Grado en Ingeniería Información

PROGRAMACIÓN I

Sesión 1

Curso 2022-2023

Marta N. Gómez

UNIVERSIDAD NEBRIJA

- Introducción
  - Lenguajes de programación
  - Traductores
  - Algoritmos

UNIVERSIDAD NEBRIJA

ÍNDICE

Sistema de Procesamiento de la Información

es un sistema que partiendo de unos ***datos*** es capaz de generar otros ***datos*** que son ***significativos*** y ***útiles*** para el ***usuario***

Sistema de Procesamiento de la Información

es un sistema que partiendo de unos ***datos*** es capaz de generar otros ***datos*** que son ***significativos*** y ***útiles*** para el ***usuario:*** **Información**

Datos de entrada: son los ***datos necesarios*** para que el **sistema los procese** y llegue a la **información útil para el usuario**.

Datos de salida: es la ***información*** a la cual se llega **después de haber procesado los datos de entrada**.

Datos de entrada: son los ***datos necesarios*** para que el **sistema los procese** y llegue a la **información útil para el usuario**.

Datos de salida: es la ***información*** a la cual se llega **después de haber procesado los datos de entrada**.

HARDWARE: Conjunto de ***componentes físicos*** (tangibles) de un ***ordenador*** (equipo físico).

SOFTWARE: Es la ***parte lógica*** (no tangible) de la computadora. Son los ***programas*** y ***aplicaciones*** informáticas que se ejecutan sobre el ***hardware***, haciendo que este funcione y sacándole el máximo rendimiento.

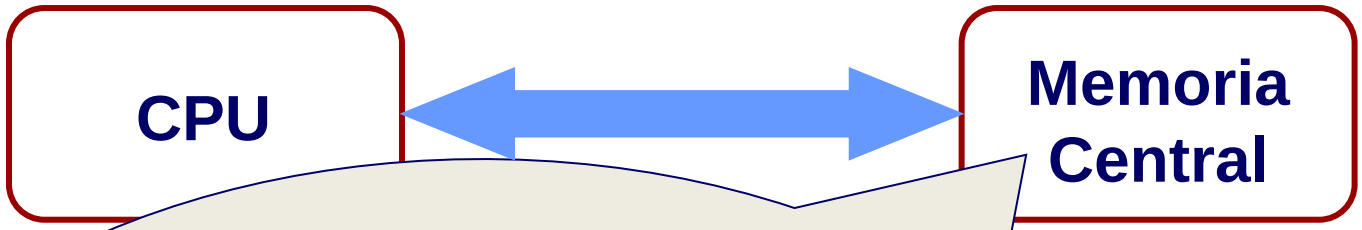
HARDWARE

CPU

Unidad Central de Proceso
(Central Processing Unit)

- Circuitos electrónicos que realizan operaciones básicas.
- Ejecuta las instrucciones de programas.
- Su velocidad y fiabilidad indica la potencia del ordenador.

HARDWARE



Memoria Central o Principal

- Almacena las instrucciones de los programas y los datos que procesa.
- Se divide en celdas a las que se accede individualmente (se direccionan).

