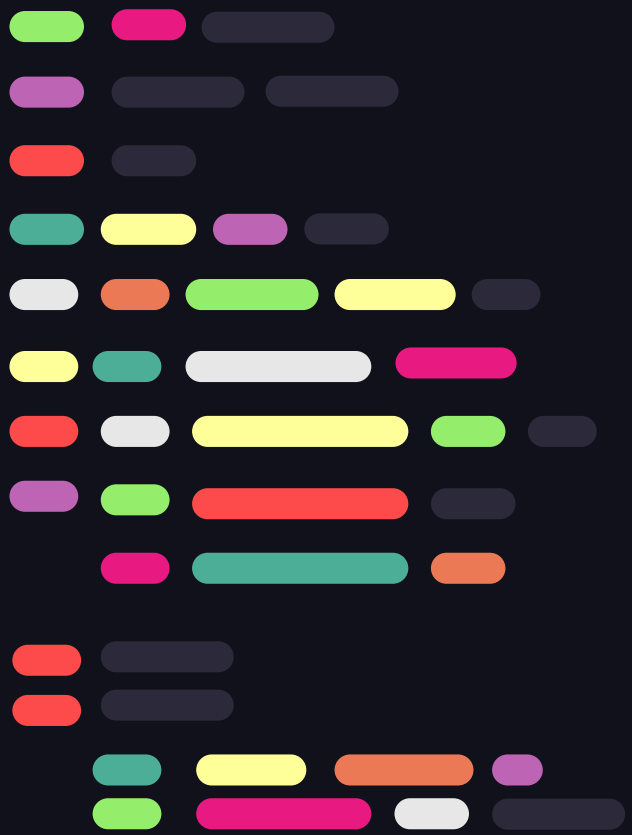
ESCUELA DE INGENIERÍA EN CIENCIAS Y SISTEMAS
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

ECYS | DITT
DESARROLLO DE TRANSFERENCIA TECNOLÓGICA

Día, Fecha:	Viernes, 26/07/2024
Hora de inicio:	09:00

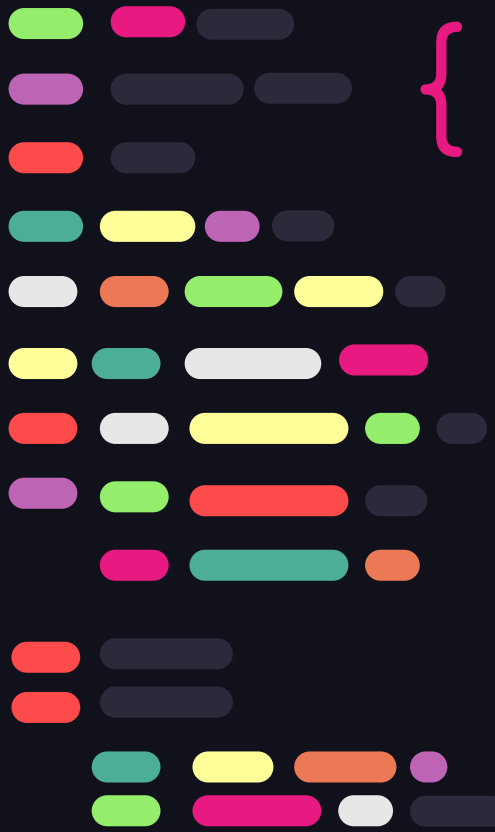
Introducción a la Programación y Computación 1 [B]

Josué Rodolfo Morales Castillo



Clase 1- Agenda

- Foro No. 1
- Asignación DTT (Formulario)
- Fundamentos de Programación y JAVA
- Lectura de Tarea No.1



Fundamentos de Programación y JAVA

Parte 1

Algoritmos

¿Qué son?

