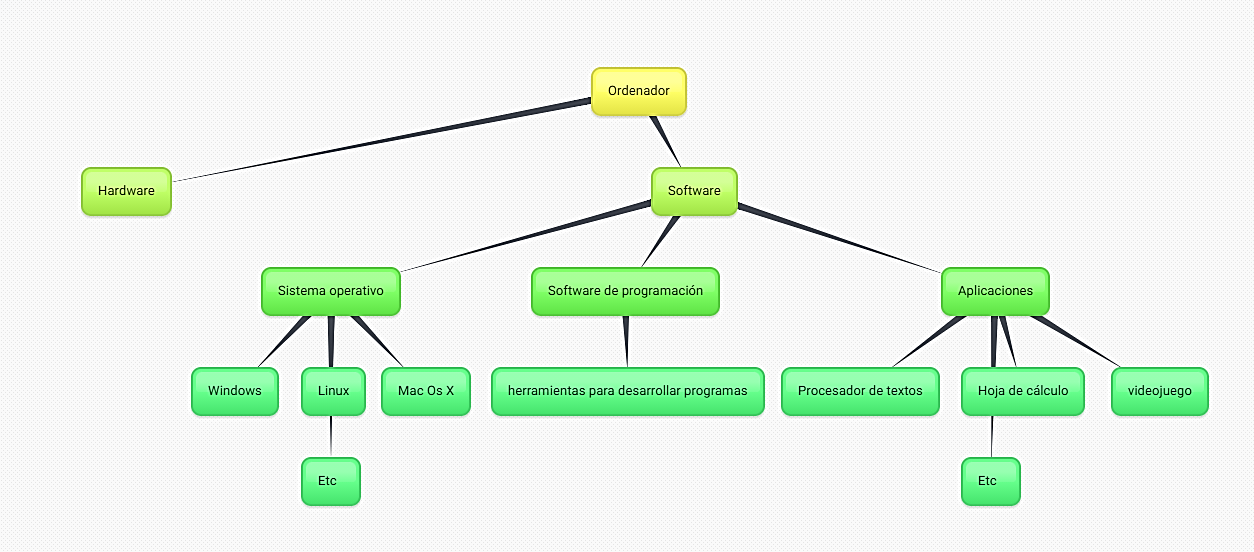
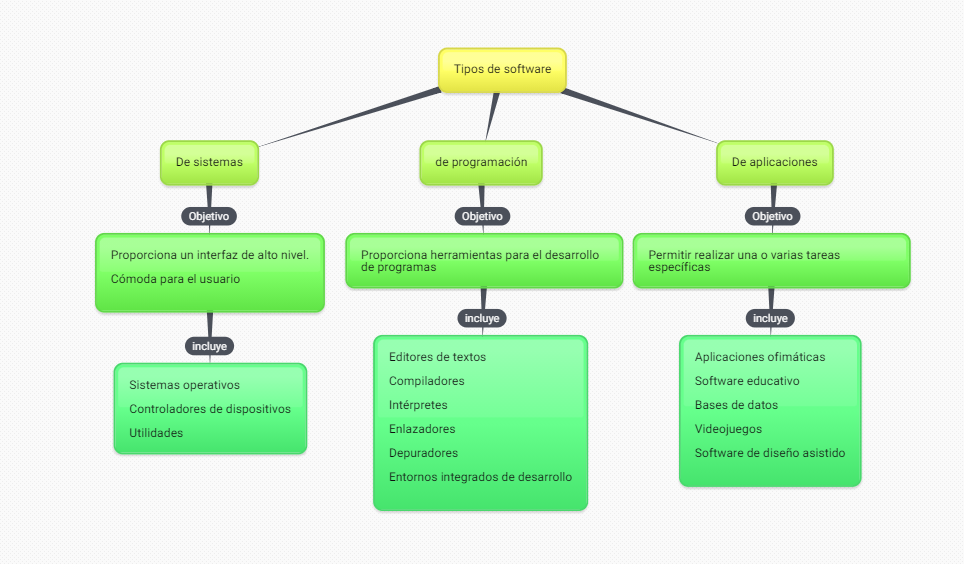
# Unidad 1: Desarrollo de software

### 1.1.- Software y programa. Tipos de software.

#### 1.1.1. Mapa conceptual





#### 1.1.2. Preguntas

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

Software

¿Qué tipo de software es un videojuego?

