**JAVA BÁSICO**

Profesor: Eduardo Corral Muñoz

[Eoi\_eduardo@corral.es](mailto:Eoi_eduardo@corral.es)

C:\Program Files\Java\jdk-19\bin

La carpeta .idea no se toca. (dentro de IntelliJ)

El archivo Introduccion.iml no se toca (dentro de IntelliJ), recoge información de nuestro progreso.

Se trabaja en src (IntelliJ)

public class Main -- Público, para todos. Todo se organiza en clases (trozos de programa). Main es el nombre que hemos decidido darle, pero valdría cualquiera. Ha de ser igual al nombre bajo la carpeta src (creamos la clase con click derecho sobre src y elegimos nombre). Ha de empezar cada palabra del título de la clase con mayúscula, y el resto con minúscula para indicar separaciones entre palabras.

La “{“ indica donde empieza el código, y el “}” dónde acaba.

Hay que usar un método, como:

Public static void main (String[] args) {}----- main es la parte del programa donde afecta, el {} indica qué se hace en ese método

Click derecho sobre el símbolo de play junto a la línea para iniciar.

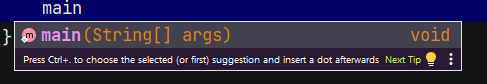
Exit code 0 --- Han habido 0 errores

Carpeta out: Contiene lo que hemos iniciado (Run) y no es editable.

Las carpetas visibles en la columna de la izquierda de IntelliJ aparecen también en la carpeta de destino del Proyecto.

En src escribimos la fuente en (escribimos y guardamos en .java) lenguaje de “alto nivel”. En out se compila (se guarda como .class), traduciéndolo a algo entendible por la máquina (lenguaje de bajo nivel). Fuente, compilación y ejecución (se emplea el JRE para interpretar y ejecutar el código)

Escribiendo “main”, ya aparece la opción de mostrar texto



Escribiendo “sout”, ya aparece el comando para escribir mi texto.



Para escribir en dos líneas: (Texto\nTexto)

Webs de consulta:

El API de Java

W3schools

Stackoverflow

Alto nivel: Lenguaje entendible por humanos

Bajo nivel: Lenguaje entendible por máquinas

Hay que comentar siempre en el código para facilitar la interpretación posterior nuestra parte o por otros programadores.

Herramienta para diagramas de flujo:

draw.io

Pseudocódigo:

Diagrama simplificado escrito en comentario. Ver diagrama normal como “Diagrama suma” en carpeta Java básico

Variable: Contenedor de datos que pueden ir cambiando durante la ejecución del programa: nombre= valor

a = 3

b = 5

Pueden cambiar durante el programa con operaciones como a = a + 1 en determinados bucles u otras funciones.

Puede ser un número, verdadero/falso (booleano), “texto”

Hemos de declarar las variables.

Matrices: Son un objeto. Supongamos la matriz = {1,3,5}

La matriz[0] es 1. La matriz[1] es 3. La matriz [3] es 5.

Las matrices multidimensionales serían matriz = {(1,10),(2,20),..)

La matriz [1][1] es 20 El primer [] se refiere a cuál de las submatrices, y el segundo [] se refiere al valor dentro de la submatriz.

El ámbito es donde existen las variables. Pueden ser globales (en todo el programa) o locales (en una parte del programa).

Constantes: Tienen un valor definido e inalterable durante todo el programa.

Operadores: Hay matemáticos, lógicos, etc

25/01/2023

Variables y tipos de datos

Nombres:

Empiezan con minúscula (en forma de varias palabras miNombreEs)

Sólo caracteres Unicode

No usar palabras reservadas

No repetir variables en el mismo ámbito

char:

Se usa para declarar caracteres alfanuméricos de la tabla Unicode, y se usa ‘a’, en lugar de “a”, ya que las dobles comillas están reservadas para texto. Se puede declarar mediante su código como ‘\u0021’

Sistemas numéricos en Java

Decimal (como siempre): a=14

Binario (añadir 0b): a=00001110

Octal (añadir 0): a=016

Hexadecimal (añadir 0x): a=0x0E

Constantes:

Siempre se escriben todo en mayúsculas

Se escribe final antes del tipo de número con la fórmula:

final tipo nombre = valor

final float EDAD = 24

String:

Se declaran String nombre = “texto”

Se declaran String nombre = new String(“Texto”) para declarar un nuevo String.

Para incluir comillas en el texto, se utiliza \”

If:

Condición? valor1:valor2

El valor1 si se cumple, y el valor 2 si no

Int resultado = 5 > 10 ? 1 : 2

Clase wraper:

Clase de cada tipo primitivo para dotarlo de funciones necesarias para que funcione como un objeto. (int --- Integer) Se llaman igual, pero con la primera letra mayúscula.