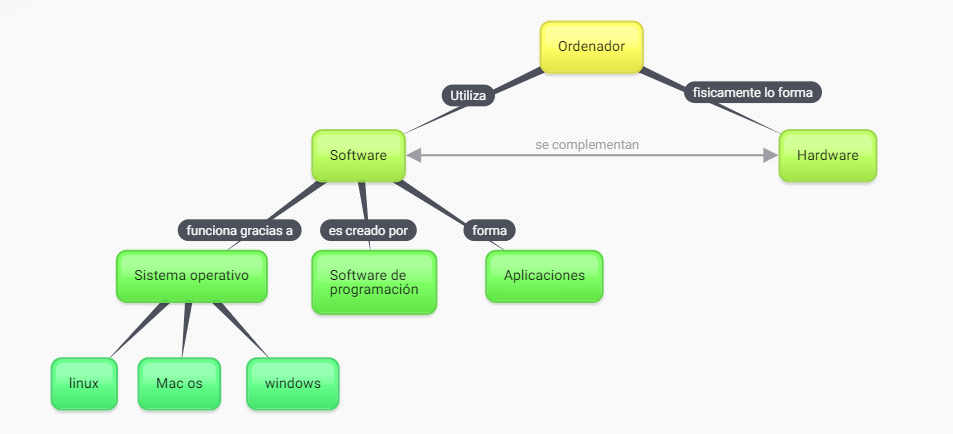
Unidad 1

# Software y programas. Tipos de software.

## Mapa conceptual:



## Preguntas y dudas:

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

Aplicaciones informáticas.

1. ¿Qué tipo de software es un videojuego?

Es un software de aplicación.

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

Software de programación.

1. ¿Qué tipo de software es un depurador?

Software de programación.

1. ¿Qué tipo de software es un compilador?

Software de programación

1. ¿De qué dos partes consta un ordenador?

Hardware y 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?

Sistema operativo.

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

Software de sistemas.

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

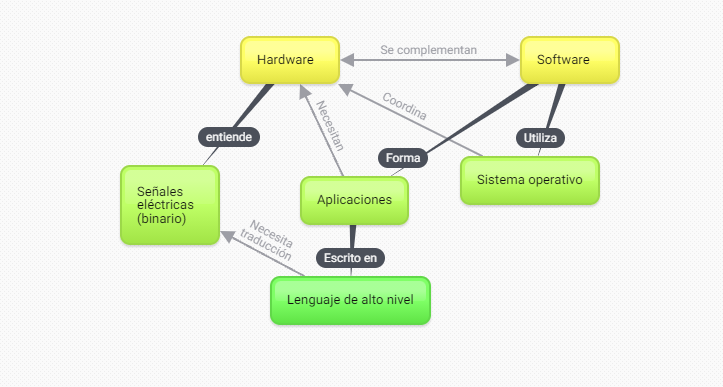
Software de aplicación.

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

Software de sistemas.

# Relación hardware-software.

## Mapa conceptual:



## Preguntas y dudas:

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

El sistema operativo (software) coordina el hardware.

2. Define hardware

Conjunto de dispositivos físicos que conforma un ordenador.

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

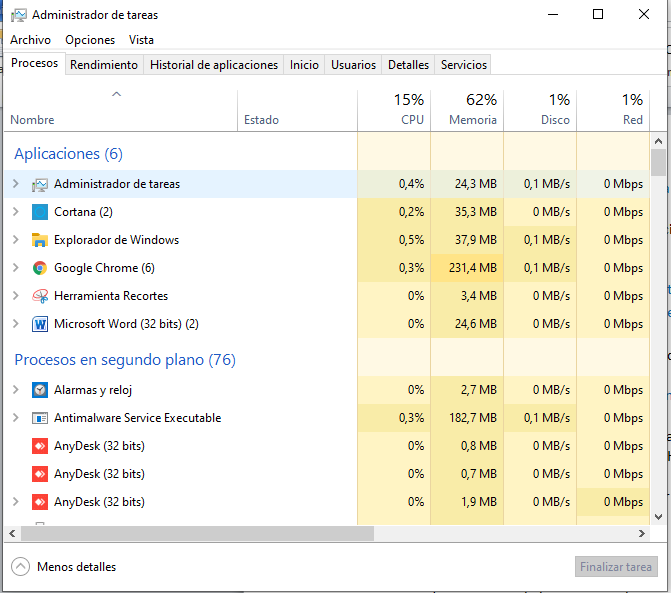
Poder almacenar no sólo los datos de una computación, sino también las instrucciones que la definen.