De aplicaciones

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

Software de programación

¿Qué tipo de software es un depurador?

Software de programación

¿Qué tipo de software es un compilador?

Software de programación

¿De qué dos partes consta un ordenador?

Hardware y software

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

Sistema operativo

¿Qué tipo de software es un controlador de dispositivo?

Software de sistemas

¿Qué tipo de software es una aplicación ofimática?

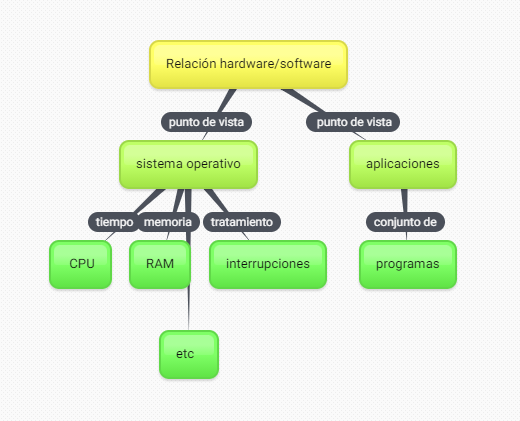
Software de aplicaciones

¿Qué tipo de software es un sistema operativo?

Software de sistemas

### 1. **2.- Relación hardware – software**

### **1.2.1. Mapa conceptual**



#### 1.2.2. Preguntas

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

El sistema operativo es el encargado de coordinar al hardware durante el funcionamiento del ordenador, actuando como intermediario entre éste y las aplicaciones que están corriendo en un momento dado

Define hardware

Conjunto de dispositivos físicos que conforman un ordenador

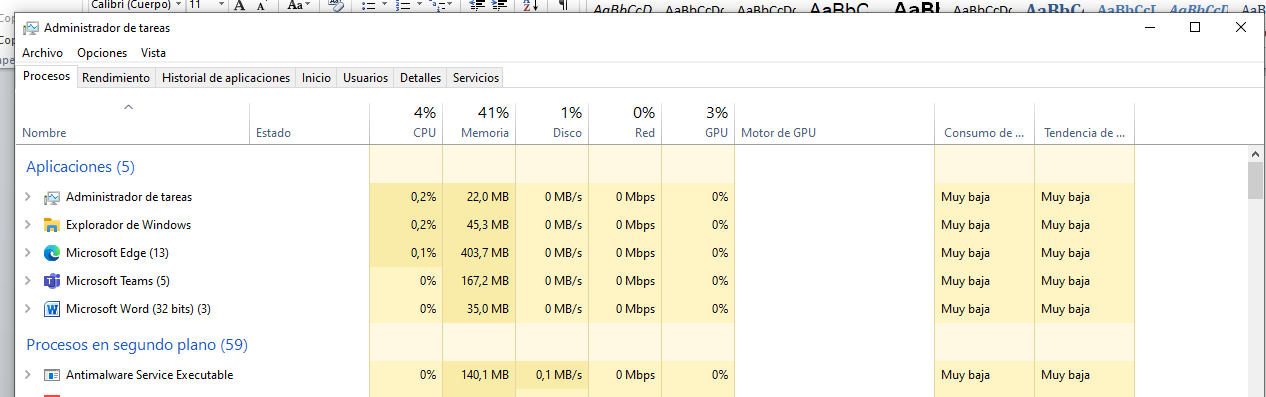
Investiga en google cual fue la gran aportación de la Arquitectura Von Neumann. (Pista busca el concepto de programa almacenado)

