Tema 1:
Introducción a la Programación Estructurada

Objetivos

- Mostrar el contexto en el que se desarrolla la asignatura.
- Comprender algunos conceptos como el de algoritmo y programa.
- Entender las tareas de edición, compilación y ejecución de un programa dentro de un entorno de programación.

Conceptos Generales

- **Informática**: estudio del tratamiento automático de la información.

INFORMÁTICA= INFORMACIÓN AUTOMÁTICA

  - Información
  - Automática

Conceptos Generales

- **Automática.-** una vez iniciado el proceso, puede seguir hasta su finalización sin intervención externa.

- **Características** y ventajas de la automatización:
  - Evita esfuerzos iterativos
  - Procesar muchos datos o ...
  - Más rapidez
  - Menos errores

Conceptos Generales

- **Información**: concepto amplio
  - Comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se poseen sobre una materia determinada.
  - Conjunto de símbolos con significado.

Conceptos Generales

- Puntos fundamentales del **tratamiento de la <u>información:</u>**
  - E (Entada: datos iniciales)
  - Tratamiento/Transformación
  - S (Salida: resultados)

- Cuando se realiza de forma automática:

¿Qué?¿Quién?

¿Cómo?

Conceptos Generales.

- ¿Qué?¿Quién?.

COMPUTADOR → Máquina electrónica dotada de una **memoria** de gran capacidad y de métodos de tratamiento de la información, capaz de **resolver problemas** matemáticos y lógicos mediante la utilización **automática** de **programas** informáticos (RAE).

