**Unidad de Trabajo 1:**

**Desarrollo de Software**

# Software y programa. Tipos de software

## Preguntas

1. **¿Cómo se denomina al conjunto de programas que tienen una finalidad más o menos concreta?**

Software.

1. **¿Qué tipo de software es un videojuego?**

Es un tipo de aplicación informática.

1. **¿Cómo se denomina al conjunto de herramientas que nos permiten desarrollar programas informáticos?**

Entorno de desarrollo.

1. **¿Qué tipo de software es un depurador?**

Es un tipo de software de programación.

1. **¿Qué tipo de software es un compilador?**

Es un tipo de software de programación.

1. **¿De qué dos partes consta un ordenador?**

Un ordenador consta de la parte física denominada hardware, y los programas que lo componen que no son físicos llamado software.

1. **¿Cómo se denomina al software base que ha de estar instalado y configurado en nuestro ordenador para que las aplicaciones puedan ejecutarse y funcionar?**

Se denomina Sistema Operativo.

1. **¿Qué tipo de software es un controlador de dispositivo?**

Es un tipo de software de sistemas.

1. **¿Qué tipo de software es una aplicación ofimática?**

Es un tipo de software de aplicaciones.

1. **¿Qué tipo de software es un sistema operativo?**

Es un tipo de software de sistemas.

# Relación Hardware-Software

## Preguntas

1. **¿Cuál es la relación hardware-software desde el punto de vista del sistema operativo?**

El sistema operativo se encarga de la coordinación del hardware mientras se ejecutan los distintos programas y aplicaciones en el ordenador, ya que estas necesitan recursos del hardware constantemente y el sistema operativo se encarga de regular esta interacción de manera oculta.

1. **Define hardware**

El hardware es el conjunto de dispositivos que conforman un ordenador.

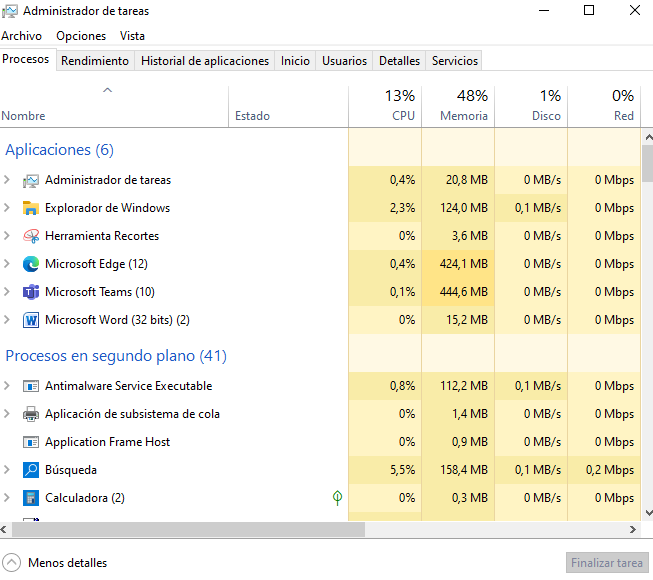
1. **Investiga en google cual fue la gran aportación de la Arquitectura Von Neumann. (Pista busca el concepto de programa almacenado)**

Introduce una parte de la memoria en la que guarda datos e instrucciones de un programa. Se basa en la implementación de una máquina de Turing y la visión de una arquitectura secuencial.

1. **¿Cuál es la relación hardware-software desde el punto de vista de las aplicaciones?**