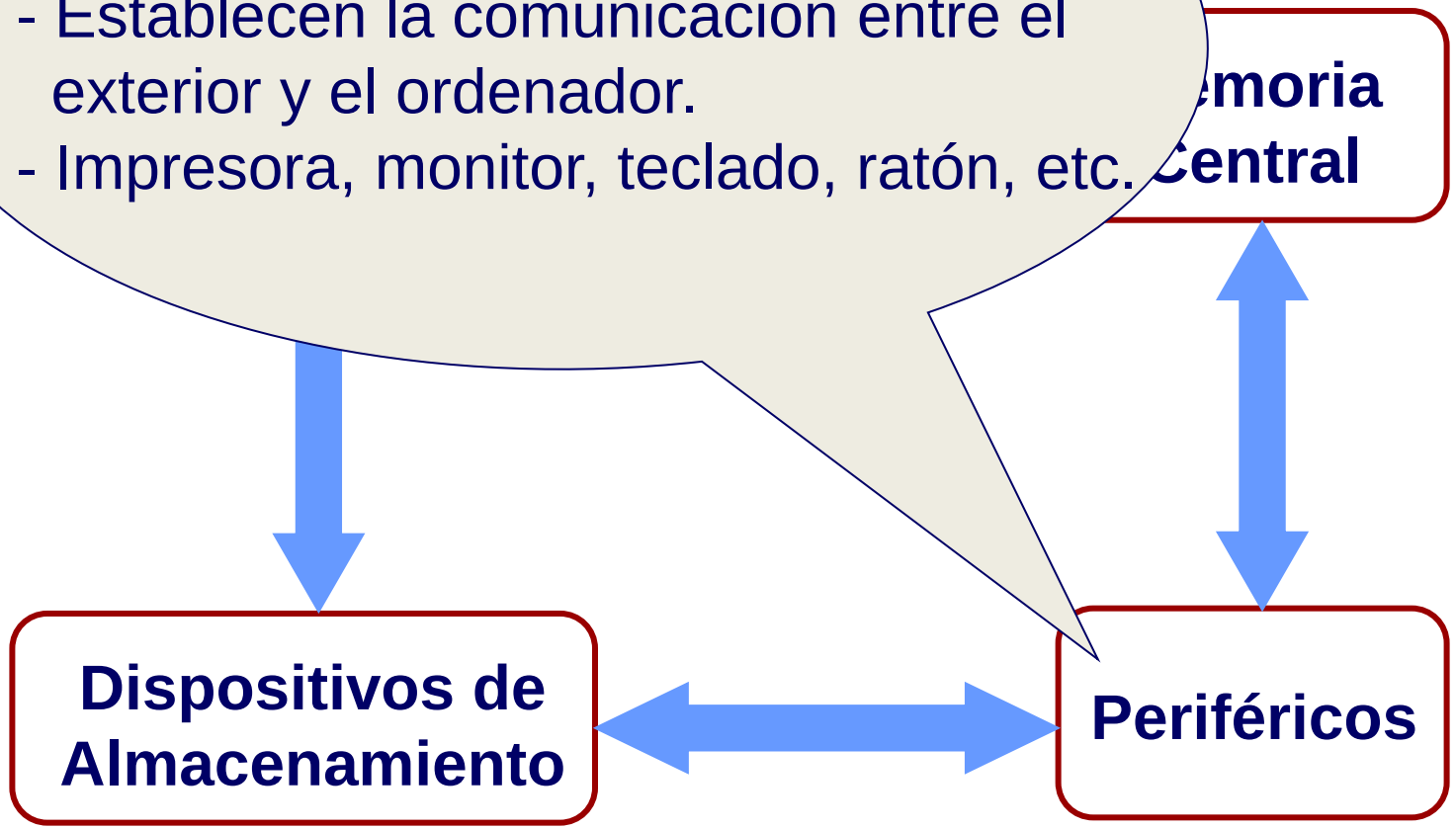
HARDWARE


CPU
Dispositivos de Almacenamiento
Memoria Secundaria o Dispositivos de Almacenamiento Secundario
- Tamaño mucho mayor que el de la memoria principal.
- Suele estar divida en archivos.