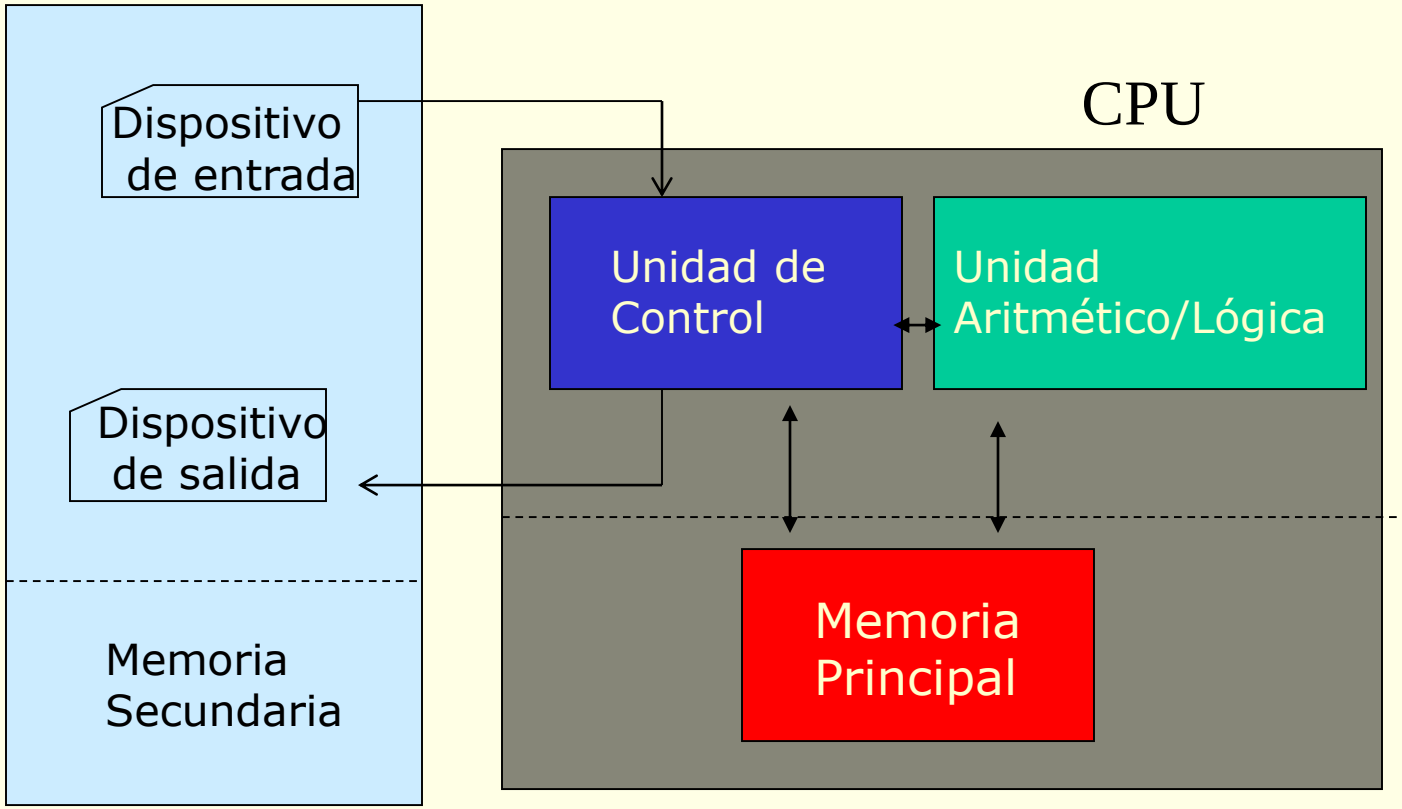
↓

HW≡Hardware ≡ sistema físico ≡ componentes físicos ≡ conjunto de dispositivos físicos

Conceptos Generales

ORDENADOR

- Componentes principales:



Conceptos Generales

- **Unidad Central de Proceso:**
  Circuitos capaces de ejecutar cálculos sencillos: +, -, * ...
  Ejecutan las instrucciones que están almacenadas en la memoria principal. La potencia de una computadora depende de la velocidad y fiabilidad de su CPU

- **Memoria:**
  Dispositivo físico, generalmente electrónico, en el que se almacenan de forma codificada los programas y datos, es decir, la información que procesa la CPU

- **Memorias auxiliares o secundarias:**
  Dispositivos físicos que permiten el almacenamiento masivo y permanente de información

- **Periféricos, unidades de entrada/salida:**
  Dispositivos a través de los cuales se alimenta la memoria y mediante los que se comunica los resultados , permiten al usuario comunicarse con la computadora: teclado, monitor, impresora...

Conceptos Generales

- Puntos fundamentales del **tratamiento de la <u>información:</u>**
  - E (Entada: datos iniciales)
  - Tratamiento/Transformación
  - S (Salida: resultados)

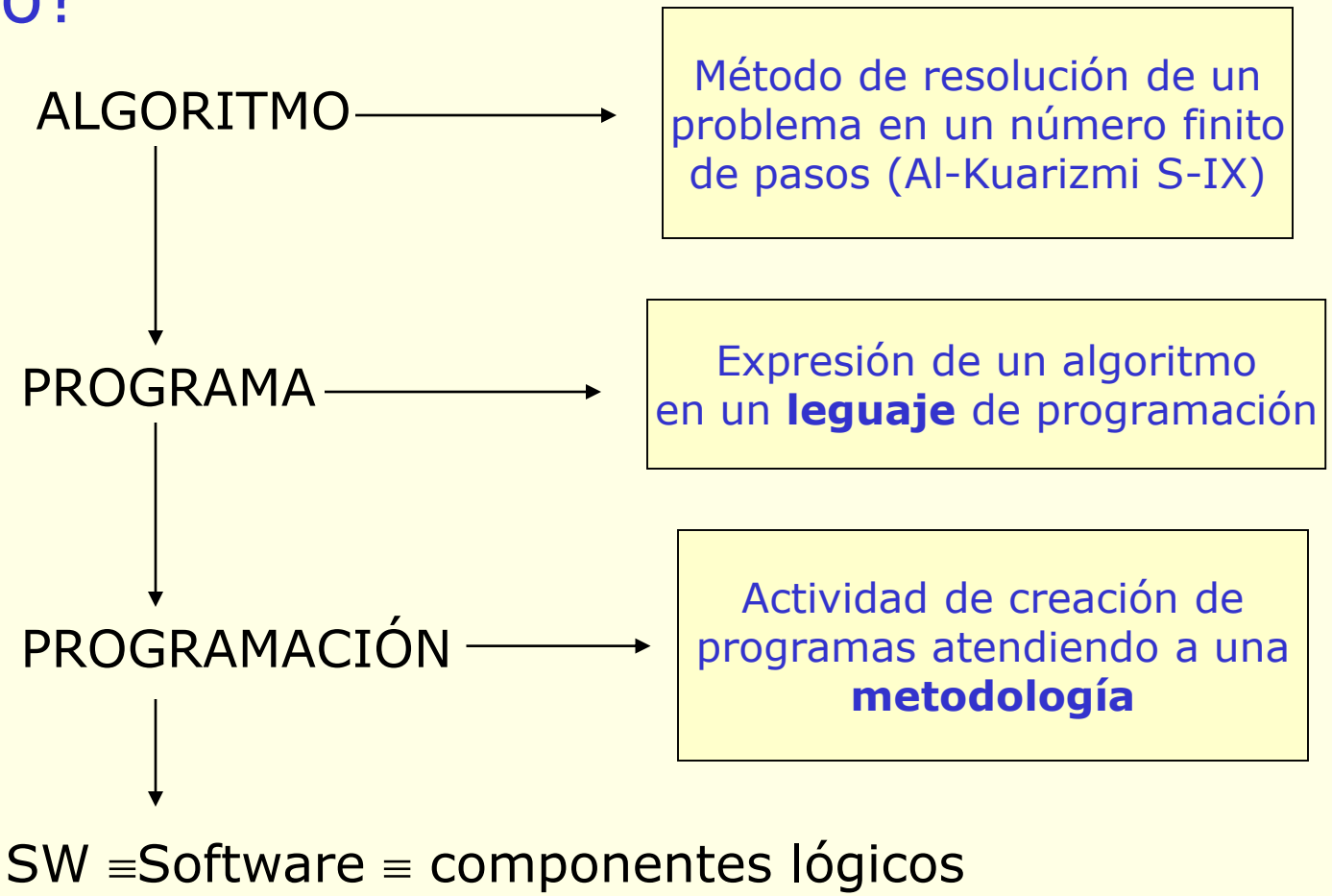
- Cuando se realiza de forma automática:

¿Qué?¿Quién?

¿Cómo?

Conceptos Generales

- ¿Cómo?



Conceptos Generales.

PROCESO	ALGORITMO	UN PASO	PROCESADOR
hacer tarta	receta	añadir azucar	cocinero
tocar una melodía	partitura musical	♫	orquesta
hacer vestido	patrón de vestido	coser un bolsillo	modisto
programa en ejecución	**método para calcular el mayor de 3 números**	**comparar primer y segundo número**	**ordenador**

Conceptos Generales

- Lenguajes de Programación:
  - Lenguaje máquina
  - Lenguaje ensamblador
  - Lenguajes de alto nivel

Conceptos Generales

- **Lenguajes de Programación**:
  - **Sintaxis** : conjunto de reglas gramaticales que establecen como pueden utilizarse correctamente los símbolos del lenguaje
  - **Semántica**: significado de las distintas construcciones del lenguaje

Conceptos Generales

Lenguaje máquina --- Alfabeto {0,1}

- Complicado: lento de redacción y programas largos
- Elevada posibilidad de cometer errores
- Orientado a la máquina
  (específica de cada tipo de máquina)
- Necesita personal especializado

Ejemplo: **0001000000000101**

cod. operación operando

BIT = Binary Digit.
Es la más pequeña unidad de representación de la información

Conceptos Generales

Lenguaje ensamblador

Alfabeto:{a,b,c ... O,1,2,...), * ,+, ...}

- Complicado: lento de redacción y prog. largos
- Menor posibilidad de cometer errores que l. máquina
- Programas más legibles y más cómodos de revisar
- Orientado a la máquina
  (específica de cada tipo de máquina)
- Necesita personal especializado
- Necesita un traductor (aumenta el trabajo del ordenador)

