**Tema 1**

[1 Software y programa. Tipos de software 3](#_Toc32431436)

[2 Relación hardware-software 3](#_Toc624722114)

[3 Licencias de Software 4](#_Toc2129062535)

[4 Ciclo de vida del software 5](#_Toc380039751)

[4.1 Modelos de ciclo de vida 6](#_Toc1591642713)

[5 Fases en el desarrollo y ejecución del software 7](#_Toc890243894)

[5.1 Análisis 8](#_Toc257235367)

[5.2 Diseño 10](#_Toc2013725377)

[5.2.1 Diseño estructurado 10](#_Toc1731155191)

[5.2.2 Diseño orientado a objetos 11](#_Toc1159279927)

[5.3 Codificación 11](#_Toc473525519)

[5.3.1 Fases en la obtención del código 11](#_Toc905609701)

[5.3.2 Máquinas virtuales 12](#_Toc868168226)

[5.3.3 Sdg 13](#_Toc1985083117)

[5.3.4 Entornos de ejecución 13](#_Toc918997427)

[5.3.5 Java Runtime Enviroment 13](#_Toc1808655007)

[5.4 Pruebas 14](#_Toc890818556)

[5.5 Documentación 14](#_Toc1397728952)

[5.6 Explotación 15](#_Toc172726155)

[5.7 Mantenimiento 16](#_Toc1334351940)

[6 Lenguajes de programación 16](#_Toc453648772)

[6.1 Concepto y características 17](#_Toc221334521)

[6.2 Lenguajes de programación estructurados 19](#_Toc1492701438)

[6.3 Lenguajes de programación orientados a objetos 20](#_Toc1024618768)

[7. Herramientas de apoyo al desarrollo del software 20](#_Toc673681252)

[8 Diagramas de flujo 22](#_Toc1185687209)

[8.1 Ejercicio diagrama flujo final apuntes 22](#_Toc1383238069)

[8.2 Ejercicio diagrama flujo final apuntes 24](#_Toc447275434)

[8.3 Ejercicio diagrama flujo final apuntes 24](#_Toc842885616)

[8.4 Ejercicio diagrama flujo final apuntes 26](#_Toc1948201552)

[8.5 Ejercicio diagrama flujo final apuntes 27](#_Toc346875452)

[8.6 Ejercicio diagrama flujo final apuntes 28](#_Toc728604545)

[8.7 Ejercicio diagrama flujo final apuntes 29](#_Toc1265475778)

[8.8 Ejercicio diagrama flujo final apuntes 30](#_Toc799308001)

# 1 Software y programa. Tipos de software

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

Aplicaciones informáticas.

2. ¿Qué tipo de software es un videojuego?

De aplicaciones.

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

Software de programación.

4. ¿Qué tipo de software es un depurador?

De programación.

5. ¿Qué tipo de software es un compilador?

De programación.

6. ¿De qué dos partes consta un ordenador?

Hardware y software.

7. ¿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.

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

De sistemas.

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

De aplicaciones.

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

De sistemas.

# 2 Relación hardware-software

1. ¿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.