Un computador con una [arquitectura de von Neumann](https://es.wikipedia.org/wiki/Arquitectura_de_von_Neumann) almacena los datos y las instrucciones del programa en la misma memoria Un computador con una [arquitectura de von Neumann](https://es.wikipedia.org/wiki/Arquitectura_de_von_Neumann) almacena los datos y las instrucciones del programa en la misma memoria

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

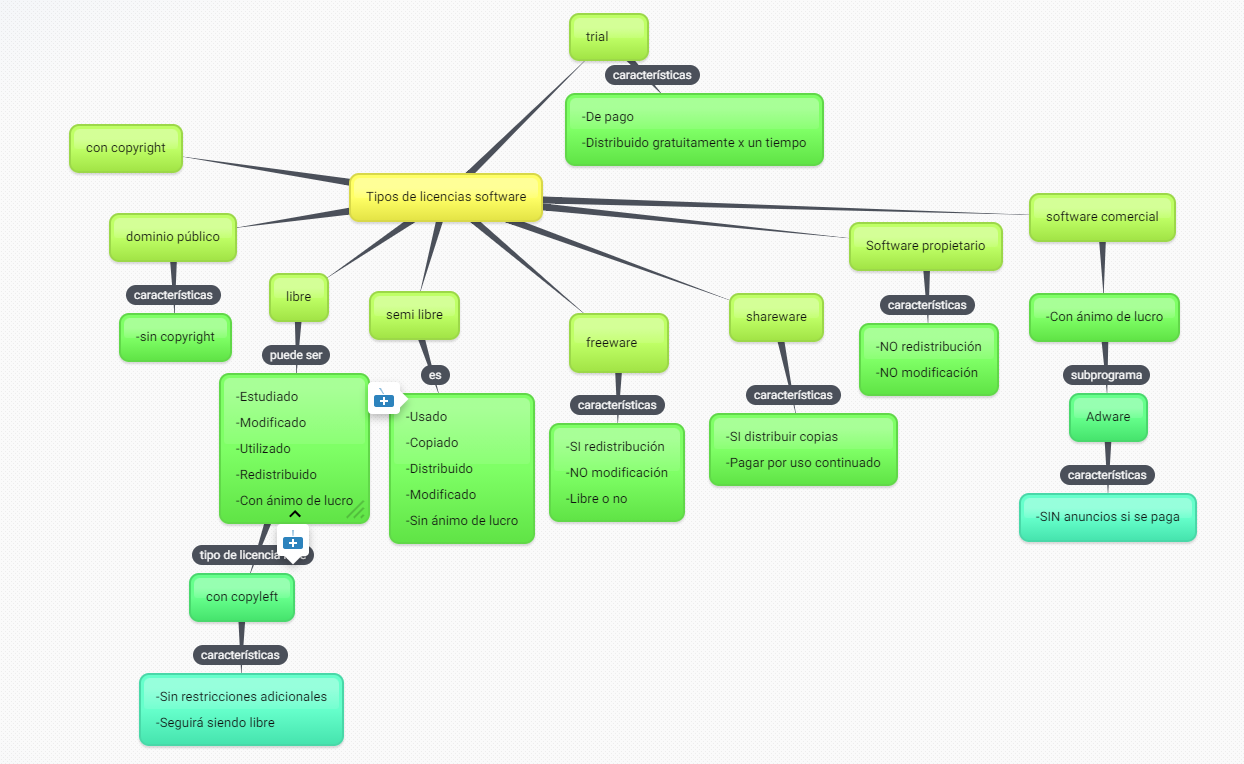
El software habla en lenguaje de programación y el hardware habla en lenguaje máquina; las aplicaciones hacen q puedan transformar el lenguaje de programación en lenguaje máquina.

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).



### 1.3.- Licencias de Software. Software libre y propietario

### 1.3.1. Mapa conceptual



#### 1.3.2. Preguntas

¿En qué se parecen el Shareware y el Freeware?

