INTRODUCCIÓN

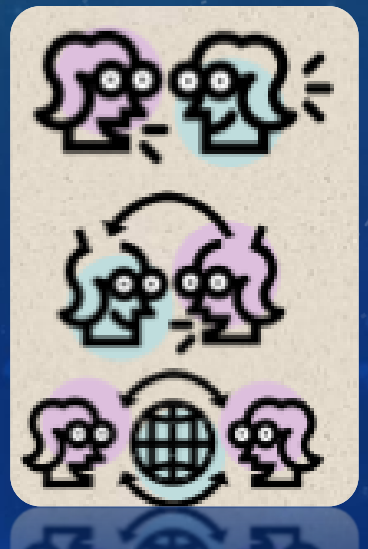
ING. IVONNE MALDONADO

PROGRAMACIÓN

- **Programar no es mas que** idear y ordenar las **acciones.**
- Lograr que un programa cumpla con una cierta tarea en un momento determinado.
- Elaboración de **programas** para la resolución de problemas mediante computadoras
- La programación es el proceso de escribir, en un lenguaje de programación, el código fuente que permite resolver un problema en particular.

PROGRAMAS

- Hace referencia a un conjunto de instrucciones individuales que son creadas por el programador, código fuente.
  - **Sintaxis:** son las reglas con las que deben unirse los elementos de un lenguaje de programación (términos y puntuación) para construir frases (líneas de código).
  - **Gramática:** es el conjunto de reglas que me permiten implementar instrucciones en un determinado lenguaje.
  - **Semántica:** trata el significado y propósito del código



LENGUAJES DE PROGRAMACIÓN

- **Lenguaje inventado para controlar una máquina.**
- **Fueron inventados para facilitar el abordaje de distintos problemas, el mantenimiento del software, su reutilización, mejorar la productividad, etc.**

CLASIFICACIÓN

Nivel de Abstracción

Lenguajes de bajo nivel

- La programación se realiza teniendo muy en cuenta las características del procesador.
- Ejemplo: Lenguajes ensamblador.

Lenguajes de nivel medio

- Permiten un mayor grado de abstracción pero al mismo tiempo mantienen algunas cualidades de los lenguajes de bajo nivel.
- Ejemplo: C puede realizar operaciones lógicas y de desplazamiento con bits, tratar todos los tipos de datos como lo que son en realidad a bajo nivel (números), etc.

Lenguajes de alto nivel

- Más parecidos al lenguaje humano. Manejan conceptos, tipos de datos, etc., de una manera cercana al pensamiento humano ignorando el funcionamiento de la máquina.
- Ejemplos: Java, Ruby.

CLASIFICACIÓN

Propósito

General

- Aptos para todo tipo de tareas.
- Ejemplo: C.

Específico

- Hechos para un objetivo muy concreto.
- Ejemplo: Csound (para crear ficheros de audio)

Programación de sistemas

- Diseñados para realizar sistemas operativos o drivers.
- Ejemplo: C.

Lenguajes de script

- Para realizar tareas varias de control y auxiliares. Antiguamente eran los llamados lenguajes de procesamiento por lotes (batch) o JCL (“Job Control Languages”). Se subdividen en varias clases (de shell, de GUI, de programación web, etc.).
- Ejemplos: bash (shell), mIRC script, JavaScript (programación web).

CLASIFICACIÓN

Manera de Ejecutarse

Lenguajes compilados

- Un programa traductor traduce el código del programa (código fuente) en código máquina (código objeto). Otro programa, el enlazador, unirá los ficheros de código objeto del programa principal con los de las librerías para producir el programa ejecutable.
- Ejemplo: C.

Lenguajes interpretados

- Un programa (intérprete), ejecuta las instrucciones del programa de manera directa.
- Ejemplo: Lisp.

Mixtos

- Como Java, que primero pasan por una fase de compilación en la que el código fuente se transforma en “bytecode”, y este “bytecode” puede ser ejecutado luego (interpretado) en ordenadores con distintas arquitecturas (procesadores) que tengan todos instalados la misma “máquina virtual” Java.

CLASIFICACIÓN

Paradigma de Programación

Procedural

- Divide el problema en partes más pequeñas, que serán realizadas por subprogramas (subrutinas, funciones, procedimientos), que se llaman unas a otras para ser ejecutadas.
- Ejemplos: C, Pascal.