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

El hardware interpreta y ejecuta el lenguaje de programación (software).

5. 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).

Tecleando control+alt gr+supr, administrador de tareas, más detalles, procesos.



# 3. Licencias de Software.

## Preguntas y dudas:

1. ¿En qué se parecen el Shareware y el Freeware?

En que en ninguno puedes modificar el programa.

2. ¿Quién elige la licencia de distribución de un software?

El desarrollador del mismo.

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

Es una patente.

4. ¿Qué es el software semi libre?

El que puedes usar, copiar distribuir y modificar siendo un particular sin fines de lucro aunque no sea libre.

5. ¿Qué es un programa adware?

Un subprograma que descarga publicidad sobre otro programa a no ser que pagues una versión comercial de este.

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

El copyright o derecho de autor.

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

En que ambos pueden ser modificados copiados, distribuidos…

8. ¿Qué es el software de dominio público?

Aquel que no está protegido con copyright.

9. ¿Qué es el software libre?

Aquel que permite ser usado, estudiado, distribuido y mejorado.

10. ¿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?

La licencia de software.

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

Software propietario.

12. ¿El software libre es gratuito?

No tiene por qué serlo.

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

En que el semilibre no puede ser usado por empresas o particulares con fines de lucro.

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

Shareware.

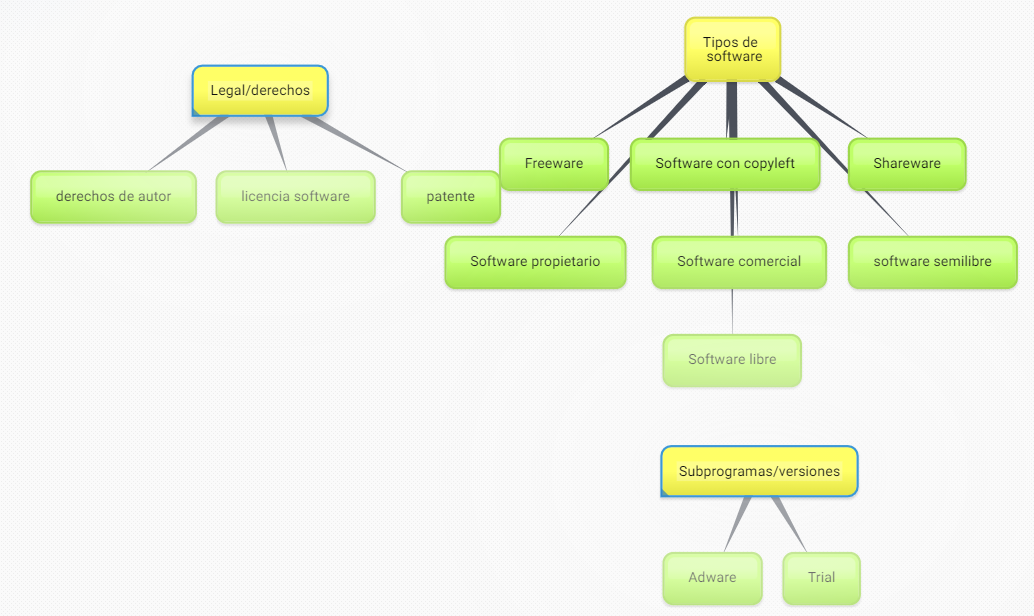
15. ¿En qué se diferencian el Shareware y el Freeware?