En que sí se pueden distribuir.

¿Quién elige la licencia de distribución de un software?

Es el desarrollador, el que elige la licencia según la cual distribuye el software.

¿Cómo se denomina a un conjunto de derechos exclusivos garantizados por un gobierno o autoridad al inventor de un nuevo producto?

Una patente

¿Qué es el software semi libre?

Aquél que no es libre, pero viene con autorización de usar, copiar, distribuir y modificar para particulares sin fines de lucro

¿Qué es un programa adware?

Subprograma que descarga publicidad sobre otro programa principal. Esto ocurre cuando un programa tiene versiones comerciales o más avanzadas que necesitan ser compradas para poder ser utilizadas.

¿Qué forma de protección proporcionan las leyes para los autores de obras originales?

El copyright.

¿En qué se parecen el software libre y el semi libre?

Pueden ser usados, distribuidos y modificados.

¿Qué es el software de dominio público?

Aquél que no está protegido con copyright es decir carece de licencia o no hay forma de determinarla pues se desconoce al autor.

¿Qué es el software libre?

Es un software cuyo código fuente puede ser estudiado, modificado, y utilizado libremente con cualquier finalidad y redistribuido con cambios o mejoras sobre ellas

¿Cómo se denomina a un contrato que se establece entre el desarrollador de un software sometido a propiedad intelectual y a derechos de autor, y el usuario?

Licencia de software

¿Cómo se denomina al software uso, redistribución o modificación están prohibidos o necesitan una autorización?

Con copyright.

¿El software libre es gratuito?

Sí

¿En qué se diferencian el software libre y el semi libre?

El libre es gratuito y el semi libre puede serlo o no.

¿Cómo se denomina al software que programas que permiten la redistribución pero no la modificación (y su código fuente no está disponible)

Freeware

¿En qué se diferencian el Shareware y el Freeware?

El Freeware puede ser con ánimo de lucro o no, el Shareware hay un momento que es con ánimo de lucro

Investiga en internet la relación entre un software con licencia trial y un software shareware

Trial: Así se denomina tradicionalmente a las versiones **Shareware** que puedes probar durante determinado tiempo, siendo lo más común **30 días**. Algunas veces este período se amplía hasta los dos, tres y hasta cuatro meses, mientras que en otras ocasiones se recorta hasta medio mes o incluso unos pocos días.

La línea que lo separa con respecto a la definición de **Shareware** es muy estrecha y, de hecho, muchas veces se utiliza una u otra indistintamente para un mismo programa.

#### 1.4.- Ciclo de vida del Software

### 1.4.1. Mapa conceptual

#### 1.4.2. Preguntas

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

Cada etapa tiene como entrada uno o varios documentos procedentes de las etapas anteriores y produce otros documentos de salida

¿Qué entendemos por ciclo de vida del software?

Conjunto de fases por las que pasa el sistema que se está desarrollando desde que nace la idea inicial hasta que el software es retirado o reemplazado por otro más adecuado.

¿Qué entendemos por desarrollo del software?

Consta de una serie de pasos de obligado cumplimiento, pues sólo así podremos garantizar que los programas creados son eficientes, fiables, seguros y responden a las necesidades de los usuarios finales

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

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

El análisis

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

El modelo evolutivo

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

El mantenimiento, porque sólo es pulir alguna cosa que ha quedado obsoleta o algún pequeño fallo.

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

Modelo en cascada

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

Modelo en cascada

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

Modelo en cascada con retroalimentación.

¿Qué modelo está basado en el modelo en cascada con retroalimentación?

¿El evolutivo? Me imagino que se creó a partir del de cascada.

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

Modelos evolutivos

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

Modelo en cascada

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

En el modelo de cascada con retroalimentación puedes volver atrás para modificar, corregir...

#### 1.5. Fases en el desarrollo y ejecución del software

#### 1.5.1. Análisis

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

Funcionales y no funcionales.

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

Se decide qué hacer.

