Tipos de paradigma:

-Imperativos:Lista de sentencias C
-Orientados a objetos:Modela la realidad en base a clases y objetos c++ java
-Funcionales:Caracterisiticas funcionales JavaScript
-Lógicos. Basados en la lógica de predicados, se tiene una serie de reglas y usando la lógica a partir de una entrada (hechos) es capaz de inferir una solución

Compiladores e Interpretes

Un compilador traduce el un programa entero de un lenguaje de programación (llamado código fuente) a otro denominado lenguaje objeto El código generado para un sistema sólo funcionará para una arquitectura hardware y software determinadas
C/C++
La ejecución del programa objeto es mucho más rápida que sis se interpreta el programa fuente. El compilador tiene una visión global del programa la informacion de errores es mas detallada

Un interprete:traduce el código fuente linea a línea como se describe a continuación: primero traduce la primera línea, detiene la traducción y, seguidamente la ejecuta; lee la siguiente linea, detiene la traducción y la ejecuta, y así sucesivamente.

Un interprete necesita menos memoria que un compilador.

En caso de detectar un error durante el proceso de traducción el interprete detiene la ejecución del programa.
PHP Perl Python
Un interprete necesita menos memoria que un compilador
Permite una mayor interactividad con el código en tiempo de desarrollo

compilador justo a tiempo"). Este tipo de compilador,rompe con el modelo habitual de compilación y traduce el código del programa durante el tiempo de ejecución, al igual que el intérprete. De esta forma, la alta velocidad de ejecución típica de los compiladores se complementa con la simplificación del proceso de desarrollo

Utilización de los entornos integrados de desarrollo.

Al conjunto de líneas de texto escritas en un lenguaje de programación que contienen la lógica de la aplicación se le denomina código fuente

Al introducir el código fuente en un compilador se obtiene el código objeto concreto para una arquitectura, por último se realiza el "linkado" que une diferentes ficheros de código objeto como el generado por el código fuente y librerías externas necesarias para obtener el ejecutable.

El código fuente se crea utilizando editores de texto, que no se han de confundir con los procesadores de texto

Notepad/++
Nano

ID: Eclipse, Netbeans,VSCode

Su interfaz clara y directa que dispone de detección automática del código y ayudas en la escritura, sangrado y coloración para identificar segmentos.

Estructura y bloques fundamentales

Bloque de declaraciones. Donde pones los datos 1ue vas a usar como constantes y variables

Bloque de instruccuiones:Acciones sobre los elementos del bloque dedeclaración que permiten lograr el objetivo del programa.

-Entrada Obtener informacion
-Proceso Su finalidad es alcanzar el objeto del programación
-Salida Se encarga de almacenar el resultado

Tambien hay Bloque de uso de elementos externos Indica las calses o funciones externas
Bloque de definición del fichero/clase con comentarios etc.

Java Orientado a objetos,tipado fuerte y multiplataforma
JVM una máquina virtual que se instala en cada uno de los dispositivos en los que se quiere ejecutar los programas, de forma que no se ejecutan instrucciones binarias, sino que se ejecuta un lenguaje intermedio sobre la máquina virtual que traduce al lenguaje máquina del anfitrión.

El lenguaje intermedio de Java se llama Bytecode

CompiladorJIT mejora el rendimiento de aplicaciónes java

El **compilador** es otro de los elementos que realiza diferentes funciones para transformar el programa escrito en Java a Bytecode, denominado Javac

Java Runtime Environment (JRE) Conjunto de programas que permite la ejecucion de Java,principalmente la JVM y librerias

Java Development Kit (JDK) software necesario para en desarrollo de aplicaciones en Java(Incluye JavaC Compilador y Javadoc genera documentacion

Variables

Es el nombre que se le da un espacio de memoria en los lenguajes de tipado fuerte es necesario indicar el tipo de dato

Tipos de datos

Byte: 8 bits en complemento a2
Short:16 bits en complemento a2
Int:Enteros en complemento a2
Long:Enteros mas grandes mayores q 0
Float:Decimales
Double:Decimales mas grandes
Boolean:Verdadero/Falso
Char: Unicode

Operadores de asignación, aritméticos y unarios.

= + - * /

Operadores unarios.

+ positivo
-negativo
++ incrementa 1
-- decrementa 1
! negacion

igualdad

==
!=
>
>=
<
<=

Destacar que el operador = indica asignación y el operador == comparación

lógicos

&& and
|| or

a nivel de bits

~. Operador NOT a nivel de bit, cambia los 0 por los 1 y viceversa.

Constantes

```
static final double PI = 3.141592653589793;
```

Comentarios

// una linea /* Varias lineas*/

```
import java.util.*;
import java.lang.Math;
class Ejercicio1 {

    public static void main (String args []){

        int peso=0,altura=0;
        double resultado;
        Scanner input= new Scanner (System.in);
        System.out.println("Introduccir la altura en cm:");
        altura=input.nextInt();
        System.out.println("Introduccir el peso en Kg:");
        peso=input.nextInt();
        resultado= (double)peso/Math.pow( (float)altura/100,2);
        System.out.println("El resultado es "+resultado+" ");
    }
}
```