El shareware es de pago y el freeware no.

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

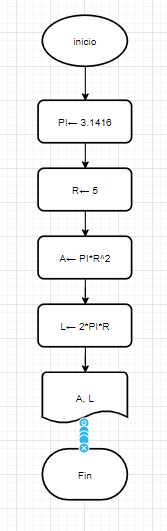
La licencia trial solo te permite usar el software shareware durante un tiempo determinado.

## Mapa conceptual:



# Diagramas de flujo.

## Diagrama1. Ejemplo:



# 4. Ciclo de vida del software

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

Análisis: en ella, se establecen los requisitos que define el cliente para su software.

Diseño: con los requisitos definidos, se establece ahora el diseño conceptual del programacon el fin de resolver el problema del cliente. Se define la **estructura del software.**

Codificación: se pasa el diseño conceptual al código máquina.

Pruebas: se comprueba con los criterios de corrección, su correcto funcionamiento.

Mantenimiento: cuando el programa pasa a las manos del cliente, se realiza unas comprobaciones de que se haya adaptado correctamente al entorno de su aplicación

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

Entendemos por ciclo de vida, el conjunto de fases, desde el inicio hasta que el software es retirado.

1. ¿Qué entendemos por desarrollo del software?

El desarrollo es el proceso por el cual, la idea conceptual se pasa al ordenador para su funcionamiento.

## 4.1 Modelos de ciclo de vida

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

El modelo evolutivo que se encarga del análisis de riesgos es el modelo en espiral.

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

La fase que conlleva mayor esfuerzo es el de las mejoras

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

El modelo que responde a esta característica es el modelo evolutivo, del tipo iterativo incremental.

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

La fase del ciclo que menor esfuerzo lleva es el de la implementación, debido a que es la parte más técnica del proceso y que “supone” menor esfuerzo mental

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

El modelo que realiza esto es 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?

El modelo aconsejable en este caso es el modelo en cascada, aunque no permite realizar cambios hacia atrás. Es solo aconsejable en proyectos muy bien definidos.

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

En este caso, el modelo que mejor responde a este enunciado es el modelo en cascada con retroalimentación, donde permite realizar algunos pequeños cambios volviendo para atrás.

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

Es el modelo iterativo incremental, donde las etapas se van refinando y dar lugar a mejoras en las siguientes fases.

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

Los modelos que realizan varias versiones más refinadas son las del tipo iterativo incremental y en espiral. Aunque se puede incluir al de cascado con retroalimentación, pero en menor grado.

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

Ese sería el más primitivo de todos, el modelo en cascada, ya que no se podrá después realizar cambios.

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

La principal diferencia sería la siguiente: en cascada simple no permite volver a los pasos anteriores, mientras que en cascada con retroalimentación, sí.

# 5. Fases en el desarrollo y ejecución del software

## 5.1 Análisis

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

En el análisis se establecen los requisitos funcionales y no funcionales del sistema.

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

En esta fase, se establece los pasos a seguir para conseguir el producto final deseado por el cliente, a partir de los requisitos que este establezca.

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

La técnica dirigida para la alta dirección es la planificación conjunta de requisitos.

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

Al terminar esta, obtenemos el documente ERS, el cual es un informe de las especificaciones de requisitos del software.

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

En la técnica de “Desarrollo conjunto de aplicaciones”

1. ¿Qué es un requisito funcional?

El requisito funcional es el conjunto de los servicios que el sistema debe proporcionar, es decir, las funciones que va a realizar la aplicación del cliente junto con su respuesta y comportamiento en situaciones inesperadas.

1. ¿Qué es un requisito no funcional?

Consiste en el grupo de las restricciones que afectan al sistema.

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

Este responde al tipo funcional, porque en ella, se le indica al sistema que debe hacer.

1. Enumera las fases del desarrollo del software

Análisis, Diseño, Construcción, Codificación, Pruebas, Instalación, Mantenimiento, Validación y verificación.

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

Este responde al tipo no funcional, ya que se le indica al sistema como debe hacer una tarea determinada ya previamente.