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

Planificación conjunta de requisitos

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

Se especifican y analizan todos los requisitos del sistema.

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

Desarrollo conjunto de aplicaciones.

1. ¿Qué es un requisito funcional?

Servicios que el sistema debe proporcionar; funciones, respuesta y cómo se comportarán las aplicaciones.

1. ¿Qué es un requisito no funcional?

Restricciones que afectaran al sistema.

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

Tipo no funcional.

1. Enumera las fases del desarrollo del software

-Análisis

-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?

Requisito no funcional.

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

Representan componentes que no forman parte del sistema

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

Diagramas Entidad-Relación

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

Representan los requisitos del sistema.

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

Se hace a través de una entrevista.

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?

Diseño de datos?

1. ¿Qué técnica de representación de requisitos describe detalladamente los datos utilizados por el sistema?
2. ¿Qué identifican los procesos en un DFD?
3. ¿Qué técnica de representación de requisitos utiliza escenarios para describir como se usa la aplicación?
4. ¿Un requisito que pide que la aplicación se pueda usar en dispositivos móviles de que tipo es?
5. ¿En qué técnica se generan ideas desde diferentes puntos de vista?
6. ¿Qué representan los almacenes en un DFD?
7. ¿Qué técnicas se utilizan para representar los requisitos?
8. ¿Qué especifica el documento de especificación de requisitos de software?

#### 1.5.2. Diseño

1. ¿Qué se realiza en esta fase?
2. ¿Qué dos tipos de diseños se pueden utilizar?

### **1.5.2.1. Diseño estructurado**

1. ¿Qué cuatro niveles tiene el diseño clásico?
2. ¿Qué fase del diseño de estructurado diseña las estructuras de datos?
3. ¿Qué fase del diseño de estructurado detalla la estructura de cada componente?
4. ¿Qué fase del diseño de estructurado divide el proyecto en módulos?
5. ¿Qué fase del diseño estructurado crea los formatos de pantalla?
6. ¿Qué fase del diseño de estructurado utiliza diagramas de flujo, pseudocódigo etc?
7. ¿Qué fase del diseño de estructurado diseña la estructura de componentes de software?

### 1.5.2.2. Diseño orientado a objetos

1. ¿Qué requisito debe cumplir la fase de análisis para poder realizar un diseño orientado a objetos?
2. ¿Qué cuatro capas define el diseño orientado a objetos?
3. ¿Qué es UML?

#### 1.5.3. Codificación

1. ¿Qué ventajas proporciona el escribir código fácil de leer?
2. ¿Qué característica deseable del código requiere que el código haga lo especificado en los requisitos?
3. ¿Qué implica la programación eficiente?
4. ¿Qué permite un código portable?
5. ¿Qué característica deseable del código lo divide en trozos pequeños?

### 1.6. Fases en la obtención del código

1. Define código fuente
2. Define algoritmo
3. ¿En qué lenguaje escribimos un código fuente y en que lenguaje debe recibirlo un ordenador?
4. ¿Cuáles son los pasos para obtener el código fuente de una aplicación?
5. ¿Qué es el código objeto?
6. ¿Mediante que dos procesos se puede obtener un código objeto?
7. ¿En qué se diferencian la compilación y la interpretación?
8. ¿Cómo se denomina el proceso de unir todos los archivos de código objeto?
9. ¿Cómo se denomina el software que realiza el proceso de enlazado?
10. ¿Se puede ejecutar el código obtenido del proceso de compilación?
11. ¿Se puede ejecutar el código obtenido del proceso de enlazado?

### 1.7. Máquinas virtuales

1. ¿Cuál es la función principal de las máquinas virtuales?
2. ¿Qué nos permite el uso de una máquina virtual?
3. ¿Cuáles son las funciones principales de una máquina virtual?