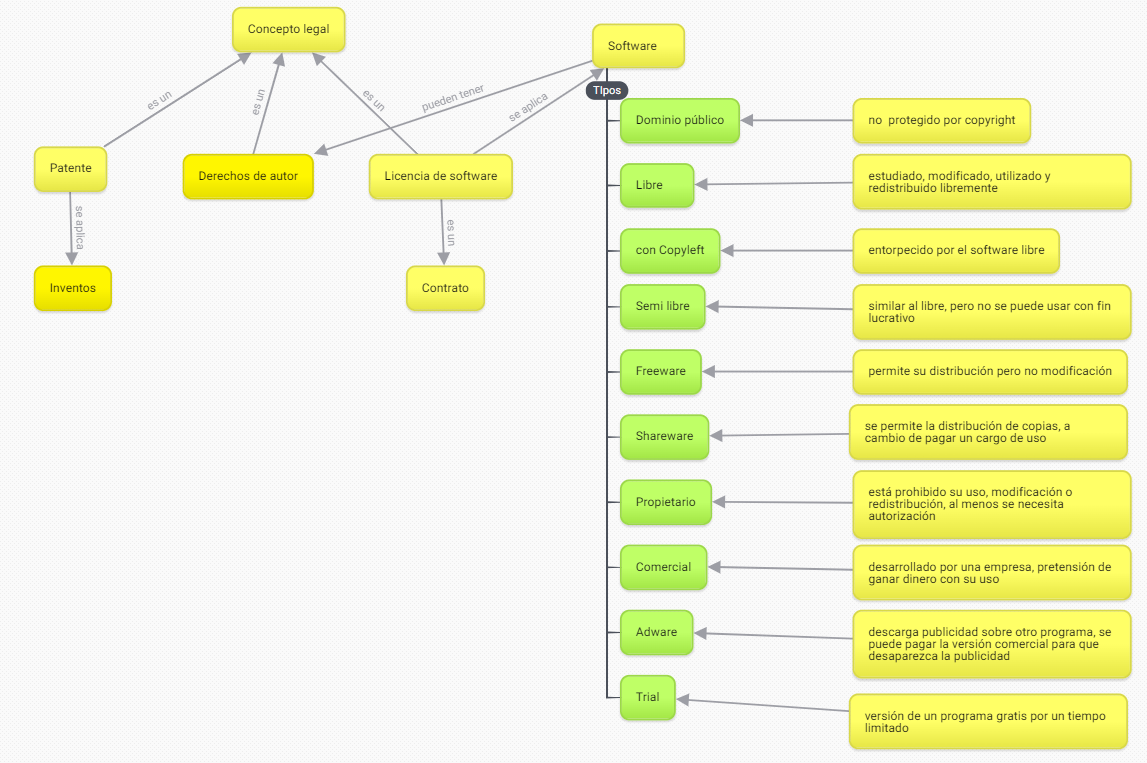
Las aplicaciones las programamos nosotros a través de distintos entornos de desarrollo, y utilizando diversos lenguajes de programación, sin embargo el hardware debe ejecutar y entender este lenguaje. Para ello, traduce el lenguaje que utilizamos a uno que él sí que puede comprender y ejecutar.

1. **Investiga cómo podemos con el “Administrador de tareas” de Windows ver como el sistema operativo asigna recursos a las aplicaciones. (Haz alguna captura donde se vea la asignación).**



# Licencias de Software. Software Libre y Propietario

## Mapa conceptual



# Ciclo de vida del software

## Preguntas

1. **¿Qué produce cada etapa y que recibe de la anterior?**

El análisis produce un modelo con requisitos.

Esto lo recibe el diseño y produce las estructuras y la arquitectura del programa.

Esto lo recibe la codificación y produce el código ejecutable que recibe la máquina.

Esto lo reciben las pruebas y garantizan el correcto funcionamiento del programa.

1. **¿Qué entendemos por ciclo de vida del software?**

Es el conjunto de fases por las que pasa el software que se está desarrollando.

1. **¿Qué entendemos por desarrollo del software?**

Es todo lo que ocurre desde que se tiene una idea hasta que esta se implementa y el software ya está funcionando e implementado.

# Modelos de ciclo de vida

## Preguntas

1. **Investiga en internet cuál de los dos modelos evolutivos realiza un análisis de riesgos en cada evolución.**

El modelo en espiral, ya que una fase del modelo es el análisis de riesgos, y se efectúa en cada fase.

1. **¿Cuál es la fase del ciclo de vida que conlleva más esfuerzo del total del proyecto?**

Es la fase de mejoras.

1. **¿Qué modelo va elaborando sucesivas versiones hasta llegar al producto final?**

El modelo evolutivo.

1. **¿Cuál es la fase del ciclo de vida que conlleva menos esfuerzo del total del proyecto?, ¿por qué crees que es así?**

La de implementación, porque una vez hecho el análisis y el diseño la implementación es más mecánico.

1. **¿En qué modelo se pasa de una etapa a otra sin que sea posible volver hacia atrás?**

En el modelo en cascada.

1. **¿Qué modelo es aconsejable usar para un proyecto pequeño en el que no habrá variaciones en los requisitos durante el desarrollo del mismo?**

En el modelo en cascada, ya que es el más corto, y no se puede volver atrás en las fases.

1. **¿Qué modelo es aconsejable usar para un proyecto que tendrá pocos cambios en los requisitos?**

El modelo en cascada con retroalimentación, ya que permite volver atrás y realizar pequeños cambios.

1. **¿Qué modelo está basado en el modelo en cascada con retroalimentación?**

El modelo iterativo incremental.

1. **¿Qué modelos realizan varias versiones del software cada vez más refinadas?**

El modelo en espiral.

1. **¿Qué modelo de ciclo de vida requiere conocer todos los requisitos de antemano?**

El modelo en cascada.

1. **¿Qué diferencia hay entre el modelo en cascada y el de cascada con retroalimentación?**

La diferencia es que en el de cascada con retroalimentación se puede volver atrás en las fases para realizar pequeños cambios, sin embargo en el de cascada no se puede volver atrás.

# Fases en el desarrollo y ejecución del Software

## Análisis

### Preguntas

1. **¿Qué dos tipos de requisitos establece el análisis?**

Requisitos funcionales y no funcionales.