1. ¿Qué representan las externas en un DFD?

Las entidades externas representan a los componentes que no forman parte del sistema. Tienen forma de rectángulo.

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

Esta representación responde a la técnica “Diagrama Entidad/ Relación DER”.

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

Los flujos de datos representan los movimientos de los datos del sistema, a través de flechas.

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

La entrevista, la técnica más tradicional, consiste en hablar con el cliente.

1. ¿Qué técnicas se utilizan en el análisis?

⇾ Entrevista

⇾ 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?

Esa técnica de representación es denominada “Diagrama de flujo de datos DFD”

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

La técnica que descripción con lujo de detalles los datos utilizados por el sistema es el “Diccionario de datos DD”

# 18. ¿Qué identifican los procesos en un DFD?

# Los procesos identifican las funciones dentro del sistema, en forma de burbujas ovaladas.

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

# La técnica que se basa en escenarios que describen como se usa el software, es “Casos de uso”

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

# Se trata de un requisito funcional, al indicar que se debe hacer.

# 21. ¿En qué técnica se generan ideas desde diferentes puntos de vista?

# En la técnica del BrainStorming, los grupos de participantes generan ideas desde diferentes puntos de vista para la resolución de un problema.

# 22. ¿Qué representan los almacenes en un DFD?

# Los almacenes representan el lugar, desde el punto de vista estático, donde se guardan los datos y a su vez, donde se pueden recuperar.

# 23. ¿Qué técnicas se utilizan para representar los requisitos?

# Las técnicas empleadas para la representación de los requisitos son:

# Diagramas de flujo de datos (DFD)

# Diagrama Entidad/Relación (DER)

# Diccionario de Datos (DD)

# Prototipos

# Casos de uso

# Un modelo de dominio

# 24. ¿Qué especifica el documento de especificación de requisitos de software?

# En él, se especifican los siguientes puntos:

# Las reuniones que se van a realizar.

# Relación cliente-sistema, desde el punto de vista de los objetivos.

# Relación requisitos funcionales-no funcionales.

# Relación objetivos prioritarios-temporización.

# Reconocimiento de requisitos mal planteados o con contradicciones.

## 5.2. Diseño

# 1. ¿Qué se realiza en esta fase?

# En esta fase, se traducen los requisitos funcionales y no funcionales, previamente definidos, en una representación del software.

# 2. ¿Qué dos tipos de diseños se pueden utilizar?

# Los dos tipos de diseño que se pueden emplear son el diseño estructurado y el orientado a objetos

# 3. ¿Qué cuatro niveles tiene el diseño clásico?

# Los cuatro niveles que dispone este diseño son los siguientes:

# Diseño de datos

# Diseño arquitectónico

# Diseño de la interfaz

# Diseño a nivel de componentes (procedimental)

# 4. ¿Qué fase del diseño de estructurado diseña las estructuras de datos?

# El diseño de datos, a partir de los diagramas DER y DD, transforma los datos y relaciones en estructuras de datos.

# 5. ¿Qué fase del diseño de estructurado detalla la estructura de cada componente?

# El diseño a nivel de componentes (procedimental) se encarga del detalle de la estructura de cada componente.

# 6. ¿Qué fase del diseño de estructurado divide el proyecto en módulos?

# Se trata del diseño arquitectónico.

# 7. ¿Qué fase del diseño estructurado crea los formatos de pantalla?

# El diseño de la interfaz crea los formatos de pantalla

# 8. ¿Qué fase del diseño de estructurado utiliza diagramas de flujo, pseudocódigo etc.?

# El diseño a nivel de componentes (procedimental) emplead estos elementos, junto con los diagramas de caja, tablas de decisión, etc.

# 9. ¿Qué fase del diseño de estructurado diseña la estructura de componentes de software?

# El diseño arquitectónico se encarga de la representación de las estructuras de los componentes del software, junto con sus propiedad e interacciones en base a los DFD.