### 1.8. Sdg

### 1.8.1. Entornos de ejecución

1. ¿Qué es un entorno de ejecución?
2. ¿De qué partes consta el entorno de ejecución?
3. ¿Qué es el runtime?
4. ¿Cuáles son las funciones de los entornos de ejecución?
5. ¿Por qué la máquina virtual y las API’s se suelen distribuir juntas?
6. ¿A quién pertenece el entorno de ejecución?

### 1.8.2. Java RuntimeEnviroment

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

#### 1.9. Pruebas

1. ¿Las pruebas de caja negra y caja blanca son pruebas de verificación o de validación?
2. ¿En qué tipo de pruebas se intenta comprobar si el software implementa correctamente la función para la que está diseñado?
3. ¿Qué pruebas se centran en validar los requisitos funcionales sin fijarse en el funcionamiento interno del programa?
4. ¿Qué dos tipos de pruebas se realizan para comprobar que hemos construido el software de forma correcta?
5. ¿En qué tipo de pruebas se intenta comprobar si el software cumple los requisitos del cliente?
6. ¿Qué pruebas se centran en validar la estructura interna del programa?

#### 1.10. Documentación

1. ¿En qué documento queda reflejada la información para la puesta en marcha de la aplicación?
2. ¿En qué documento quedan reflejadas las pruebas realizadas?
3. ¿Cuál es el objetivo de la guía de instalación?
4. ¿En qué documento quedan reflejados ejemplos de uso del programa?
5. ¿Cuál es el objetivo de la guía técnica?
6. ¿En qué documento quedan reflejados la solución de posibles problemas que puedan surgir al utilizar la aplicación?
7. ¿En qué documento queda reflejada información sobre la seguridad del sistema?
8. ¿A quién va dirigida la guía de instalación?
9. ¿En qué documento queda reflejado la codificación de los programas?
10. ¿A quién va dirigida la guía técnica?
11. ¿En qué documento queda reflejada la descripción de la funcionalidad de la aplicación?
12. ¿En qué documento queda reflejado el diseño de la aplicación?
13. ¿Qué fases de un proyecto es necesario documentar?
14. ¿En qué documento quedan reflejados requisitos software de la aplicación?
15. ¿A quién va dirigida la guía de uso?
16. ¿Cuál es el objetivo de la guía de uso?
17. ¿En qué documento queda reflejada la forma de comenzar a ejecutar la aplicación?

#### 1.11. Explotación

1. ¿En qué consiste la fase de explotación?
2. ¿En qué consisten las beta test?
3. ¿Una vez se alcanza la fase de explotación del software es necesaria alguna labor por parte del equipo de desarrollo?
4. ¿Quién se encarga de realizar las configuraciones en la fase de configuración?
5. ¿Quién debería realizar la configuración del software si es una aplicación hecha a medida del cliente?

#### 1.12. Mantenimiento

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.?
2. ¿Cómo se denomina a los cambios que se realizan en el software para satisfacer nuevas necesidades del cliente?
3. La duración de la etapa de mantenimiento es corta. ¿Verdadero o falso?
4. ¿Cómo se denomina a los cambios que se realizan en el software para mejorar la funcionalidad del software?
5. ¿Cómo se denomina a los cambios que se realizan en el software para eliminar errores?

#### 1.13. Lenguajes de programación

1. Enumera los 4 tipos de lenguajes de programación en orden evolutivo
2. Define lenguaje de programación
3. ¿Qué tipo de lenguaje de programación utiliza mnemotécnicos?
4. ¿Qué tipo de lenguaje de programación es único para cada procesador?
5. ¿Qué tipo de lenguaje de programación utiliza sentencias y órdenes derivadas del idioma inglés?
6. ¿En qué tipo de lenguaje de programación se programa gráficamente usando el ratón?
7. ¿Qué tipo de lenguaje de programación entiende directamente el ordenador y no necesita por lo tanto traducción?