2. Define hardware

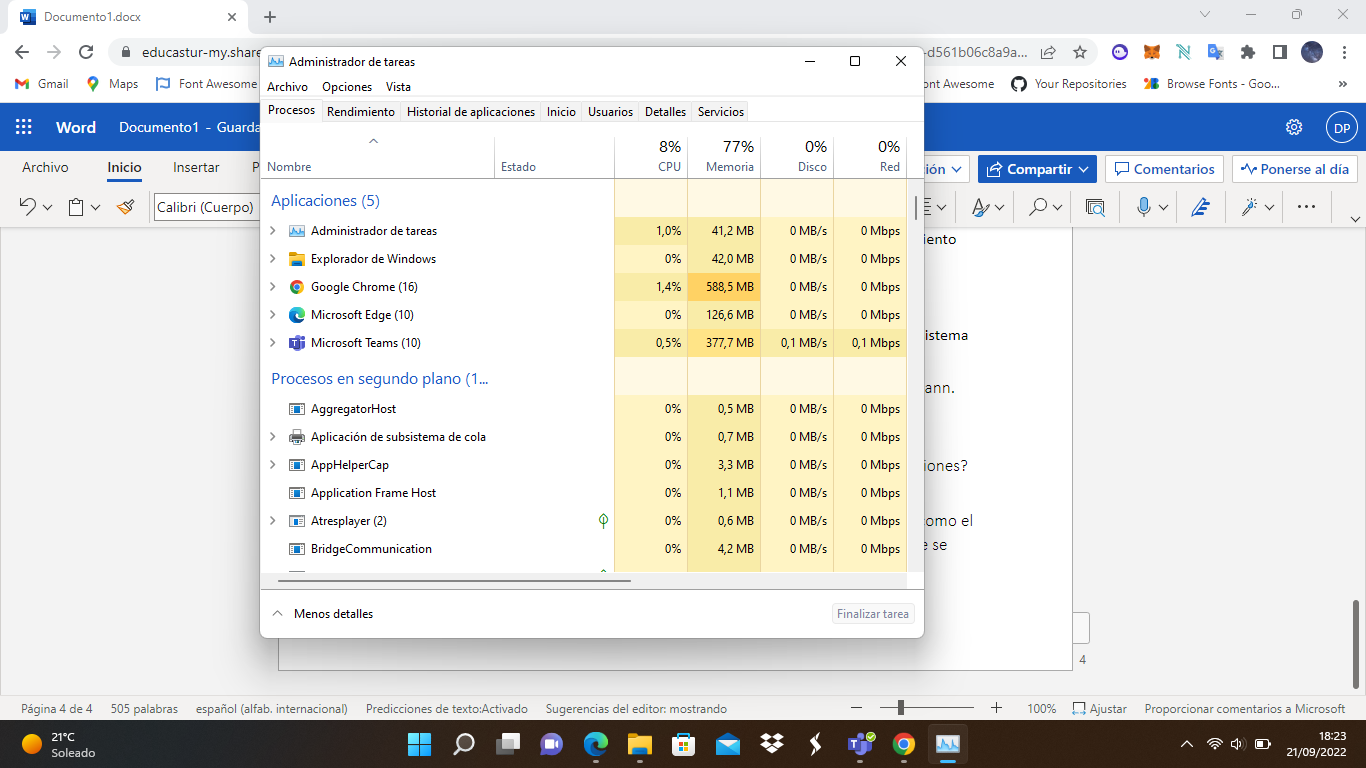
Conjunto de elementos físicos o materiales que constituyen una computadora o un sistema informático.

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

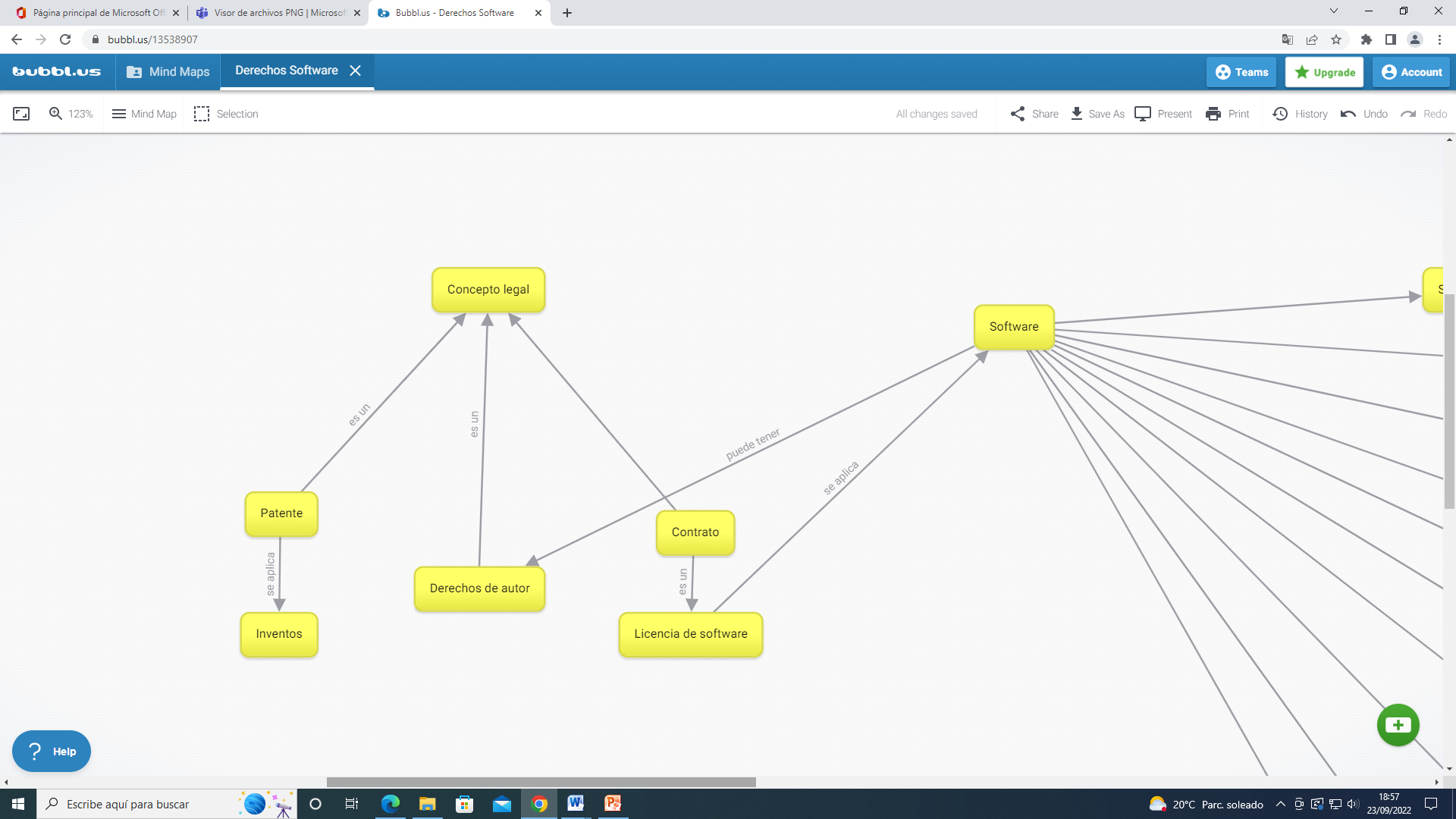
4. ¿Cuál es la relación hardware-software desde el punto de vista de las aplicaciones?