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

# El análisis debe estar orientado a objetos (AOO) para que sea posible posteriormente realizar un diseño orientado a objetos.

# 11. ¿Qué cuatro capas define el diseño orientado a objetos?

# Los niveles que se encuentran dentro de este diseño son:

# Subsistema

# Clases y objetos

# Mensajes

# Responsabilidades

# 

## 3. Codificación

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

# La principal ventaja que aporta que el código se escriba en una lectura fácil de leer, es que esto permite que su desarrollo y mantenimiento futuro sea más efectivo.

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

# La característica de la corrección es la que establece que el código realice lo que se pida realmente.

# 3. ¿Qué implica la programación eficiente?

# La programación eficiente supone el buen uso de los recursos.

# 4. ¿Qué permite un código portable?

# Esta característica permite que se pueda implementar el código en cualquier equipo.

# 5. ¿Qué característica deseable del código lo divide en trozos pequeños?

# La característica de modularidad establece que el código se divida en trozos pequeños.

# 6. Define código fuente

# El código fuente es el conjunto de instrucciones que el programa deberá realizar, escritas por los programadores en un lenguaje de alto nivel, utilizando un editor de texto.

# 7. Define algoritmo

# Un algoritmo es un conjunto de pasos a seguir para obtener la solución del problema.

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

# El código los escribimos en un inicio en nuestro lenguaje y después, lo traducimos al lenguaje máquina para que el ordenador pueda entenderlo.

# 9. ¿Cuáles son los pasos para obtener el código fuente de una aplicación?

# Los pasos necesarios para obtener el código fuente son:

# Las etapas de análisis y diseño.

# Diseño del algoritmo.

# Elección del “Lenguaje de Programación”.

# Codificación del algoritmo.

# 10. ¿Qué es el código objeto?

# El código objeto es un código intermedio, resultante del proceso de traducir el código fuente a un código binario que aún no puede ejecutarse en la computadora.

# 11. ¿Mediante que dos procesos se puede obtener un código objeto?

# El código objeto se pueden adquirir, a través de estos dos procesos:

# Compilación.

# Interpretación.

# 12. ¿En qué se diferencian la compilación y la interpretación?

# Estas se diferencian en los siguientes aspectos:

# Mientras que en el primero se traduce todo el código fuente en un solo paso, el segundo traducirá línea por línea.

# A consecuencia de la traducción masiva, el primero dispone de un código intermedio que deberá ser enlazado, a diferencia del segundo.

# El software responsable del primero es el compilador y del segundo, el intérprete.

# El segundo es recomendable para un programador inexperto, ya que, de esta forma, la detección de errores es más detallada.

# 13. ¿Cómo se denomina el proceso de unir todos los archivos de código objeto?

# El proceso de enlazar todos los archivos de código objeto, se denomina enlazar.

# 14. ¿Cómo se denomina el software que realiza el proceso de enlazado?

# El software empleado para este proceso es el “linker”.

# 15. ¿Se puede ejecutar el código obtenido del proceso de compilación?

# No se puede ejecutar este código, al ser uno de tipo objeto, que, en otras palabras, es código binario que no se puede ejecutar en el ordenador.

# 16. ¿Se puede ejecutar el código obtenido del proceso de enlazado?

# Este sí se puede ejecutar al haberse unido todos los códigos objetos, dando lugar a un código ejecutable.

# 17. ¿Cuál es la función principal de las máquinas virtuales?

# La función principal de estos es separar el funcionamiento del ordenador de los componentes hardware instalados.

# 18. ¿Qué nos permite el uso de una máquina virtual?

# Con estas máquinas, podemos desarrollar y ejecutar una aplicación sobre cualquier equipo, independientemente de las características de los componentes físicos.

# 19. ¿Cuáles son las funciones principales de una máquina virtual?

# Las funciones principales de las máquinas virtuales son:

# Portabilidad de las aplicaciones.

# Reserva memoria empleada y liberación de memoria.

# Comunicación máquina-sistema para el control de los dispositivos hardware.

# Cumplimiento de las normas de seguridad.

# 20. ¿Qué es un entorno de ejecución?

# Un entorno de ejecución es un servicio de la máquina virtual que sirve como base software para la ejecución de programas.