1. **¿Qué se establece en la fase de análisis?**

Se establece como se va a llevar a cabo el proyecto en cuestión.

1. **¿En qué técnica las entrevistas están dirigidas a la alta dirección?**

En la planificación conjunta de requisitos.

1. **¿Qué se obtiene al terminar la fase de análisis?**

Se obtiene el ERS, que establece la especificación de los requisitos del Software.

1. **¿En qué técnica participan muchos tipos de miembros del proyecto y se apoya en la dinámica de grupos?**

En el desarrollo conjunto de aplicaciones.

1. **¿Qué es un requisito funcional?**

Son los servicios que el sistema debe proporcionar y las funciones que tendrá que realizar.

1. **¿Qué es un requisito no funcional?**

Son las restricciones que van a afectar al sistema.

1. **¿Un requisito que pide que el usuario pueda hacer una copia de seguridad de que tipo es?**

Es de tipo funcional, ya que es una interacción con el usuario.

1. **Enumera las fases del desarrollo del software**

-Análisis de requisitos

-Diseño

-Codificación

-Pruebas

-Documentación

-Explotación

-Mantenimiento

1. **¿Un requisito que pide que el usuario pueda visualizar en un mapa una ruta en coche de que tipo es?**

Es de tipo funcional, ya que es una interacción con el usuario y una función que la aplicación debe prestar al usuario.

1. **¿Qué representan las entidades externas en un DFD?**

Representan los componentes externos al sistema, es decir, que no forman parte del mismo y proporcionan y reciben datos.

1. **¿En qué técnica de representación de requisitos se representan con rectángulos las entidades y con rombos la relaciones?**

En los DER. Diagramas Entidad / Relación.

1. **¿Qué representan los flujos de datos en un DFD?**

Una descripción detallada de los datos utilizados por el sistema.

1. **¿Qué técnica consiste en hablar con el cliente?**

En las entrevistas.

1. **¿Qué técnicas se utilizan en el análisis?**

-Entrevistas

-Desarrollo conjunto de aplicaciones

-Planificación conjunta de requisitos

-Brainstorming

1. **¿Qué técnica de representación de requisitos representa el flujo de los datos entre los distintos procesos, entidades externas y almacenes que forman el sistema?**

Los diagramas de flujo de datos (DFD).

1. **¿Qué técnica de representación de requisitos describe detalladamente los datos utilizados por el sistema?**

El diccionario de datos (DD).

1. **¿Qué identifican los procesos en un DFD?**

Identifican las funciones dentro del sistema.

1. **¿Qué técnica de representación de requisitos utiliza escenarios para describir como se usa la aplicación?**

Los casos de usos, utilizando UML.

1. **¿Un requisito que pide que la aplicación se pueda usar en dispositivos móviles de que tipo es?**

Es de tipo no funcional, ya que es parte de las restricciones del sistema.

1. **¿En qué técnica se generan ideas desde diferentes puntos de vista?**

En el brainstorming, así se puede ver el problema desde puntos de vista distintos, y porta una gran cantidad de ideas.

1. **¿Qué representan los almacenes en un DFD?**

Representan los datos desde un punto de vista estático.

1. **¿Qué técnicas se utilizan para representar los requisitos?**

-Diagramas de flujo de datos (DFD)

-Diagramas Entidad / Relación (DER)

-Diccionario de datos (DD)

-Prototipos

-Casos de uso (UML)

-Un modelo de dominio

1. **¿Qué especifica el documento de especificación de requisitos de software?**

Especifica la planificación de las reuniones que van a tener lugar, la relación entre los objetivos del usuario y del sistema, la relación de los requisitos funcionales y no funcionales del sistema, la relación de los objetivos prioritarios y temporización y el reconocimiento de requisitos mal planteados o que conllevan contradicciones.

## Diseño

### Preguntas

1. **¿Qué se realiza en esta fase?**

Se traducen los requisitos a una representación de software.

1. **¿Qué dos tipos de diseños se pueden utilizar?**

Diseño estructurado o diseño orientado a objetos.

### Diseño estructurado

1. **¿Qué cuatro niveles tiene el diseño clásico?**

-Diseño de datos

-Diseño arquitectónico

-Diseño de la interfaz

-Diseño a nivel de componentes

1. **¿Qué fase del diseño de estructurado diseña las estructuras de datos?**

En el diseño de datos.

1. **¿Qué fase del diseño de estructurado detalla la estructura de cada componente?**

El diseño a nivel de componentes.

1. **¿Qué fase del diseño de estructurado divide el proyecto en módulos?**

El diseño arquitectónico.

1. **¿Qué fase del diseño estructurado crea los formatos de pantalla?**

El diseño de la interfaz.