Ejemplo: **LOAD A**

cod. op. operando

Conceptos Generales

Lenguaje de alto nivel

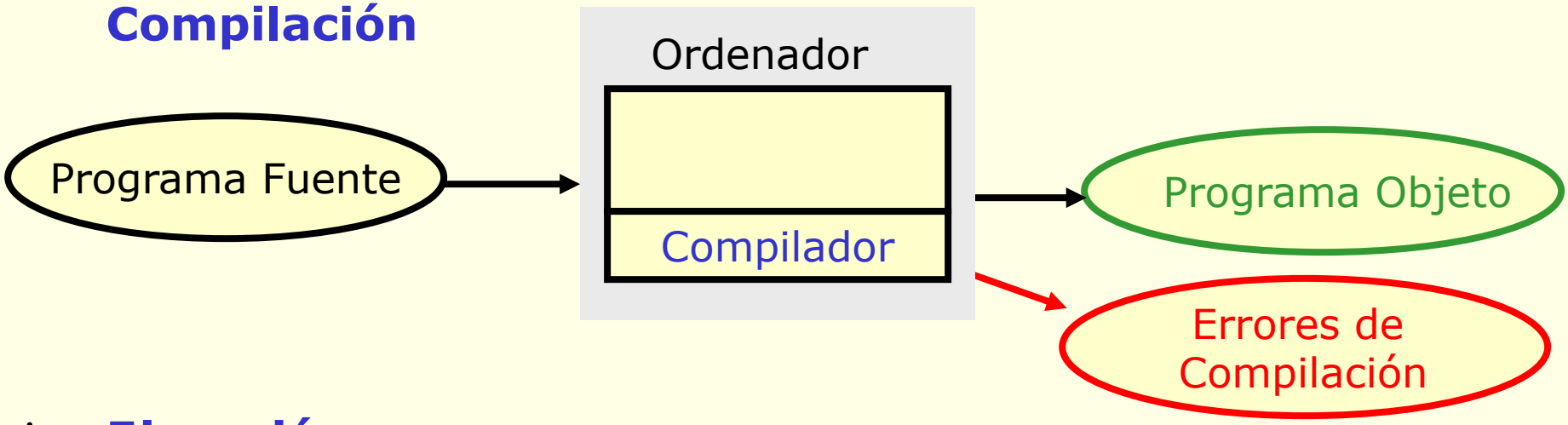
Alfabeto:{a,b,c ... 0,1,2,...), * ,+, ...}

- Universales --- independientes de la máquina
- Orientados a Problemas
- Programas más cortos
- Necesitan un traductor:
  - Compiladores
  - Interpretes

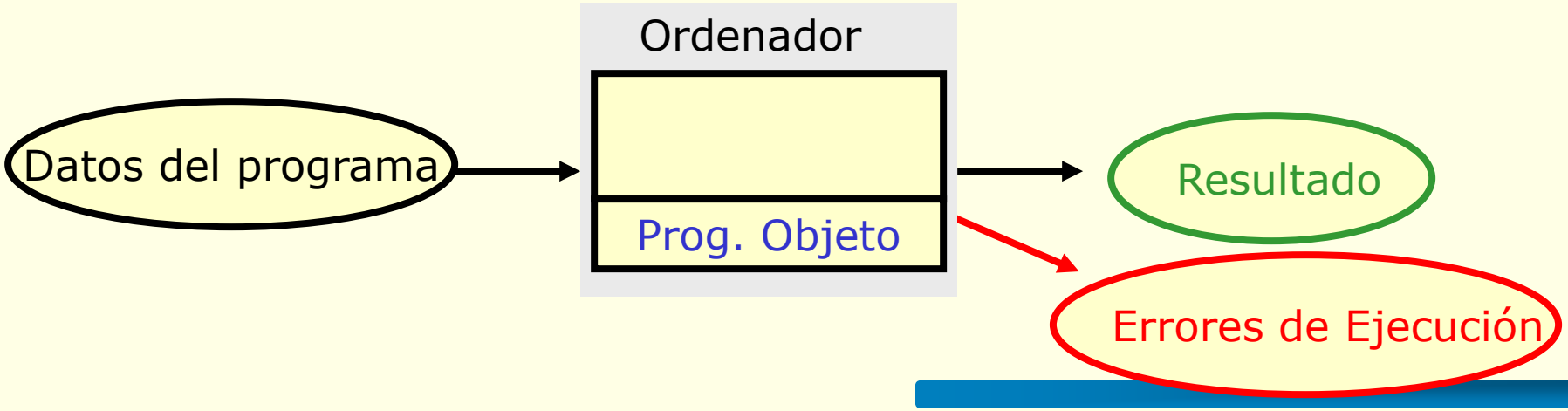
Conceptos Generales

Ejecución de un programa:

Compilación



Ejecución



Conceptos Generales

- **Programa fuente:**

  algoritmo escrito en un lenguaje de programación.