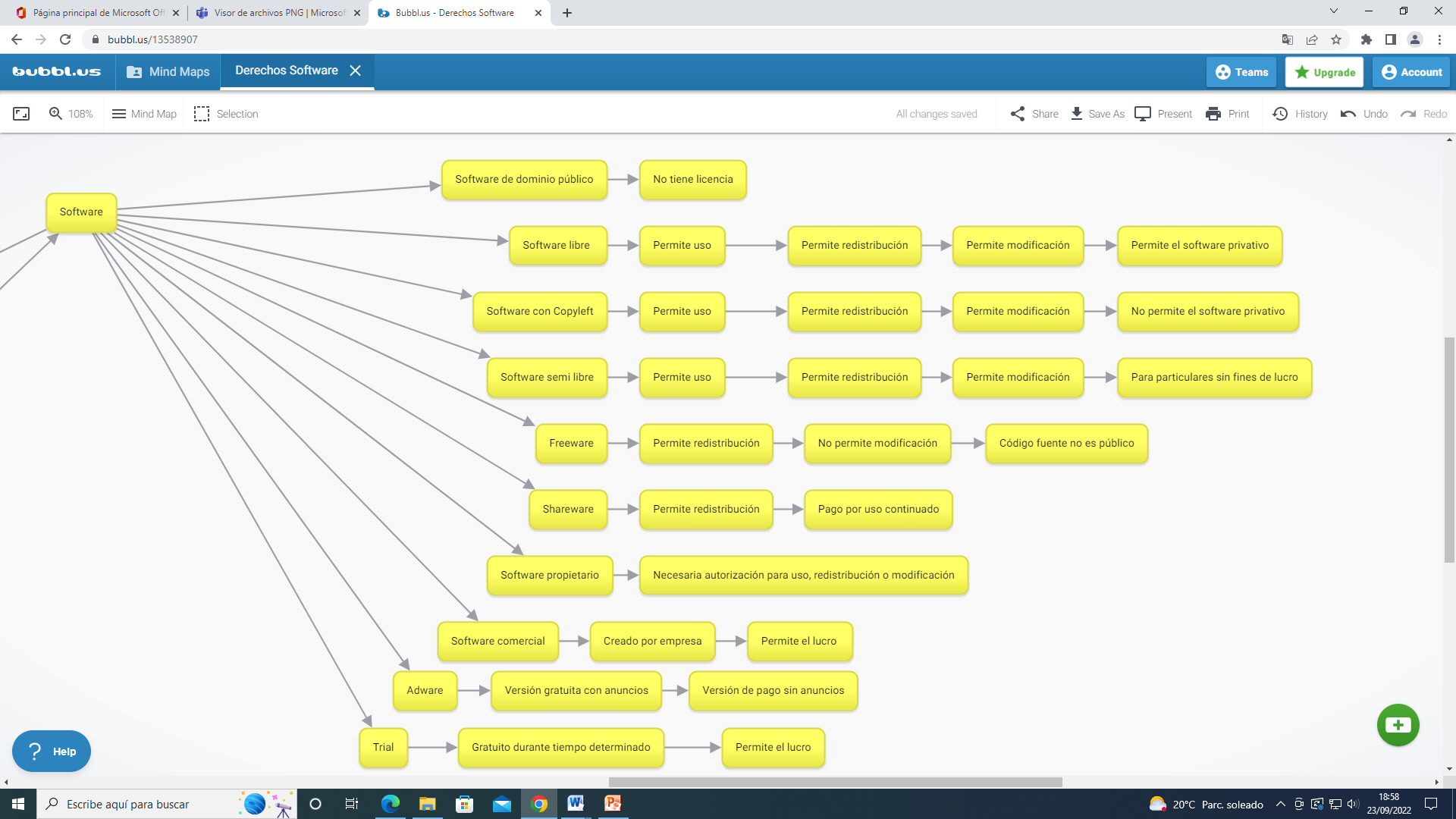
El hardware debe de interpretar y ejecutar las instrucciones de los programas.