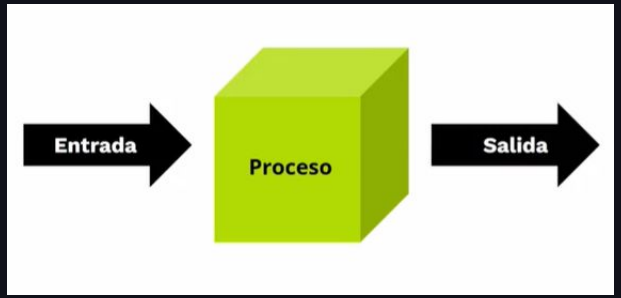
Un algoritmo es una secuencia de instrucciones finitas que llevan a cabo una serie de procesos para dar respuesta a determinados problemas. Es decir, un algoritmo resuelve cualquier problema a través de unas instrucciones y reglas concisas, mostrando el resultado obtenido.

¿Para qué sirven?

Permiten al programador resolver el problema antes de escribirlo en un lenguaje de programación que entienda la máquina u ordenador. Antes de escribir el código de un programa hay que resolver con un algoritmo el problema que se nos plantea

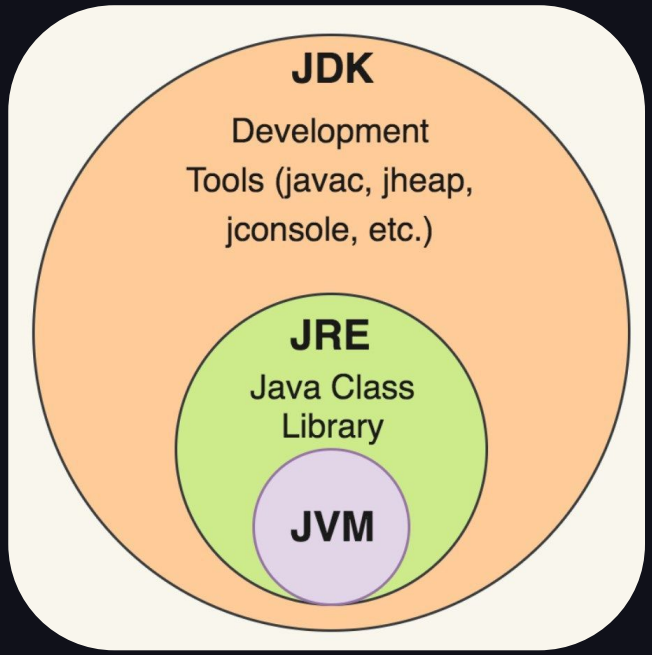
Características

- El algoritmo debe finalizar después de un número finito de pasos. No debe ejecutarse indefinidamente.
- Cada paso del algoritmo debe estar claramente definido y comprensible. No debe haber ambigüedad en las instrucciones.
- Debería ser independiente del lenguaje de programación específico. Un algoritmo bien definido debe poder implementarse en cualquier lenguaje.



¿Qué es Java?

Es un lenguaje de programación de propósito general que fue desarrollado por Sun Microsystems a mediados de la década de 1990 y posteriormente adquirido por Oracle Corporation. Es conocido por ser un lenguaje de programación versátil, orientado a objetos, portable y multiplataforma.



JDK

Java Development Kit, utilizado por los desarrolladores para escribir, compilar, depurar y ejecutar aplicaciones Java.

JRE

Java Runtime Environment, es un conjunto de software que proporciona la JVM y otras bibliotecas de tiempo de ejecución necesarias para ejecutar aplicaciones Java

JVM

Virtual Machine, interpreta y ejecuta instrucciones expresadas en un código binario el cual es generado por el compilador del lenguaje Java.

Características de Java

Orientado a Objetos: lenguaje de programación orientado a objetos, lo que significa que se basa en el concepto de objetos, que encapsulan datos y comportamientos.

Portabilidad: La JVM permite ejecutar programas Java en cualquier dispositivo o sistema operativo que la tenga instalada.

Manejo Automatico de Memoria: Java utiliza un recolector de basura (garbage collector) para gestionar automáticamente la memoria.