- **Programa Objeto:**

  es el programa fuente traducido, mediante el compilador, al lenguaje máquina.

- **Compilador:**

  programa que se encarga de traducir un programa escrito en un lenguaje de programación al lenguaje máquina.

- **Sistema operativo:**

  es un programa (windows, ms-dos, unix...) que actúa como interfaz entre el usuario y la máquina, componen el Software más básico del ordenador. Controlan y gestionan el funcionamiento del hardware.

Conceptos Generales. Resumen

- Resumen. Conceptos principales:
  - Memoria Principal ⇔ Memoria Secundaria
  - Ordenador – Hw
  - **Algoritmo – Programa** – Sw
  - Lenguajes de Programación
  - **Compilador**
  - **Programa Fuente – Programa Objeto**

Conceptos Generales. Definiciones

De la RAE (Real Academia Española):

Ordenador, ra.

m. *Esp.* **computadora electrónica.**

Computadora electrónica.

f. Máquina electrónica, analógica o digital, dotada de una memoria de gran capacidad y de métodos de tratamiento de la información, capaz de resolver problemas matemáticos y lógicos mediante la utilización automática de programas informáticos.

Conceptos Generales. Definiciones

De la RAE:

Bit.

(Del ingl. *bit*, acrón. de *bi*nary digi*t*, dígito binario).

m. *Inform.* Unidad de medida de información equivalente a la elección entre dos posibilidades igualmente probables.

Octeto.

m. *Inform.* Carácter o unidad de información compuesto de ocho bits.

Byte. (Voz ingl.).

m. *Inform.* **octeto** (‖ unidad de información).

Conceptos Generales. Definiciones

De la RAE:

Tratamiento de la información.

m. *Inform.* Aplicación sistemática de uno o varios programas sobre un conjunto de datos para utilizar la información que contienen.

Memoria.

f. *Fís.* Dispositivo físico, generalmente electrónico, en el que se almacenan datos e instrucciones para recuperarlos y utilizarlos posteriormente.

Compilador, ra.

m. *Inform.* Programa que convierte el lenguaje informático empleado por el usuario en lenguaje propio del computador.

Conceptos Generales. Definiciones

De la RAE:

Algoritmo.

m. Conjunto ordenado y finito de operaciones que permite hallar la solución de un problema.

Programa.

m. *Inform.* Conjunto unitario de instrucciones que permite a un ordenador realizar funciones diversas, como el tratamiento de textos, el diseño de gráficos, la resolución de problemas matemáticos, el manejo de bancos de datos, etc.

Programación.

f. Acción y efecto de programar.

Programar.

tr. *Inform.* Elaborar programas para su empleo en ordenadores.

Conceptos Generales. Definiciones

De la RAE:

Software. (Voz inglesa).

m. *Inform.* Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Hardware. (Voz inglesa).

m. *Inform.* **equipo** (‖ conjunto de aparatos de una computadora).

Sistema Operativo.

m. *Inform.* Programa o conjunto de programas que realizan funciones básicas y permiten el desarrollo de otros programas.

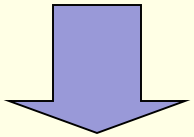
Conceptos Generales

¿Como obtenemos el

Programa Fuente?

Ciclo de Vida. Metodología de Programación

- **Fases para la obtención de resultados desde el planteamiento de un programa**:
  - Análisis del problema
  - Diseño del algoritmo
  - Codificación del algoritmo. Programa.
  - Pruebas y puesta a punto



Documentación del programa

Ciclo de Vida. Metodología de Programación

Dado un enunciado:

1º Entender el enunciado, entender las características de entrada, salida…, analizar todos los requisitos que aparecen en él

2º Buscar un algoritmo para solucionar el problema

En el curso se estudiarán distintos modelos de ALGORITMOS para resolver diversos tipos de problemas

Para expresar algoritmos se puede utilizar un pseudocódigo (repres. gráfica)

3º Codificar el algoritmo en un LENGUAJE DE PROGRAMACIÓN (entendible por una computadora), es decir escribir un programa

En el curso se estudiará el lenguaje de programación **C**

4º Editar el programa en un entorno de desarrollo
5º Compilar el programa en un entorno de desarrollo
6º Ejecutar el programa en un entorno de desarrollo

En el curso se utilizará el entorno de desarrollo **CodeBlock**

Ciclo de Vida. Metodología de Programación

Dado un enunciado:

ANÁLISIS:

1º Entender el enunciado, entender las características de entrada, salida…, analizar todos los requisitos que aparecen en él.