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).



# 3 Licencias de Software





# 4 Ciclo de vida del software

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

Cada etapa produce una documentación que se pasa a la siguiente etapa y de la anterior reciben también documentación.

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

Es un periodo que comprende desde el momento en que tenemos la idea hasta que el producto es retirado.

1. ¿Qué entendemos por desarrollo del software?

Es un periodo que comprende desde el momento en que tenemos la idea hasta que el programa esta implementado en el ordenador y funcionando.

## 4.1 Modelos de ciclo de vida

1. Investiga en internet cuál de los dos modelos evolutivos realiza un análisis de riesgos en cada evolución.
2. ¿Cuál es la fase del ciclo de vida que conlleva más esfuerzo del total del proyecto?

Las 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 adaptación, ya que sería solo instalarlo.

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

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 en cascada.

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

El modelo en cascada.

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

Modelo iterativo incremental.

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

Los modelos evolutivos.

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

Modelo en cascada.

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

Que en el de cascada con retroalimentación se puede volver para atrás y en el de cascada no.

# 5 Fases en el desarrollo y ejecución del software

## 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?
2. ¿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?

Especificación de requisitos de software.

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. Qué funciones tendrá que realizar la aplicación. Qué respuesta dará la aplicación ante todas las entradas. Cómo se comportará la aplicación en situaciones inesperadas.

1. ¿Qué es un requisito no funcional?

Restricciones que afectaran al sistema. Tiempos de respuesta del programa, legislación aplicable, tratamiento ante la simultaneidad de peticiones, etc.

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

Requisito funcional.

1. Enumera las fases del desarrollo del software

1-Análisis de requisitos 2-Diseño 3-Codificación 4-Pruebas 5-Documentación 6-Explotación 7-Mantenimiento.

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

Funcional.

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

Las entidades externas representan componentes que no forman parte del sistema. Que proporcionan datos o los reciben. (Rectángulos)

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

Diagramas de entidad.

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

El movimiento de datos del sistema.

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

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?

Diagramas de flujo de datos DFD.

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

Dirección de datos DD.

1. ¿Qué identifican los procesos en un DFD?

Funciones dentro del sistema.

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

Casos de uso.

1. ¿Un requisito que pide que la aplicación se pueda usar en dispositivos móviles de que tipo es?
2. ¿En qué técnica se generan ideas desde diferentes puntos de vista?

Brainstorming.

1. ¿Qué representan los almacenes en un DFD?

Los datos desde el punto de vista estático, lugar donde se almacenan o desde donde se recuperan. (Dos líneas horizontales paralelas)

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, Un modelo de dominio.

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

La planificación de las reuniones que van a tener lugar.

Relación de los objetivos del usuario cliente y del sistema.

Relación de los requisitos funcionales y no funcionales del sistema.

Relación de objetivos prioritarios y temporización.

Reconocimiento de requisitos mal planteados o que conllevan contradicciones, etc.

## 5.2 Diseño

1. ¿Qué se realiza en esta fase?

Se traducen los requisitos funcionales y no funcionales en una representación de software.

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

El diseño estructurado y el diseño orientado a objetos.

### 5.2.1 Diseño estructurado

1. ¿Qué cuatro niveles tiene el diseño clásico?