Simple: funcionalidad de un lenguaje potente, derivado de C y C++, pero sin las características menos usadas

Comentarios

Una Linea

```
// Comentario de una sola línea
```

Multilinea

```
/* Este es un comentario
     tradicional.  Puede
     dividirse en muchas líneas */
```

Variables

Una variable es un espacio de almacenamiento con un nombre asociado que se utiliza para representar y manipular datos en un programa. Las variables tienen un tipo de dato que define el tipo de valores que pueden almacenar y operaciones que se pueden realizar con ellos.

tipoDeDato nombreDeVariable;

tipoDeDato nombreDeVariable = valorInicial;

```
int edad; // Declaración de la variable
edad = 25; // Asignación de un valor a la variable
```

```
double precio = 49.99; // Declaración e
//inicialización de la variable en una sola línea
```

Tipos de datos en Java

Se utilizan para declarar variables y especificar qué tipo de valores pueden almacenarse en esas variables. Se dividen en dos:

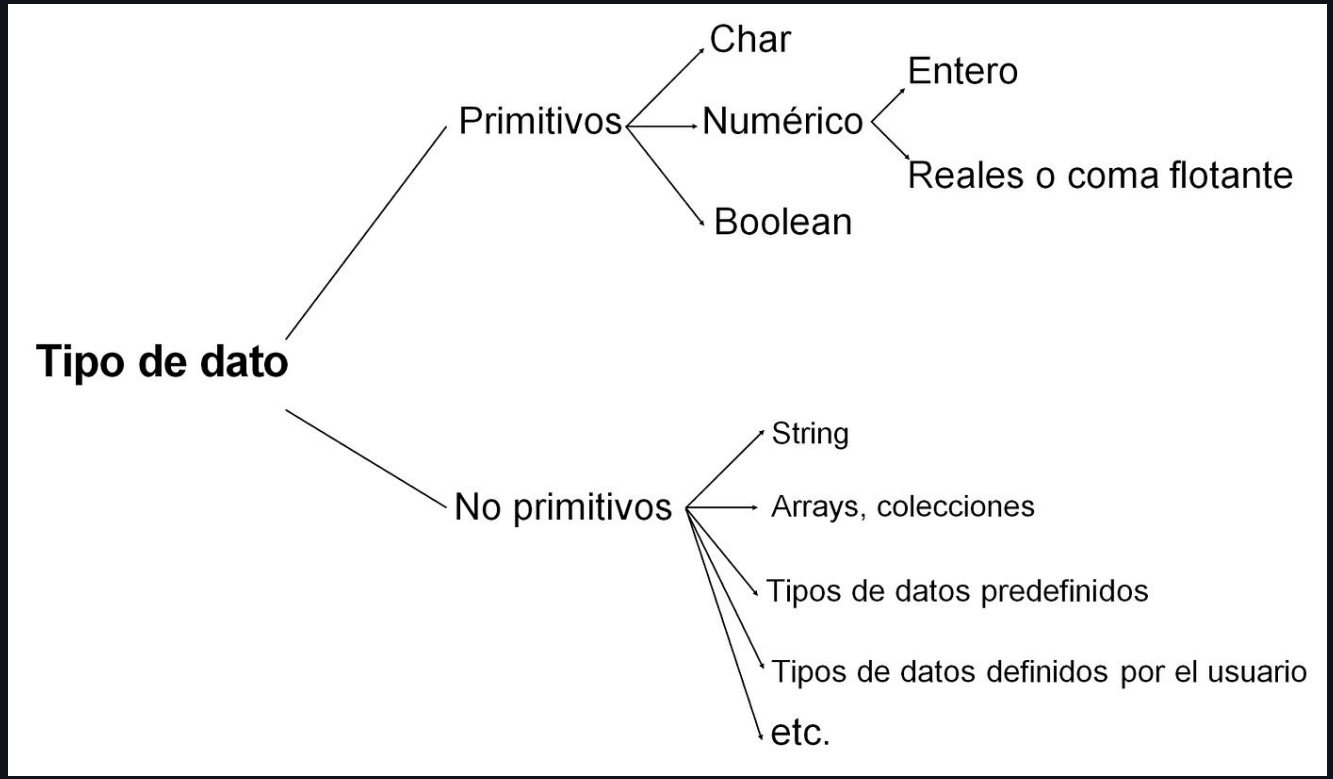
Primitivos: Se refiere a un tipo de dato básico que no es derivado de ningún otro tipo.