1. **¿Qué fase del diseño de estructurado utiliza diagramas de flujo, pseudocódigo etc?**

El diseño a nivel de componentes.

1. **¿Qué fase del diseño de estructurado diseña la estructura de componentes de software?**

El diseño arquitectónico

### Diseño orientado a objetos

1. **¿Qué requisito debe cumplir la fase de análisis para poder realizar un diseño orientado a objetos?**

Debe ser un análisis orientado a objetos, en el que se determinen las clases necesarias, atributos, etc.

1. **¿Qué cuatro capas define el diseño orientado a objetos?**

-Subsistema

-Clases y objetos

-Mensajes

-Responsabilidades

1. **¿Qué es UML?**

Es un lenguaje de modelado basado en diagramas.

## Codificación

### Preguntas

1. **¿Qué ventajas proporciona el escribir código fácil de leer?**

Facilita su desarrollo y su futuro mantenimiento.

1. **¿Qué característica deseable del código requiere que el código haga lo especificado en los requisitos?**

La corrección.

1. **¿Qué implica la programación eficiente?**

Que hace un buen uso de los recursos.

1. **¿Qué permite un código portable?**

Que se puede implementar en cualquier equipo.

1. **¿Qué característica deseable del código lo divide en trozos pequeños?**

La modularidad.

#### Fases en la obtención de código

1. **Define código fuente**

Es el conjunto de instrucciones que deberá realizar el ordenador. Lo escriben los programadores utilizando un editor de texto.

1. **Define algoritmo**

Es el conjunto de pasos a seguir para solucionar el problema.

1. **¿En qué lenguaje escribimos un código fuente y en que lenguaje debe recibirlo un ordenador?**

Se escribe en lenguaje de alto nivel y el ordenador lo recibe en lenguaje máquina.

1. **¿Cuáles son los pasos para obtener el código fuente de una aplicación?**

-Se debe empezar desde las anteriores etapas de análisis y diseño.

-Se diseña un algoritmo con los pasos a seguir para la solución del problema

-Se elige un lenguaje de programación apropiado para el software que se quiere codificar

-Se procede a la codificación del algoritmo previamente diseñado

1. **¿Qué es el código objeto?**

Es un código intermedio. Es decir es un código equivalente a la traducción del código fuente, pero no puede ser ejecutado directamente todavía.

1. **¿Mediante qué dos procesos se puede obtener un código objeto?**

Mediante compilación o mediante interpretación.

1. **¿En qué se diferencian la compilación y la interpretación?**

En la compilación se realiza sobre todo el código en un paso, mientras que en la interpretación se realiza línea a línea.

1. **¿Cómo se denomina el proceso de unir todos los archivos de código objeto?**

Se denomina enlazar.

1. **¿Cómo se denomina el software que realiza el proceso de enlazado?**

Se denomina linker (enlazador).

1. **¿Se puede ejecutar el código obtenido del proceso de compilación?**

No, se debe enlazar previamente para ser ejecutable.

1. **¿Se puede ejecutar el código obtenido del proceso de enlazado?**

Sí.

### Máquinas Virtuales

1. **¿Cuál es la función principal de las máquinas virtuales?**

Separa el funcionamiento del ordenador de los componentes hardware.

1. **¿Qué nos permite el uso de una máquina virtual?**

Podemos desarrollar y ejecutar una aplicación sobre cualquier equipo, y garantiza la portabilidad de las aplicaciones

1. **¿Cuáles son las funciones principales de una máquina virtual?**

-Conseguir que las aplicaciones sean portables

-Reservar memoria para los objetos que se crean y liberar la memoria no utilizada

-Comunicarse con el sistema donde se instala la aplicación para controlar los dispositivos hardware.

-Cumplir las normas de seguridad de las aplicaciones.

#### Entornos de ejecución

1. **¿Qué es un entorno de ejecución?**

Es un servicio de máquina virtual que sirve como base software para la ejecución de los programas.

1. **¿De qué partes consta el entorno de ejecución?**

La máquina virtual y los API’s.

1. **¿Qué es el runtime?**

Es el tiempo que tarda un programa en ejecutarse.

1. **¿Cuáles son las funciones de los entornos de ejecución?**

-Configurar la memoria principal disponible

-Enlazar los archivos del programa con las bibliotecas y los subprogramas

-Depurar los programas

1. **¿Por qué la máquina virtual y las API’s se suelen distribuir juntas?**

Porque necesitan ser compatibles entre sí.

1. **¿A quién pertenece el entorno de ejecución?**

A veces al sistema operativo, pero se puede instalar también como software independiente.

#### Java runtime environment

1. **¿Qué es el Java RuntimeEnvironment (JRE)?**

Son un conjunto de utilidades que permitirá la ejecución de programas java sobre cualquier plataforma.

## Pruebas

### Preguntas

1. **¿Las pruebas de caja negra y caja blanca son pruebas de verificación o de validación?**

Son pruebas de validación.

