\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|Software y programas Tema 1|

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Tipos de Software

\* Hardware: Componentes físicos que se pùeden ver y tocar:

\* Software: Conjuntosde programas informaticos que actuan sobre el hardware

para ejecutar lo que el usuario desee

-------------------------

\* Clasificacion Software:

-------------------------

## Por licencia

\* Software Libre: Se puede usar sin ninguna licencia.

\* Software Propietario: Se necesita una licencia.

## Tipos de tarea que realiza

\* Sistema: Software basico necesario para que funcione un equipo

(linux, unix,windows,macos,android).

\* Programacion:conjunto de apps que permiten elaborar otras apps.

(IDE's).

\* Aplicacion:Aquellos prgramas con un proposito concreto

(calculadora,vmware).

-------------------

| Que es programa?|

-------------------

Es un conjunto de instrucciones que permiten realizar una tarea

--------------------------

//Entornos de desarrollo.//

-------------------------

Es un tipo de software que facilitan y mejoran la programacion.

-------------

//Algoritmo.//

-------------

Es una secuencia ordenada de pasos para resolver un problema.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

####################################

#EJERCICIO1. Cajero de supermercado#

####################################

1 Escoger un producto

2 Tiene codigo?

2/1 no --> buscar precio en las tablas.

2/1 si --> escanear codigo

3 se suma precio

4 Repetir hasta que no haya mas productos pasos 2-3

5 cuando no haya mas productos informar del tota al cliente

6 cobrar total

7 Esperar al siguiente cliente y volver al paso 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Representacion Graficas de algoritmos

\* Diagrama de flujo: Es un esquema con figuras de un algoritmo

\*Pseudocodigo: Es una secuencia de instrucciones que representa las

proposiociones/sentencias de un programa.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|Lenguaje de programacion|

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Lenguaje binario: Sistema de 0 y 1 que la máquina entiende directamente.
* Lenguaje máquina: Instrucciones en código binario ejecutadas por la CPU.
* Lenguaje ensamblador: Lenguaje de bajo nivel que usa símbolos en lugar de binario.
* Lenguaje de alto nivel: Lenguaje más comprensible para humanos (Python, Java, C++).
* Lenguaje estructurado: Organiza el código en bloques lógicos como funciones o procedimientos.
* Código fuente: Conjunto de instrucciones escritas por el programador en un lenguaje de alto nivel.
* Código objeto: Versión traducida del código fuente a lenguaje máquina (tras la compilación).
* Compilación: Proceso que traduce el código fuente completo a código máquina antes de ejecutarlo.
* Interpretación: Ejecución del código línea por línea sin necesidad de compilarlo previamente.
* Entorno de desarrollo integrado (IDE): Aplicación que combina editor, compilador y depurador para programar fácilmente.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Maquina virtuales

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Máquina virtual (MV): Software que simula una computadora para ejecutar programas de forma independiente del hardware real.
* MV de sistema: Emula un sistema operativo completo (ej. VirtualBox, VMware).
* MV de proceso: Ejecuta un solo programa en un entorno aislado (ej. Java Virtual Machine).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Fases del Desarrollo

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Optimización de una aplicación: Proceso de mejorar el rendimiento, eficiencia y consumo de recursos de un programa.
* Seguridad de una aplicación: Conjunto de medidas para proteger datos y evitar accesos o acciones no autorizadas.
* ERS (Especificación de Requisitos del Software): Documento que define qué debe hacer el software y sus condiciones.
* Software maduro: Programa probado y estable, con pocas fallas y alto nivel de confiabilidad.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Fases de mantenimiento

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Prueba de caja negra: Evalúa el funcionamiento del software sin conocer su código interno, solo sus entradas y salidas.
* Prueba de caja blanca: Analiza la lógica interna del código para verificar su correcto funcionamiento.
* Mantenimiento correctivo: Corrige errores detectados tras la entrega del software.
* Mantenimiento adaptativo: Ajusta el software a cambios en el entorno o nuevas necesidades.
* Mantenimiento perfectivo: Mejora funciones o rendimiento sin que existan fallos previos.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Roles en desarrollo de software

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-Director de proyecto: Planifica, coordina y supervisa todo el proceso de desarrollo del software.