- Representan valores simples
- Ocupan una cantidad fija de memoria
- Se accede directamente al valor almacenado

No Primitivos: Se refiere a un tipo que no almacena directamente el valor, sino que almacena una referencia a un objeto en memoria.

- Almacenan referencias a objetos.
- No ocupan una cantidad fija de memoria.
- Suelen tener métodos.

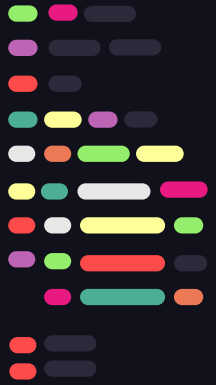
Tipos de datos en Java



Ejemplo de datos Primitivos

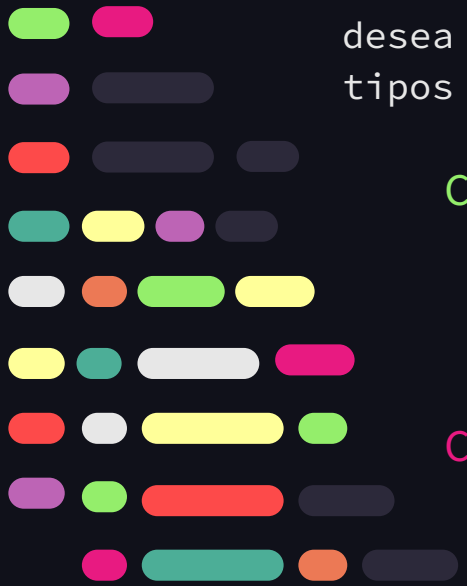
	NOMBRE	TIPO	OCUPA	RANGO APROXIMADO
TIPOS PRIMITIVOS (sin métodos; no son objetos; no necesitan una invocación para ser creados)	byte	Entero	1 byte	-128 a 127
	short	Entero	2 bytes	-32768 a 32767
	int	Entero	4 bytes	$2*10^9$
	long	Entero	8 bytes	Muy grande
	float	Decimal simple	4 bytes	Muy grande
	double	Decimal doble	8 bytes	Muy grande
	char	Carácter simple	2 bytes	---
	boolean	Valor true o false	1 byte	---

Ejemplo de datos No Primitivos



TIPOS OBJETO (con métodos, necesitan una invocación para ser creados)	Tipos de la biblioteca estándar de Java	String (cadenas de texto) Muchos otros (p.ej. Scanner, TreeSet, ArrayList…)
	Tipos definidos por el programador / usuario	Cualquiera que se nos ocurra, por ejemplo Taxi, Autobus, Tranvia
	arrays	Serie de elementos o formación tipo vector o matriz. Lo consideraremos un objeto especial que carece de métodos.
	Tipos envoltorio o wrapper (Equivalentes a los tipos primitivos pero como objetos.)	Byte
		Short
		Integer
		Long
		Float
		Double
		Character
		Boolean

Casteos



Se refiere a la conversión de un tipo de dato a otro. También se conoce como "casting". Este proceso es necesario cuando se desea operar o asignar valores entre variables de diferentes tipos de datos. Existen dos tipos principales:

Casting Implícito

Este tipo de casting se produce de manera automática por el compilador cuando no hay pérdida de información en la conversión.

Casting Explícito

A diferencia del casting implícito, el casting explícito requiere la intervención explícita del programador.

Operadores Aritméticos

Son utilizados para realizar operaciones matemáticas en variables numéricas.