1. **¿En qué tipo de pruebas se intenta comprobar si el software implementa correctamente la función para la que está diseñado?**

En las pruebas de verificación.

1. **¿Qué pruebas se centran en validar los requisitos funcionales sin fijarse en el funcionamiento interno del programa?**

Las pruebas de caja negra.

1. **¿Qué dos tipos de pruebas se realizan para comprobar que hemos construido el software de forma correcta?**

Pruebas de verificación y pruebas de validación.

1. **¿En qué tipo de pruebas se intenta comprobar si el software cumple los requisitos del cliente?**

En las pruebas de validación.

1. **¿Qué pruebas se centran en validar la estructura interna del programa?**

Las pruebas de caja blanca.

# Documentación

## Preguntas

1. **¿En qué documento queda reflejada la información para la puesta en marcha de la aplicación?**

En la guía de instalación.

1. **¿En qué documento quedan reflejadas las pruebas realizadas?**

En la guía técnica.

1. **¿Cuál es el objetivo de la guía de instalación?**

Garantizar que la implementación de la aplicación se haga de manera segura, confiable y precisa.

1. **¿En qué documento quedan reflejados ejemplos de uso del programa?**

En la guía de uso.

1. **¿Cuál es el objetivo de la guía técnica?**

Facilitar un correcto desarrollo, hacer correcciones y permitir su futuro mantenimiento.

1. **¿En qué documento quedan reflejados la solución de posibles problemas que puedan surgir al utilizar la aplicación?**

En la guía de uso.

1. **¿En qué documento queda reflejada información sobre la seguridad del sistema?**

En la guía de instalación.

1. **¿A quién va dirigida la guía de instalación?**

Al personal informático que se encarga de la instalación, y a los usuarios que la van a utilizar.

1. **¿En qué documento queda reflejado la codificación de los programas?**

En la guía técnica.

1. **¿A quién va dirigida la guía técnica?**

Al personal técnico. Analistas y programadores.

1. **¿En qué documento queda reflejada la descripción de la funcionalidad de la aplicación?**

En la guía de uso.

1. **¿En qué documento queda reflejado el diseño de la aplicación?**

En la guía técnica.

1. **¿Qué fases de un proyecto es necesario documentar?**

Todas las fases deben ser documentadas a la perfección.

1. **¿En qué documento quedan reflejados requisitos software de la aplicación?**

En la guía de uso.

1. **¿A quién va dirigida la guía de uso?**

A los usuarios que utilizarán la aplicación.

1. **¿Cuál es el objetivo de la guía de uso?**

Dar información para poder utilizar la aplicación a los usuarios que la vayan a usar.

1. **¿En qué documento queda reflejada la forma de comenzar a ejecutar la aplicación?**

En la guía de uso.

# Explotación

## Preguntas

1. **¿En qué consiste la fase de explotación?**

Es la fase en la que los usuarios conocen la aplicación y la empiezan a utilizar.

1. **¿En qué consisten las beta test?**

Son las últimas pruebas que se realizan. Se hacen en los equipos del cliente.

1. **¿Una vez se alcanza la fase de explotación del software es necesaria alguna labor por parte del equipo de desarrollo?**

No, una vez distribuido, se deberá dar soporte al usuario cuando la solicite.

1. **¿Quién se encarga de realizar las configuraciones en la fase de configuración?**

La pueden hacer tanto el programador como el usuario siempre y cuando tenga la guía de instalación.

1. **¿Quién debería realizar la configuración del software si es una aplicación hecha a medida del cliente?**

Debería ser el fabricante el encargado de la configuración.

# Mantenimiento

## Preguntas

1. **¿Cómo se denomina a los cambios que se realizan en el software para adaptarse a las nuevas tendencias del mercado, a nuevos componentes hardware, etc.?**

Cambios adaptativos.

1. **¿Cómo se denomina a los cambios que se realizan en el software para satisfacer nuevas necesidades del cliente?**

Cambios evolutivos.

1. **La duración de la etapa de mantenimiento es corta. ¿Verdadero o falso?**

Falso. Es la etapa más larga de todo el proceso de construcción de software.

1. **¿Cómo se denomina a los cambios que se realizan en el software para mejorar la funcionalidad del software?**

Cambios perfectivos.

1. **¿Cómo se denomina a los cambios que se realizan en el software para eliminar errores?**

Cambios correctivos.

# Lenguajes de Programación

## Preguntas

1. **Enumera los 4 tipos de lenguajes de programación en orden evolutivo**

Lenguaje máquina >> Lenguaje ensamblador >> Lenguaje de alto nivel >> Lenguaje visual.

1. **Define lenguaje de programación**

Es un idioma creado de forma artificial, el cual nos sirve para programar y luego obtener un código que el ordenador pueda entender y ejecutar.