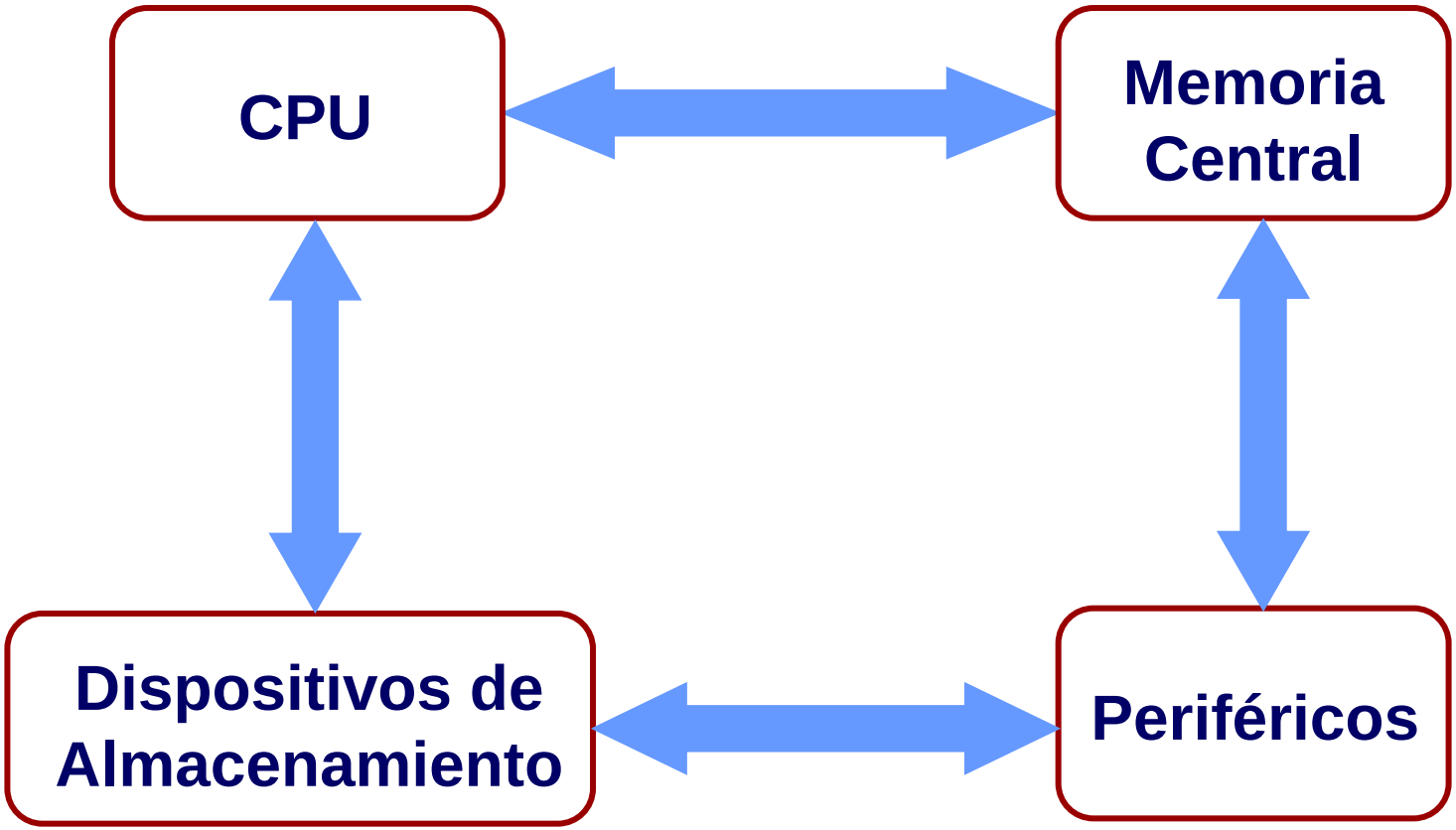
HAR

Periféricos o Dispositivos de E/S

- Establecen la comunicación entre el exterior y el ordenador.
- Impresora, monitor, teclado, ratón, etc.



HARDWARE



SOFTWARE

Existe una comunicación **bidireccional** entre el ***hardware*** y el ***software***.

Sistema operativo:

- Software que va por encima del hardware.
- Conjunto de programas que **administran** eficazmente los recursos hardware del computador.
- Proporciona una **interfaz** entre el resto de programas del ordenador, los dispositivos hardware y el usuario.

PROGRAMA

Un **programa** es un ***conjunto de instrucciones*** que el ***ordenador*** **interpreta** (**ejecuta**) para realizar una ***determinada tarea***.

Lenguaje de programación:

Un conjunto de ***palabras*** (órdenes), ***operadores*** y ***reglas sintácticas*** que permiten al programador crear los ***programas***, con el fin de indicarle al ordenador ***qué tarea*** debe realizar y ***de qué modo*** debe hacerla.