Operador	Descripción	Ejemplo de expresión	Resultado del ejemplo
-	operador unario de cambio de signo	-4	-4
+	Suma	2.5 + 7.1	9.6
-	Resta	235.6 - 103.5	132.1
*	Producto	1.2 * 1.1	1.32
/	División (tanto entera como real)	0.050 / 0.2 7 / 2	0.25 3
%	Resto de la división entera	20 % 7	6

Operadores Compuestos

Operador	Operación	Equivale a
++	a++	a = a + 1
--	a--	a = a - 1
+=	a+=b	a = a + b
-=	a-=b	a = a - b
*=	a*=b	a = a * b
/=	a/=b	a = a / b
%=	a%=b	a = a % b

Operadores Relacionales

Son utilizados para comparar dos valores y determinar la relación que existe entre ellos. Estos operadores retornan un valor booleano (true o false) dependiendo de si la relación especificada es verdadera o falsa.

X=5; Y=3

Operación	Operador	Ejemplo	Comparación	Resultado/Binario
Mayor que	>	X > Y	¿Es X mayor que Y?	Verdadero (1)
Menor que	<	X < Y	¿Es X menor que Y?	Falso (0)
Mayor o igual que	>=	X >= Y	¿Es X mayor o igual que Y?	Verdadero (1)
Menor o igual que	<=	X <= Y	¿Es X menor o igual que Y?	Falso (0)
Igual a	==	X == Y	¿Es X igual a Y?	Falso (0)
Diferente a	!=	X != Y	¿Es X diferente a Y?	Verdadero (1)

Operadores Lógicos

Se refieren a las acciones que puedes realizar con valores booleanos utilizando operadores lógicos. Estos operadores permiten combinar o modificar expresiones booleanas para evaluar condiciones más complejas.

Operadores	Uso de los símbolos	Significado
&& (Y)	a == b && c > d	Se cumple si 'a' es igual que 'b' **y** 'c' es mayor que 'd'
|| (O)	a < b || c < d	Se cumple si 'a' es menor que 'b' **o** 'c' es menor que 'd'
! (No)	!(a==b)	Se cumple si 'a' **no** es igual a 'b'

Prioridad entre operadores Aritméticas

Operador	Nombre	Nivel Jerárquico
( )	Paréntesis	1er nivel
^	Potencia	2do nivel
X o *	Multiplicación	3er nivel
÷ o /	División	
+	Suma	4to nivel
-	Resta	

Prioridad entre operadores Lógicos

Operadores Lógicos		
Nombre	Operador	Jerarquía
NOT	!	1
AND	&&	2
OR	||	3

Input y Output

Input

Se refiere a los datos o información que un programa recibe.

La clase principal para la entrada de datos es Scanner. Esta clase se encuentra en el paquete java.util y proporciona varios métodos para leer diferentes tipos de datos.

Output

Se refiere a la información o resultados que un programa envía, por ejemplo la impresión en consola.

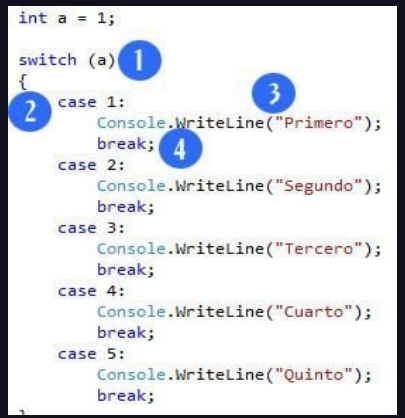
Para la salida, se utiliza principalmente la clase System.out. Se puede utilizar los métodos print (sin salto de línea) y println (con salto de línea).