1. **¿Qué tipo de lenguaje de programación utiliza mnemotécnicos?**

El lenguaje ensamblador.

1. **¿Qué tipo de lenguaje de programación es único para cada procesador?**

El lenguaje máquina.

1. **¿Qué tipo de lenguaje de programación utiliza sentencias y órdenes derivadas del idioma inglés?**

El lenguaje de alto nivel basado en código.

1. **¿En qué tipo de lenguaje de programación se programa gráficamente usando el ratón?**

En los lenguajes visuales.

1. **¿Qué tipo de lenguaje de programación entiende directamente el ordenador y no necesita por lo tanto traducción?**

El lenguaje máquina.

## Concepto y características

### Preguntas

1. **¿Cuáles son los dos tipos de lenguajes de bajo nivel?**

Lenguaje ensamblador y leguaje máquina.

1. **¿El lenguaje LISP que tipo de lenguaje es según la clasificación de la forma de ejecutarse?**

Es un lenguaje interpretado.

1. **¿Cómo se denomina a los lenguajes que están más próximos al funcionamiento interno del ordenador?**

Lenguajes de bajo nivel.

1. **¿Cómo se denomina a los lenguajes que utilizan un enlazador o linker?**

Lenguajes compilados.

1. **¿Cómo se denomina a las normas de construcción permitidas de los símbolos del lenguaje en un lenguaje de programación?**

Sintaxis.

1. **¿Cómo se denomina a los lenguajes hechos para un objetivo muy concreto?**

Lenguajes de propósito específico.

1. **¿Cómo se denomina a los lenguajes diseñados para realizar sistemas operativos o drivers?**

Lenguajes de programación de sistemas.

1. **¿De qué tres elementos está formado un lenguaje de programación?**

De un alfabeto, una sitaxis y una semántica.

1. **¿Cómo se denomina a los lenguajes que primero pasan por una fase de compilación y luego es interpretado?**

Lenguajes mixtos.

1. **¿Cómo se denomina a los lenguajes que indican que hay que hacer?**

Los lenguajes declarativos.

1. **¿Cómo se denomina al conjunto de símbolos permitidos por un lenguaje de programación?**

Alfabeto.

1. **¿Cómo se denomina a los lenguajes en los que un programa ejecuta las instrucciones del programa de manera directa?**

Lenguajes interpretados.

13. **¿El lenguaje LISP que tipo de lenguaje es según la clasificación del paradigma de programación?**

Es un lenguaje declarativo,

14. **¿Cómo se denomina a los lenguajes que están más próximos al razonamiento humano?**

Lenguajes de alto nivel.

15. **¿El lenguaje C que tipo de lenguaje es según la clasificación del paradigma de programación?**

Es un lenguaje imperativo.

16. **¿El lenguaje Csound que tipo de lenguaje es según el tipo de problemas que resuelve?**

Es un lenguaje de propósito general

17. **¿Cómo se denomina al significado de las construcciones para hacer acciones válidas en un lenguaje de programación?**

Semántica

18. **¿Cómo se denomina a los lenguajes aptos para todo tipo de tareas?**

Lenguajes de propósito general.

19. **¿El lenguaje C que tipo de lenguaje es según el tipo de problemas que resuelve?**

Es un lenguaje de propósito general.

20. **¿Cómo se denomina a los lenguajes diseñados para realizar tareas de control y auxiliares?**

Lenguajes de script.

21. **¿Cómo se denomina a los lenguajes en los que un programa traductor traduce el código del programa (código fuente) en código máquina (código objeto)?**

Lenguajes compilados.

22. **¿Cómo se denomina a los lenguajes que indican como hay que hacer la tarea?**

Lenguajes imperativos.

23. **¿El lenguaje JAVA que tipo de lenguaje es según la clasificación del paradigma de programación?**

Es un lenguaje de programación orientado a objetos.

24. **¿El lenguaje C que tipo de lenguaje es según la clasificación de la forma de ejecutarse?**

Es un lenguaje compilado.

## Lenguajes de programación estructurados

### Preguntas

1. **¿Qué tres tipos de sentencia permite la programación estructurada?**

Secuencias secuenciales, selectivas y repetitivas.

1. **¿Cuáles son las ventajas de la programación estructurada?**

Programas fáciles de leer, mantenimiento sencillo y estructura clara y sencilla.

1. **¿Cuáles son las desventajas de la programación estructurada?**

No permite reutilización del código eficaz y todo el programa se centra en un bloque.

1. **¿Hacía que evolucionó la programación estructurada?**

Hacia la modular, que divide el programa en módulos en lugar de un solo bloque.

## Lenguajes de programación orientados a objetos

### Preguntas

1. **¿El lenguaje Pascal es estructurado u orientado a objetos?**

Estructurado.

1. **¿De qué están compuestos los programas orientados a objetos?**

De clases y objetos.