PROGRAMACIÓN

Proceso por el cual se ***escribe***, se ***compila***, se ***depura***, se ***prueba*** y se ***mantiene*** el código fuente de un programa informático.

TIPOS DE LENGUAJES

- ✓ Lenguajes de **Bajo Nivel**.
- ✓ Lenguajes de **Alto Nivel**.

Lenguajes de Bajo Nivel:

Se caracterizan por tener una ***dependencia de la arquitectura*** de la máquina (ordenador) en el que se va a ejecutar el programa creado.

Lenguaje Máquina:

Único lenguaje que el ordenador es capaz de comprender.

Las instrucciones para que un programa **pueda ser ejecutado** tienen que estar escritas en lenguaje máquina.

Es un lenguaje **muy complicado** para las personas ya que se expresa íntegramente en **sistema binario** (0 y 1).

Lenguaje Ensamblador:

La **necesidad de recordar secuencias de programación** (en sistema binario) para las acciones usuales llevó a sustituirlas por palabras fáciles de memorizar y asociar: ADD (sumar), SUB (restar), MUL (multiplicar), CALL (ejecutar subrutina), etc.

Estas palabras se denominan:

mnemónicos o ***nemónicos***.

Lenguaje Ensamblador:

El ***nemónico*** es una palabra que tiene sentido para el programador y que está **asociada a una instrucción del lenguaje máquina**.

Lenguajes de Alto Nivel:

- Surgen sobre los S.O. con la finalidad de ***independizar el lenguaje de programación de la arquitectura*** de la **máquina** sobre la que se **ejecutará** el programa.
- Son ***más simples*** (no requieren utilizar el sistema binario y su estructura sintáctica es similar a los lenguajes humanos) y ***facilitan el proceso de la programación***.
- Su objetivo es lograr **independencia de la máquina** en la que se va a ejecutar, pudiendo **utilizar un mismo programa en diferentes equipos**.

Lenguajes de Alto Nivel:

Necesita disponer de un ***programa traductor*** (***compilador***), que se encarga de generar el **programa ejecutable** en lenguaje binario apropiado para la máquina en cuestión.

Incluyen ***rutinas de uso frecuente*** (las de entrada/salida), funciones matemáticas, etc., que se pueden utilizar siempre que se quieran sin necesidad de programarlas cada vez.

Permiten crear ***programas genéricos***, mientras que los de bajo nivel están orientados a la solución de problemas en una arquitectura o una computadora concreta.

Características de los Lenguajes de Alto Nivel

1. Tener una **gran simplicidad** en cuanto a las **operaciones que debe realizar** y en cuanto a las **reglas sintácticas**.
2. Debe de **tener un proceso de traducción eficiente**, el programa ejecutable que obtengamos debe ser lo **más rápido y corto posible**; el proceso de traducción perfectamente adaptado al código máquina.
3. Debe **ser legible**, y con una **simbología fácilmente asimilable**, que permita entender de una forma rápida y sencilla lo que hace el programa.
4. Debe ser un medio **conveniente para resolver los problemas a los que está orientado**.

Ventajas de los Lenguajes de Alto Nivel:

- **Transportabilidad**. El programa fuente escrito en una computadora puede ejecutarse en otra computadora sin realizar cambios.
- Utiliza **expresiones muy familiares** como READ, WRITE, IF THEN ELSE, etc.
- Facilidad para **mantenimiento y depuración** al contrario que los lenguajes de **bajo nivel**.
- El tiempo necesario de **formación** menor para su utilización (sintaxis sencilla, el programador no necesita conocer la arquitectura física de la computadora o procesador sobre la que está trabajando).

Inconvenientes de los Lenguajes de Alto Nivel:

- **No se conocer la arquitectura física** de la máquina y **tampoco se aprovechan las posibles ventajas de éstas**.
- El **tiempo de ejecución**, en general, es **mayor** (el programa fuente tiene que traducirse a lenguaje máquina, y normalmente se genera más código que si se programase en lenguaje **máquina** o lenguaje **ensamblador**).
- **Mayor uso de memoria** durante el **proceso de compilación o traducción** del lenguaje de **alto nivel**.