#### 1.14. Concepto y características

1. ¿Cuáles son los dos tipos de lenguajes de bajo nivel?
2. ¿El lenguaje LISP que tipo de lenguaje es según la clasificación de la forma de ejecutarse?
3. ¿Cómo se denomina a los lenguajes que están más próximos al funcionamiento interno del ordenador?
4. ¿Cómo se denomina a los lenguajes que utilizan un enlazador o linker?
5. ¿Cómo se denomina a las normas de construcción permitidas de los símbolos del lenguaje en un lenguaje de programación?
6. ¿Cómo se denomina a los lenguajes hechos para un objetivo muy concreto?
7. ¿Cómo se denomina a los lenguajes diseñados para realizar sistemas operativos o drivers?
8. ¿De qué tres elementos está formado un lenguaje de programación?
9. ¿Cómo se denominaa los lenguajes que primero pasan por una fase de compilación y luego es interpretado?
10. ¿Cómo se denominaa los lenguajes que indican **que** hay que hacer?
11. ¿Cómo se denomina al conjunto de símbolos permitidos por un lenguaje de programación?
12. ¿Cómo se denomina a los lenguajes en los que un programa ejecuta las instrucciones del programa de manera directa?
13. ¿El lenguaje LISP que tipo de lenguaje es según la clasificación del paradigma de programación?
14. ¿Cómo se denomina a los lenguajes que están más próximos al razonamiento humano?
15. ¿El lenguaje C que tipo de lenguaje es según la clasificación del paradigma de programación?
16. ¿El lenguaje Csound que tipo de lenguaje es según el tipo de problemas que resuelve?
17. ¿Cómo se denomina al significado de las construcciones para hacer acciones válidas en un lenguaje de programación?
18. ¿Cómo se denomina a los lenguajes aptos para todo tipo de tareas?
19. ¿El lenguaje C que tipo de lenguaje es según el tipo de problemas que resuelve?
20. ¿Cómo se denomina a los lenguajes diseñados para realizar tareas de control y auxiliares?
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)?
22. ¿Cómo se denominaa los lenguajes que indican **como** hay que hacer la tarea?
23. ¿El lenguaje JAVA que tipo de lenguaje es según la clasificación del paradigma de programación?
24. ¿El lenguaje C que tipo de lenguaje es según la clasificación de la forma de ejecutarse?

#### 1.15. Lenguajes de programación estructurados

1. ¿Qué tres tipos de sentencia permite la programación estructurada?
2. ¿Cuáles son las ventajas de la programación estructurada?
3. ¿Cuáles son las desventajas de la programación estructurada?
4. ¿Hacía que evolucionó la programación estructurada?

#### 1.16. Lenguajes de programación orientados a objetos

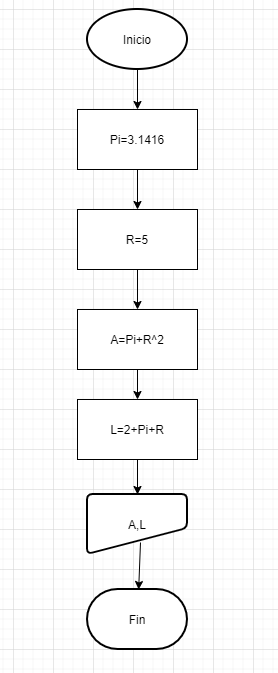
1. ¿El lenguaje Pascal es estructurado u orientado a objetos?
2. ¿De qué están compuestos los programas orientados a objetos?
3. ¿El lenguaje Java es estructurado u orientado a objetos?
4. ¿Qué ventajas presenta la programación orientada a objetos?
5. ¿El lenguaje C es estructurado u orientado a objetos?
6. ¿Qué nos permite diferenciar los objetos entre sí en un programa orientado a objetos?
7. ¿Qué nombre recibe una colección de objetos con características similares?
8. ¿Cómo se comunican los objetos con otros produciéndose un cambio de estado de los mismos?
9. ¿El lenguaje C++ es estructurado u orientado a objetos?