1. **¿El lenguaje Java es estructurado u orientado a objetos?**

Es orientado a objetos.

1. **¿Qué ventajas presenta la programación orientada a objetos?**

Es más fácil de depurar y por tanto de encontrar errores, y el código es reutilizable.

1. **¿El lenguaje C es estructurado u orientado a objetos?**

Es estructurado.

1. **¿Qué nos permite diferenciar los objetos entre sí en un programa orientado a objetos?**

Los podemos diferenciar por los atributos.

1. **¿Qué nombre recibe una colección de objetos con características similares?**

Se llama clase.

1. **¿Cómo se comunican los objetos con otros produciéndose un cambio de estado de los mismos?**

Mediante métodos.

1. **¿El lenguaje C++ es estructurado u orientado a objetos?**

Es orientado a objetos.

# Herramientas de apoyo al desarrollo del software

## Preguntas

1. **¿Qué es un IDE?**

Es un entorno de programación.

1. **¿Cada IDE puede funcionar con diferentes lenguajes de programación o está diseñado para un lenguaje en concreto?**

Cada uno puede funcionar con distintos lenguajes.

1. **¿Qué es un framework?**

Es una estructura que sirve para ayudar al programador y no empezar desde cero, sino desde una base previa.

1. **¿Qué ventajas presenta el uso de un framework?**

Diseño uniforme y portabilidad de aplicaciones, y desarrollo rápido y reutilización de partes de código.

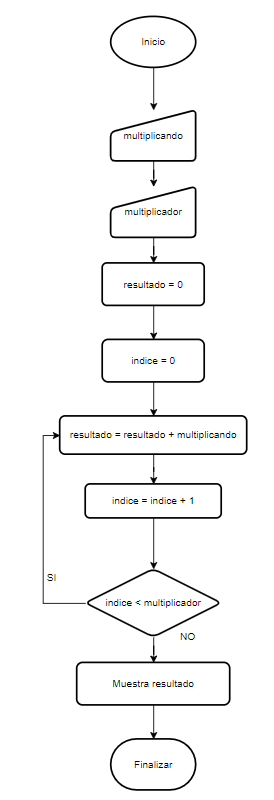
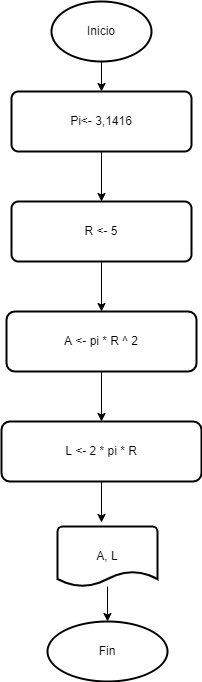
1. **¿Qué desventajas presenta el uso de un framework?**

Hay una dependencia del código del framework, y su utilización consume bastantes recursos de sistema

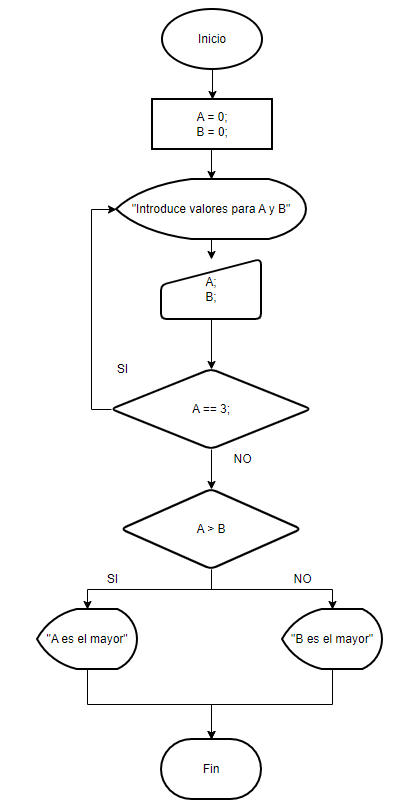
1. **Pon dos ejemplos de frameworks:**

NET y Spring de Java.