# 21.¿De qué partes consta el entorno de ejecución?

# Este consta de la máquina virtual y los API’s, que se suelen distribuir conjuntamente.

# 22. ¿Qué es el runtime?

# Se le denomina runtime al tiempo que el programa tarda en ejecutarse en el ordenador.

# 23. ¿Cuáles son las funciones de los entornos de ejecución?

# Las funciones que tienen los entornos de ejecución son:

# Configuración de la memoria principal del sistema

# Enlazamiento de los archivos del programa con las bibliotecas y subprogramas existentes y realizados.

# Depuración de los programas

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

# Estas precisan distribuirse conjuntamente porque estas deben ser compatibles entre sí.

# 25.¿A quién pertenece el entorno de ejecución?

# Este elemento le pertenece, en general, al propio sistema operativo, aunque en ocasiones puede ser un software independiente que funciona por debajo de la aplicación.

# 26. ¿Qué es el Java RuntimeEnvironment (JRE)?

# El JRE es el conjunto de utilidades que permitirá la ejecución de programas Java sobre cualquier tipo de plataforma.

# 

## 4. Pruebas

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

# Las pruebas de caja negra y blanca son del tipo verificación, ya que se encargan de validar los requisitos funcionales y la estructura interna del programa.

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

# Este enunciado hace referencia a las pruebas de verificación.

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

# Se trata de las pruebas de caja negra, que únicamente se centran en los requisitos funcionales.

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

# Los dos tipos de pruebas que se realizan para comprobar la correcta construcción del software son las pruebas de caja negra y blanca.

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

# En la prueba de validación, nos encargamos de comprobar que el producto cumple con los requisitos del cliente.

# 6. ¿Qué pruebas se centran en validar la estructura interna del programa?

# Las pruebas de caja blanca se centran en lo descrito en el enunciado.

## 5. Documentación

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

# La información para la puesta en marcha se encuentra en la guía de instalación.

# 2-. ¿En qué documento quedan reflejadas las pruebas realizadas?

# Las pruebas realizadas se muestran en la guía técnica.

# 3. ¿Cuál es el objetivo de la guía de instalación?

# El objetivo de esta guía es dar toda la información necesaria para garantizar que la implantación de la aplicación se realice de forma correcta y segura.

# 4. ¿En qué documento quedan reflejados ejemplos de uso del programa?

# Los ejemplos de uso quedan reflejados en la guía de uso.

# 5. ¿Cuál es el objetivo de la guía técnica?

# EL objetivo de la guía técnica es facilitar el correcto desarrollo y realización de las correcciones en los programas, permitiendo un mantenimiento futuro.

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

# Las soluciones a los posibles problemas a futuro se reflejan en la guía de uso.

# 7. ¿En qué documento queda reflejada información sobre la seguridad del sistema?

# Esta información se encuentra en la guía de instalación.

# 8. ¿A quién va dirigida la guía de instalación?

# La guía de instalación va dirigida al personal informático responsable de la instalación junto con los clientes.

# 9. ¿En qué documento queda reflejado la codificación de los programas?

# La codificación se muestra en la guía técnica.

# 10. ¿A quién va dirigida la guía técnica?

# Esta guía está dirigida al personal técnico en informática.

# 11.¿En qué documento queda reflejada la descripción de la funcionalidad de la aplicación?

# La descripción de la funcionalidad se encuentra dentro de la guía de uso.

# 12. ¿En qué documento queda reflejado el diseño de la aplicación?

# El diseño de la aplicación queda reflejado en la guía técnica.

# 13. ¿Qué fases de un proyecto es necesario documentar?

# Es preciso documentar todas las etapas de desarrollo del software para dar toda la información a los usuarios del software y acometer futuras revisiones.