DISEÑO:

2º Buscar un algoritmo para solucionar el problema.

CODIFICACIÓN:

3º Codificar el algoritmo en un LENGUAJE DE PROGRAMACIÓN.

4º Editar el programa en un entorno de desarrollo.

PRUEBAS:

5º Compilar y probar el programa.

Análisis del problema

- Objetivo: "QUÉ"
- Especificación de requisitos SW:
  - Requisito: La condición o característica que tener o cumplir un sistema para satisfacer un contrato, norma,...
  - Especificación: establecimiento conciso de un requisito
    - Especificación de requisitos de entrada
    - Especificación de requisitos de salida
    - Especificaciones de requisitos de diseño

Ciclo de Vida. Metodología de Programación

- Problema: Hacer un programa que sume dos números

  - Especificaciones de entrada:
    - Los valores se leerán por teclado
    - Estos valores serán enteros

  - Especificaciones de salida
    - Se escribirá en pantalla la suma de los valores leídos

  - Especificaciones de diseño
    - No se pueden utilizar estructuras de datos

Ciclo de Vida. Metodología de Programación

Diseño del algoritmo

Objetivo: "CÓMO": Algoritmo

- Diseño descendente del algoritmo

  (usando diagramas de acción o pseudocódigo)
- Diseño modular
- Programación estructurada

Ciclo de Vida. Metodología de Programación

- Diseño descendente
  - Primera aproximación:

leer los 2 números
calcular la suma
escribir el resultado

Ciclo de Vida. Metodología de Programación

- Diseño descendente. Segunda aproximación:

Parte Operativa

leer (entero1, entero2)

evaluar entero1+entero2 y
guardar el resultado en suma

escribir (suma)

Parte Declarativa

entero1, entero2: variables enteras
para almacenar los tres números
suma: variable entera
para almacenar la suma
de los números leídos

Ciclo de Vida. Metodología de Programación

- Diseño descendente. Tercera aproximación:

Parte Operativa

```
leer (numero1, numero2)

suma ← numero1 + numero2

escribir (suma)
```

Parte Declarativa

entero1, entero2: variables enteras
para almacenar los tres números
suma: variable entera
para almacenar la suma
de los números leídos

Ciclo de Vida. Metodología de Programación

Diseño del algoritmo. Programación estructurada

Establece las siguientes reglas:

1. Todo programa se compone de una serie de acciones o sentencias que se ejecutan en **secuencia** (una a continuación de otra, desde la primera hasta la última sentencia).

2. Cualquier acción puede ser sustituida por dos o más acciones en secuencia.

3. Cualquier acción puede ser sustituida por cualquier estructura de control: **compuesta** (también llamada secuencial), **bifurcación** (también llamada selección o alternativa) y **repetición** (también llamada iterativa).

4. **Toda estructura de control tiene un solo punto de entrada y un solo punto de salida**.

Ciclo de Vida. Metodología de Programación

Codificación del algoritmo. Programa.

- Objetivo: “PROGRAMA”
- Documentación:

  Texto autodocumentado del programa:

  - Comentarios
  - Formateado de programas
  - Código autodocumentado

Ciclo de Vida. Metodología de Programación

```
/* Programa que suma dos numeros */

#include <stdio.h>

void main()
{
   int entero1;
   int entero2;
   int suma;

   printf( "Introduzca el primer entero\n" );
   scanf( "%d", &entero1 );                  /* lee el primer entero */
   printf( "Introduzca el segundo entero\n" );
   scanf( "%d", &entero2 );                  /* lee el segundo entero */

   suma = entero1 + entero2;                 /* asigna el total a suma */

   printf( "La suma es %d\n", suma );        /* imprime la suma */

}
```

Pruebas y puesta a punto

- Prepararemos distintos juegos de "datos de prueba" para verificar si con dichos datos el funcionamiento del programa es correcto.

- No existe ninguna técnica automática que nos garantice el correcto funcionamiento de un programa.