{ .. Estructuras de control

Son elementos que permiten controlar el flujo de ejecución de un programa. Estas estructuras determinan el orden en el que se ejecutan las instrucciones dentro de un programa y permiten la toma de decisiones.

if else if else switch

```
if(Condición){
    acción;
}else if(otraCondición){
    otraAcción;
}else if(otraMás){
    másAcciones;
}else{
    últimaAcción;
}
```

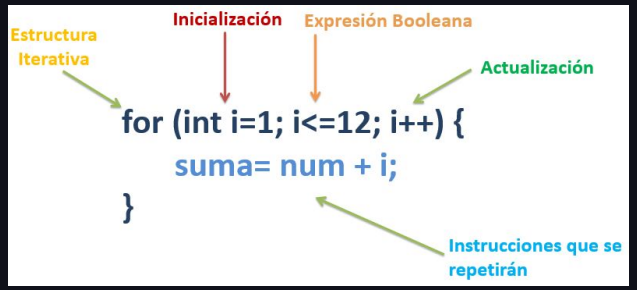


* } ..

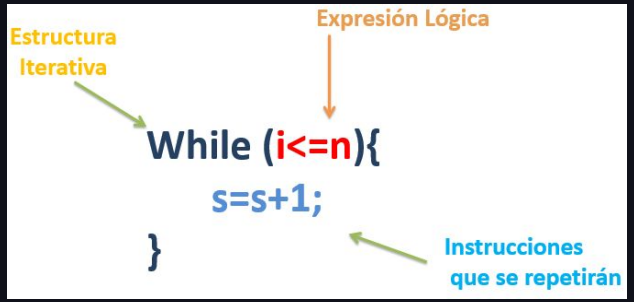
Ciclos

Es la repetición de ciertos bloques de código.

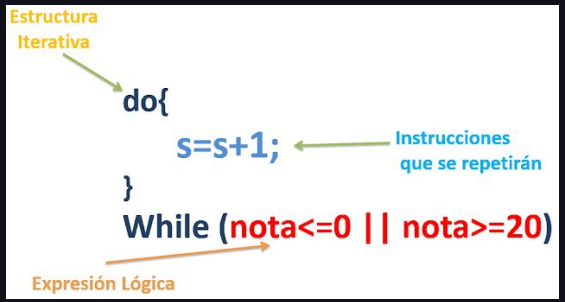
for



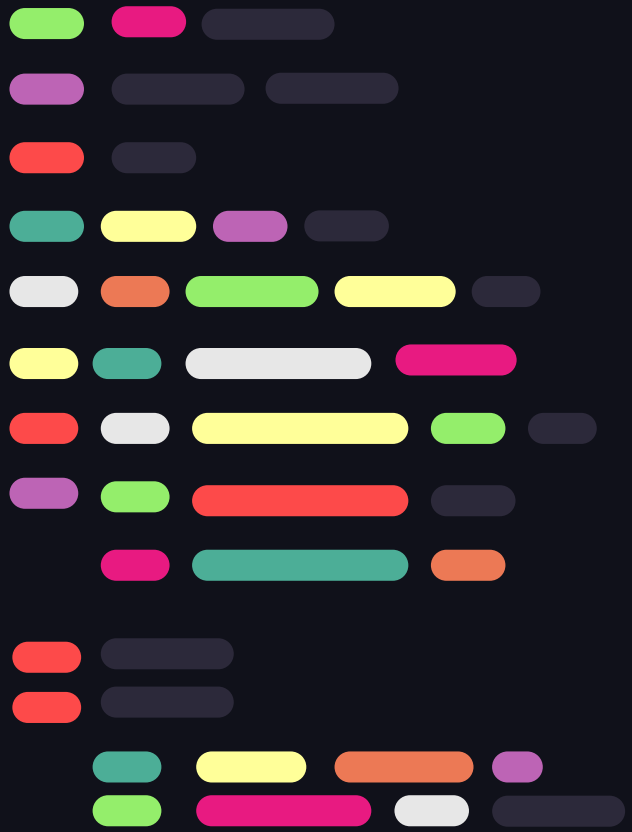
while



do-while




SI ALGUNA
DUDA TIENES
PREGUNTAR
AHORA DEBES



{ ..

Thanks

} ..