Existen dos tipos de programas que **traducen** de lenguaje de **alto nivel** a lenguaje **máquina**:

los **Intérpretes** y los **Compiladores**

Intérpretes:

Traductores que **realizan su tarea por bloques**:

Según se ***analiza*** un **bloque** del **programa fuente**, se ***genera*** el **código máquina** correspondiente, y se ***ejecuta***.

El **ciclo** se repite hasta que acaba el programa.

El **análisis**, la **traducción**, y la **ejecución** están **fuertemente ligados**.

Los bloques de traducción corresponden a una **única instrucción**.

A cada instrucción de alto nivel le suelen corresponder **varias de lenguaje máquina**, en función de la complejidad de la instrucción.

Compiladores:

Traductores que **realizan su tarea globalmente**.

Se **analiza** todo el programa fuente, se **genera** el código máquina correspondiente, y se **almacena**, **todo de una vez**.

Una vez realizada la traducción, el **programa objeto**, que se ha almacenado, **se puede ejecutar tantas veces como se quiera** sin tener que volver a traducir.

El **análisis** y la **traducción**, son **secuenciales**.

La **ejecución** es **independiente**.

El ser humano se enfrenta continuamente a **problemas**.

Cuando el **nivel de abstracción** es suficientemente alto, la mente humana realiza el mismo trabajo:

RESOLVER UN PROBLEMA

El proceso seguido es:

1. **Identificar** el problema.
2. **Definir** y **representar** el problema.
3. **Explorar** diferentes **estrategias**.
4. **Aplicar** y **mejorar** las estrategias.
5. **Mirar atrás** y **evaluar** lo realizado **(¿se ha resuelto el problema?).**

RESOLVER UN PROBLEMA

1. **Identificar** el problema → Decidir **QUÉ** hacer.
2. **Definir** y **representar** el problema → Decidir **CÓMO** hacerlo.
3. **Explorar** las diferentes **estrategias** → **HACERLO.**
4. **Aplicar** y **mejorar** las estrategias → **PROBAR** el resultado.
5. **Mirar atrás y evaluar** lo realizado → **USAR** lo realizado.
   (*¿se ha resuelto el problema?).*

Construir un puente

1. **QUÉ:** determinación de la localización exacta del puente, número de carriles del puente, peso que debe soportar, etc.
2. **CÓMO:** diseño del puente y detalle de cómo debería construirse.
3. **REALIZACIÓN:** construcción del puente.
4. **PRUEBA:** comprobación de que el puente cumple las características requeridas. Nótese que, en este caso, la mayor parte de las pruebas se llevan a cabo durante la fase de diseño, a través de modelos. Lo que se comprueban son los conceptos y las decisiones de diseño.
5. **USO:** cuando se comienza la construcción física de un puente, se asume que el diseño es correcto. Una vez llevada a cabo la revisión del diseño, bien se refina el diseño para subsanar deficiencias o bien es aceptado, pasándose a este paso de uso real del puente.

FASE DE RESOLUCIÓN DEL PROBLEMA

Análisis.-
Comprender y definir el problema.

Solución General (ALGORITMO).-
Desarrollar una secuencia lógica de pasos que será utilizada para resolver el problema.

Prueba.-
Seguimiento exacto de los pasos establecidos para ver si la solución resuelve realmente el problema.

FASE DE IMPLEMENTACIÓN

Solución Específica (PROGRAMA).-
Traducir el algoritmo a un lenguaje de programación (código).

Prueba.-
Hacer que la computadora siga las instrucciones. Comprobar los resultados y hacer las correcciones oportunas hasta que las respuestas sean correctas.

Uso.-
Utilizar el programa.

ALGORITMO:

Conjunto **ordenado de pasos** que especifican la **secuencia de operaciones** que se han de realizar para **resolver un problema**. Y todas las **acciones concretas** a realizar con **independencia de los datos** a procesar.

Características:

1.- Punto de **INICIO**.

2.- Ser **preciso**.

3.- Bien **definido**.

4.- **Fácil** de **leer** e **implementar**.

5.- **Finito**.

6.- **Independiente** del lenguaje de programación.