-Analista de sistemas: Estudia las necesidades del cliente y define los requisitos del sistema.

-Diseñador de software: Crea la arquitectura y estructura del programa antes de codificarlo.

-Desarrollador de software: Escribe, prueba y mantiene el código del programa.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Conceptos básicos de programación

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

- Variables: Espacios en memoria que almacenan datos que pueden cambiar durante la ejecución.

-Constantes: Valores que no cambian durante la ejecución del programa.

-Operaciones: Acciones realizadas sobre los datos (ej. suma, resta, comparación).

-Instrucciones: Órdenes que indican a la computadora qué hacer paso a paso.

-Comentarios: Líneas del código que no se ejecutan; sirven para explicar o documentar el programa.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Modelos de ciclo de vida del software

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-Modelo en cascada: El desarrollo se realiza por etapas consecutivas (análisis → diseño → implementación → pruebas → mantenimiento). Cada fase debe completarse antes de pasar a la siguiente.

-Modelo incremental: El software se construye y entrega en partes (incrementos), cada uno añade nuevas funciones hasta completar el sistema.

-Modelo de construcción de prototipos: Se crea un modelo inicial (prototipo) para comprender mejor los requisitos antes del desarrollo final.

-Modelo en espiral: Combina etapas del modelo en cascada con ciclos repetitivos de planificación, análisis de riesgos y mejora continua.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Marco de trabajo ágil

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-Marco de trabajo ágil: Conjunto de métodos que promueven entregas rápidas, trabajo en equipo y adaptación a cambios durante el desarrollo del software.

\*\*\*\*\*\*

Scrum

\*\*\*\*\*\*

-Scrum: Método ágil basado en ciclos cortos llamados sprints para entregar partes funcionales del software con colaboración constante.

-Product Owner (Propietario del producto): Define los requisitos y prioriza las tareas del proyecto; representa las necesidades del cliente.

-Scrum Master: Facilita el proceso Scrum, elimina obstáculos y asegura que el equipo siga las prácticas ágiles correctamente.

-Equipo de desarrollo Scrum: Grupo multidisciplinario que diseña, construye, prueba y entrega el producto en cada sprint.

-Product Backlog: Lista priorizada de todas las funciones, mejoras y correcciones pendientes del producto.

-Sprint Backlog: Conjunto de tareas seleccionadas del product backlog que se realizarán durante un sprint.

-Incremento Scrum: Resultado funcional del trabajo completado al final de cada sprint; debe estar listo para ser entregado o usado.

-Reunión diaria Scrum (Daily Scrum): Reunión corta (15 min aprox.) donde el equipo revisa avances, planes y obstáculos.

-Sprint: Periodo corto y fijo (generalmente de 1 a 4 semanas) donde el equipo desarrolla un conjunto de funcionalidades.

-Sprint Planning: Reunión al inicio del sprint donde se decide qué tareas se harán y cómo.

-Sprint Review: Revisión al final del sprint donde se muestra el incremento al cliente y se obtienen comentarios.

-Sprint Retrospective: Reunión de mejora interna del equipo para analizar qué funcionó bien y qué se puede optimizar.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Otros marcos ágiles

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

- Kanban: Método visual de gestión del trabajo basado en tarjetas y columnas que muestran el flujo de tareas (pendiente, en progreso, terminado).

- Tablero Kanban: Herramienta visual donde se organizan las tareas según su estado, ayudando a controlar el progreso y limitar el trabajo en curso.

- Scrumban: Enfoque híbrido que combina la estructura de Scrum con la flexibilidad visual de Kanban para mejorar la eficiencia y adaptabilidad.