ c\ =\ c\ -\ 2$
$/=$
$int\ c\ =\ 10;$
$c/=2;$ $\quad //equivalente\ a\ c\ =\ c\ /\ 2$
$*=$
$int\ c\ =\ 5;$
$c*=5;$ $\quad //equivalente\ a\ c\ =\ c\ *\ 5$