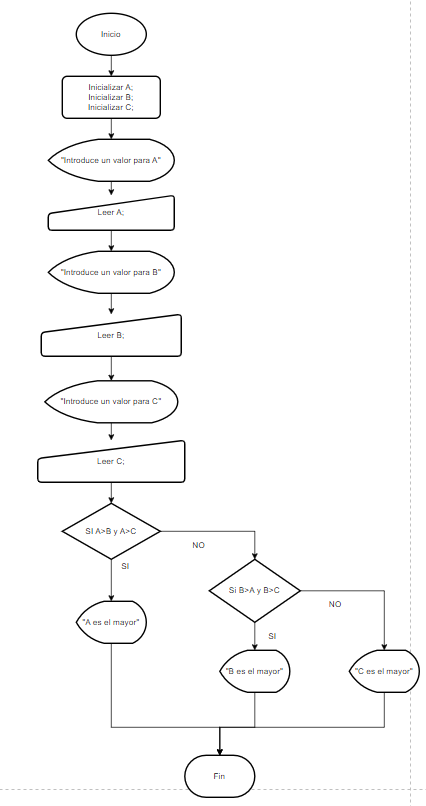
# **Diagramas de flujo**



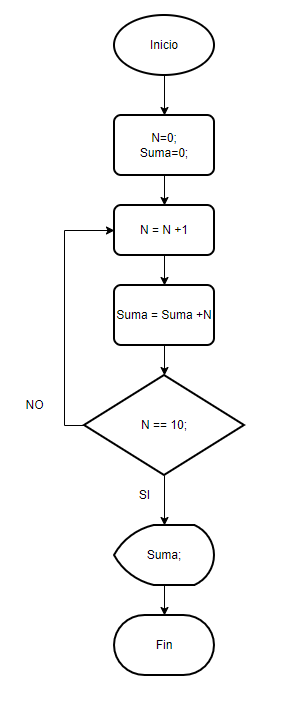
Ejercicio 1



Ejercicio 2

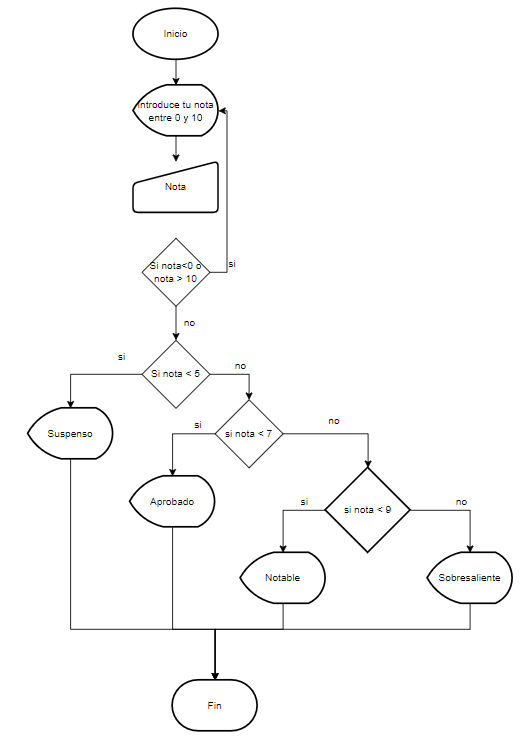


Ejercicio 3

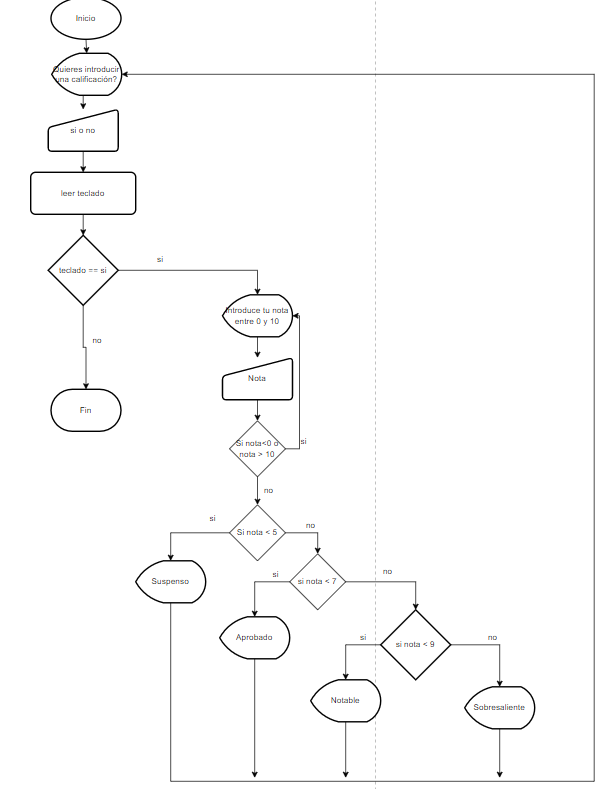


Ejercicios extra

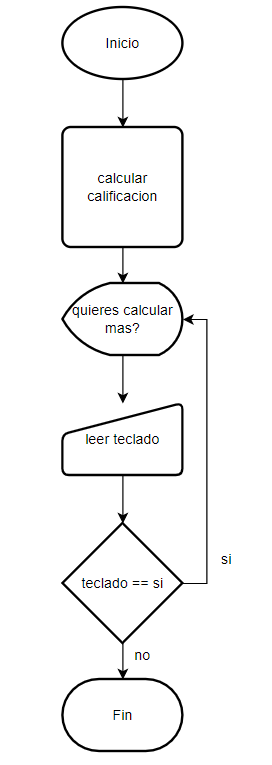
Ej1.



Ej 2.



Ej 3.



Ej 4.