Orientada a objetos

- Crean un sistema de clases y objetos siguiendo el ejemplo del mundo real, en el que unos objetos realizan acciones y se comunican con otros objetos.
- Ejemplos: C++, Java.

Funcional

- La tarea se realiza evaluando funciones, (como en Matemáticas), de manera recursiva.
- Ejemplo: Lisp.

Lógica

- La tarea a realizar se expresa empleando lógica formal matemática. Expresa qué computar.
- Ejemplo: Prolog.

CLASIFICACIÓN

Realización Visual

Visual

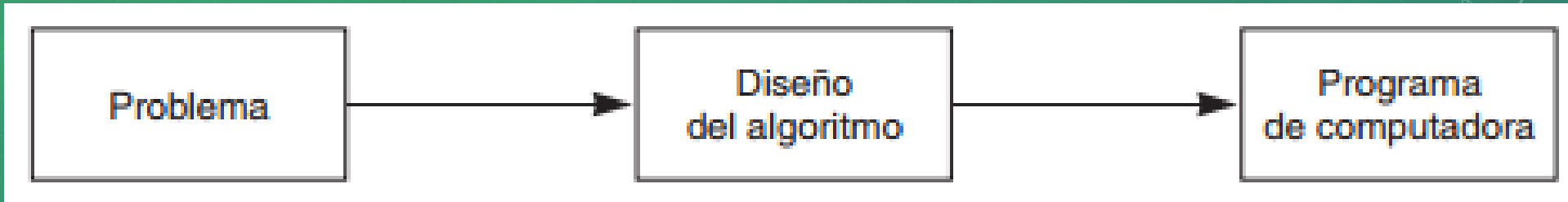
- El programa se realiza moviendo bloques de construcción de programas (objetos visuales) en un interfaz adecuado para ello. No confundir con entornos de programación visual, como Microsoft Visual Studio y sus lenguajes de programación textuales (como Visual C#).
- Ejemplo: Mindscript.

Textual

- El código del programa se realiza escribiéndolo.
- Ejemplos: C, Java, Lisp.

ALGORITMOS

- Un algoritmo es un método para resolver un problema.
- Conjunto finito y ordenado de acciones con las que podemos resolver un determinado problema.
- Llamamos “problema” a una situación que se nos presenta y que, mediante la aplicación de un algoritmo, pretendemos resolver.
  - Esperar a que la luz del semáforo peatonal esté en verde (esperarSemaforo);
  - Cruzar la calle (cruzarCalle)
- Algoritmos + Estructuras de datos = Programas, lo que significa que sólo se puede llegar a realizar un buen programa con el diseño de un algoritmo y una correcta estructura de datos.
- La resolución de un problema exige el diseño de un algoritmo que resuelva el problema propuesto.



- Los pasos para la resolución de un problema son:
- 1. Diseño del algoritmo, que describe la secuencia ordenada de pasos —sin ambigüedades— que conducen a la solución de un problema dado. (Análisis del problema y desarrollo del algoritmo.)
- 2. Expresar el algoritmo como un programa en un lenguaje de programación adecuado. (Fase de codificación.)
- 3. Ejecución y validación del programa por la computadora.

- Para llegar a la realización de un programa es necesario el diseño previo de un algoritmo, de modo que sin algoritmo no puede existir un programa.
- Los algoritmos son independientes tanto del lenguaje de programación en que se expresan como de la computadora que los ejecuta.
- En cada problema el algoritmo se puede expresar en un lenguaje diferente de programación y ejecutarse en una computadora distinta; sin embargo, el algoritmo será siempre el mismo.

CARACTERÍSTICAS DE LOS ALGORITMOS

- Debe ser preciso e indicar el orden de realización de cada paso.
- Debe estar definido. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.
- Debe ser finito. Si se sigue un algoritmo, se debe terminar en algún momento; o sea, debe tener un número finito de pasos.

La definición de un algoritmo debe describir tres partes: Entrada, Proceso y Salida.

REPRESENTACIÓN DE ALGORITMOS

- Cuando se quiere que un computador ejecute un algoritmo es indispensable, por lo menos hasta hoy, representar ese algoritmo mediante algún formalismo. Las técnicas utilizadas más comúnmente para la representación de algoritmos son:
  - Pseudocódigo
  - Diagrama de flujo

EJERCICIOS

- 1. Pasos a seguir para una actividad diaria.(escriban los pasos detalladamente)
- 2. Rutina para venir a clase (desde que se levanta hasta que está sentado en la clase)