#### 1.17. Herramientas de apoyo al desarrollo del software

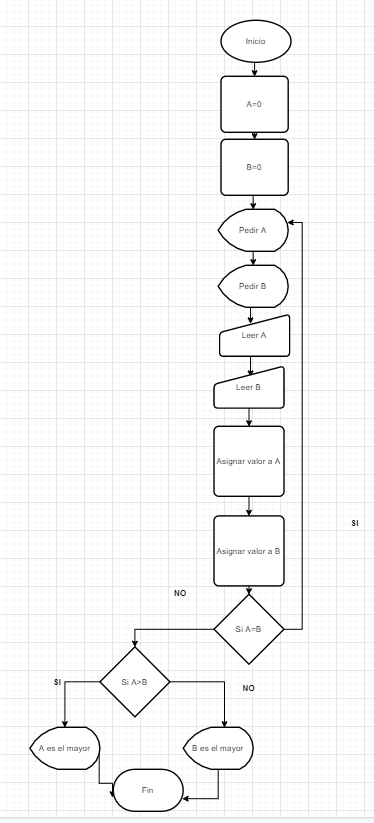
1. ¿Qué es un IDE?
2. ¿Cada IDE puede funcionar con diferentes lenguajes de programación o está diseñado para un lenguaje en concreto?
3. ¿Qué es un framework?
4. ¿Qué ventajas presenta el uso de un framework?
5. ¿Qué desventajas presenta el uso de un framework?
6. Pon dos ejemplos de frameworks

### 2.1Diagramas de flujo

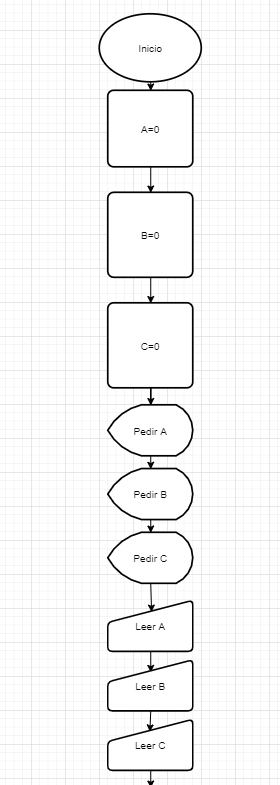
#### 2.1.1. Ejemplo de diagrama de flujo

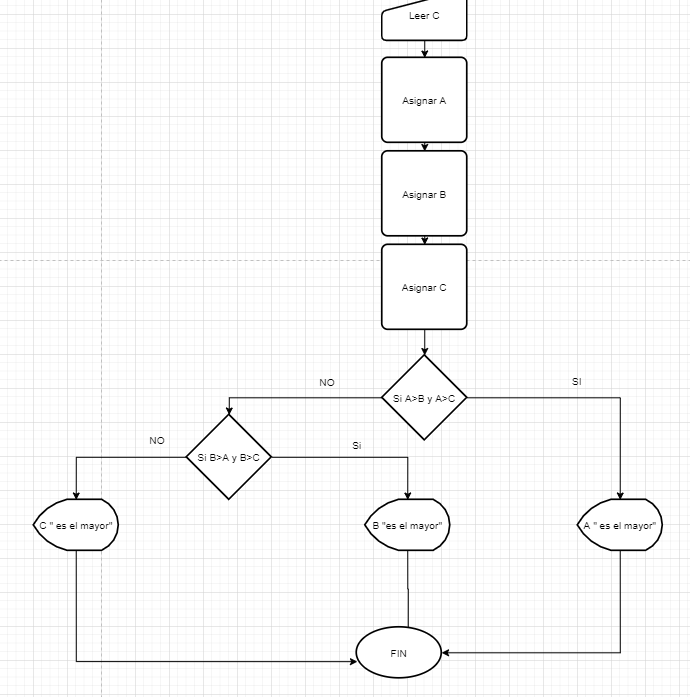


#### 2.1.1.2. Ejercicio 1



#### 2.1.1.3. Ejercicio 2





#### 2.1.1.4. Ejercicio 3