# 14. ¿En qué documento quedan reflejados requisitos software de la aplicación?

# La guía de uso establece los requerimientos software de la aplicación.

# 15. ¿A quién va dirigida la guía de uso?

# La guía de uso va dirigida para los clientes, los usuarios que van a usar la aplicación.

# 16. ¿Cuál es el objetivo de la guía de uso?

# Su objetivo es el de dar a los clientes toda la información necesaria acerca del funcionamiento de la aplicación.

# 17. ¿En qué documento queda reflejada la forma de comenzar a ejecutar la aplicación?

# La guía de uso explica la forma de comenzar la ejecución de la aplicación.

## 6. Explotación

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

# Esta fase consiste en el conocimiento de la aplicación y su utilización inicial por parte de los usuarios finales.

# 2. ¿En qué consisten las betas test?

# Estos test son las últimas pruebas que se realizan en los equipos de los clientes.

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

# Sí, es precioso la fase de configuración, en la cual asignamos los parámetros de funcionamiento normal de la empresa y se prueba que la aplicación es operativa.

# 4. ¿Quién se encarga de realizar las configuraciones en la fase de configuración?

# Esta fase se le puede otorgar a los propios clientes, siempre y cuando se les haya entregado la guía de instalación previamente.

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

# En este caso, la configuración debería realizarla el personal que lo haya fabricado.

# 

## 7. 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.?

# A estos cambios acorde con las tendencias del mercado son los cambios adaptivos.

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

# Estos cambios que satisfacen las necesidades de los clientes son los cambios evolutivos.

# 3. La duración de la etapa de mantenimiento es corta. ¿Verdadero o falso?

# Falso, la etapa de mantenimiento es la más larga de todo el ciclo, ya que el software por su naturaleza, precisa de actualizaciones con el tiempo.

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

# Los cambios que se realizan en el software para la mejora de su funcionamiento son los cambios perfectivos.

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

# Los cambios correctivos son los encargados de solventar los errores en el futuro.

# 6. Lenguajes de programación

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

Los tipos de lenguaje de programación que se han visto a través del tiempo, son los siguientes:

* Lenguaje de máquina
* Lenguaje ensamblador
* Lenguaje de alto nivel
* Lenguaje visual

1. Define lenguaje de programación

Un lenguaje de programación es un idioma creado de forma artificial, formado por un conjunto de símbolos y normas que se aplican sobre un alfabeto para obtener un código, que el hardware puede entender y ejecutar. En conclusión, es el medio de comunicación entre hardware-ordenador.

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

El lenguaje ensamblador utiliza estas instrucciones complejas.

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

El lenguaje de máquina es uno único para cada procesador, es decir, no es portable.

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

El lenguaje de alto nivel basados en código emplea sentencias y ordenes derivadas del inglés en lugar de los mnemotécnicos.

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

Los lenguajes visuales, en lugar de sentencias escritas, programa gráficamente con el ratón. Diseña de esta forma la apariencia del software.

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

El lenguaje de máquina es el único idioma que entiende perfectamente el ordenador.

# Concepto y características

## Mapa conceptual

## 

## Preguntas y dudas

## 

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

Los dos lenguajes de bajo nivel son:

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

Según la forma de ejecutarse, el lenguaje LISP forma parte de los lenguajes interpretados, o ejecución de las instrucciones de forma directa por parte del programa.

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

Los Lenguajes de Programación de Bajo Nivel, según su nivel de abstracción, son los que están más cercanos al funcionamiento interno de la computadora.

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

Los Lenguajes Compiladores son los que emplean esta herramienta para los unir los ficheros de código objeto que crearon previamente al traducir el programa al código máquina.

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

La norma de construcción que describe el enunciado, corresponde a la sintaxis.

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

A los lenguajes con un objetivo muy concreto se les denomina Lenguajes de Propósito Específico.

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

Los lenguajes de Programación de Sistemas se encargan de diseñar los sistemas operativos.

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

El lenguaje de programación está formado conjuntamente por estos 3 elementos: Alfabeto, Sintaxis y Semántica.

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

**es interpretado?**

Los lenguajes mixtos como Java, primero pasan por la compilación y posteriormente, por la interpretación.

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

Los lenguajes imperativos indican como hacer la tarea.

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

A este conjunto se le denomina como Alfabeto.

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

**programa de manera directa?**

Los lenguajes interpretados ejecutan las instrucciones de manera directa.

1. **¿El lenguaje LISP que tipo de lenguaje es según la clasificación del paradigma de**

**programación?**

Según el paradigma de programación, son del tipo declarativo.

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

A estos lenguajes se les denomina Lenguajes de Programación de Alto Nivel.

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

El lenguaje C entra dentro de 2 categorías dentro de esta clasificación; se puede identificar como lenguaje imperativo y de programación.

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

El Csound corresponde a los lenguajes de propósito específico, ya que su fin es el de crear ficheros de audio.

1. **¿Cómo se denomina al significado de las construcciones para hacer acciones válidas en un**

**lenguaje de programación?**

La semántica es el significado de las construcciones para hacer acciones válidas.

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

El lenguaje de propósito general es apto para todo tipo de tareas.

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

El lenguaje C corresponde a la clase de propósito general.

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

Los lenguajes de Script se encargan realizar tareas de control y auxiliares.

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

Los lenguajes compilados se encargan de la traducción descrita.

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

Los lenguajes declarativos indican simplemente la forma de ejecución de la tare exclusivamente, no determina los pasos a seguir.

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

A partir de esta clasificación, este lenguaje entraría dentro de los lenguajes de Programación Orientados a Objetos.

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

El lenguaje C entraría dentro del grupo de los lenguajes compilados.

# Lenguajes de Programación estructurados

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

Las sentencias permitidas para la programación estructurada son:

* Sentencias secuenciales
* Sentencias selectivas (condicionales)
* Sentencias repetitivas (iteraciones o bucles)

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

Las ventajas de la programación estructurada son las siguientes:

* Programas fácilmente legibles
* Mantenimiento sencillo
* Estructuración sencilla

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

Sus desventajas principales son:

* Concentración en único bloque de todo el programa
* Inexistencia de reutilización eficiente del código

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

Esta programación evoluciono a la Programación Modular (división del programa en trozos de código con una función concreta que pueden ser reutilizables)

# Lenguajes de programación orientados a objetos

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

El lenguaje Pascal es un lenguaje estructurado.

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

Estos programas se componen de objetos independientes que colaboran entre ellos para realizar acciones. Estos objetos son reutilizables.

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

El lenguaje JAVA es un lenguaje orientado a objetos.

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

Sus principales ventajas son:

* Su código es reutilizable
* Si existe algún error, su localización y depuración es sencilla

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

El lenguaje C es un lenguaje estructurado.

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

Los atributos son los elementos que nos permiten diferenciar los objetos entre ellos.

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

A esa colección se le denomina “Clase”

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

Mediante los “Métodos”, los objetos se comunican entre sí, produciendo un cambio en su estado.

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

El lenguaje C++ es un lenguaje orientado a objetos.

# 7. El lenguaje C++ es un lenguaje orientado a objetos.¿Qué es un IDE?

IDE o Entornos de Desarrollo Integrado son los entornos de programación que permiten llevar a cabo la codificación y prueba de los programas.

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

Un mismo IDE puede funcionar puede funcionar con varios lenguajes de programación.

1. **¿Qué es un framework?**

El framework es una estructura de ayuda al programador, con el cual podemos desarrollar proyectos sin partir de 0.

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

Entre sus ventajas, podemos encontrar:

* Desarrollo rápido del software
* Reutilización de partes del código para otras aplicaciones
* Diseño único para el software
* Portabilidad de las aplicaciones

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

Las desventajas de un framework son las siguientes:

* Gran dependencia del código respecto al framework utilizado
* Consumición de bastantes recursos del sistema.

1. **Pon dos ejemplos de frameworks**

Dos ejemplos de esta herramienta con: .NET y Spring de Java.