Diseño de datos, diseño arquitectónico, diseño de interfaz, diseño a nivel de componentes.

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

Diseño de datos.

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

Diseño arquitectónico.

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

Diseño arquitectónico.

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

Diseño de la interfaz.

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

Diseño a nivel de componentes.

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

Diseño a nivel de componentes.

### 5.2.2 Diseño orientado a objetos

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

Hacer un análisis orientado a objetos.

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

Subsistema, clases y objetos, mensajes, responsabilidades.

1. ¿Qué es UML?

Lenguaje de modelado basado en diagramas que sirve para expresar modelos y que se ha convertido en un estándar de las metodologías de desarrollo orientado a objetos.

## 5.3 Codificación

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

Facilita su desarrollo y su mantenimiento en el futuro.

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

Corrección.

1. ¿Qué implica la programación eficiente?

Que haga un buen uso de los recursos.

1. ¿Qué permite un código portable?

Que se pueda implementar en cualquier equipo.

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

Modularidad.

### 5.3.1 Fases en la obtención del código

1. Define código fuente

El código fuente es el conjunto de instrucciones que la computadora deberá realizar.

1. Define algoritmo

Un conjunto de pasos a seguir para obtener la solución del problema.

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

En algún lenguaje de alto nivel y debe ser traducido al lenguaje máquina.

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

1. Se debe partir de las etapas anteriores de análisis y diseño.

2. Se diseñará un algoritmo que simbolice los pasos a seguir para la resolución del problema.

3. Se elegirá una Lenguajes de Programación de alto nivel apropiado para las características del software que se quiere codificar.

4. Se procederá a la codificación del algoritmo antes diseñado.

1. ¿Qué es el código objeto?

El código objeto es un código intermedio. Es el resultado de traducir código fuente a un código equivalente formado por unos y ceros que aún no puede ser ejecutado directamente por la computadora. Es decir, es el código resultante de la compilación del código fuente.

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

Traduciendo el código fuente o compilandolo.

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

La compilación se realiza sobre todo el código y la interpretación línea a línea.

1. ¿Cómo se denomina el proceso de unir todos los archivos de código objeto?
2. ¿Cómo se denomina el software que realiza el proceso de enlazado?

Linker.

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

No.

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

Si.

### 5.3.2 Máquinas virtuales

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

Separar el funcionamiento del ordenador de los componentes hardware instalados.

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

Con el uso de máquinas virtuales podremos desarrollar y ejecutar una aplicación sobre cualquier equipo, independientemente de las características concretas de los componentes físicos instalados.

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 (huésped), para el control de los dispositivos hardware implicados en los procesos.

• Cumplimiento de las normas de seguridad de las aplicaciones.

### 5.3.3 Sdg

### 5.3.4 Entornos de ejecución

1. ¿Qué es un entorno de ejecución?

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

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

Máquina virtual y API.

1. ¿Qué es el runtime?

El tiempo que tarda un programa en ejecutarse en la computadora.

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

• Configurar la memoria principal disponible en el sistema.

• Enlazar los archivos del programa con las bibliotecas existentes y con los subprogramas creados. Considerando que las bibliotecas son el conjunto de subprogramas que sirven para desarrollar o comunicar componentes software pero que ya existen previamente y los subprogramas serán aquellos que hemos creado a propósito para el programa.

• Depurar los programas: comprobar la existencia (o no existencia) de errores semánticos del lenguaje (los sintácticos ya se detectaron en la compilación).

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?

Es un intermediario entre el sistema operativo y el lenguaje de programación.

### 5.3.5 Java Runtime Enviroment

1. ¿Qué es el Java Runtime Environment (JRE)?

Es un conjunto de utilidades permite la ejecución de programas java sobre cualquier tipo de plataforma.

## 5.4 Pruebas

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 de verificación.

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

En las de caja negra.

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

De verificación y validación.

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

En las de validación.

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

Las de caja blanca.

## 5.5 Documentación

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

Guía de instalación.

¿En qué documento quedan reflejadas las pruebas realizadas?

Guía técnica.

¿Cuál es el objetivo de la guía de instalación?

Dar toda la información necesaria para garantizar que la implantación de la aplicación se realice de forma segura, confiable y precisa.

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

Guía de uso.

¿Cuál es el objetivo de la guía técnica?

Facilitar un correcto desarrollo, realizar correcciones en los programas y permitir un mantenimiento futuro.

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

Guía de uso.

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

Guía de instalación.

¿A quién va dirigida la guía de instalación?

Al personal informático responsable de la instalación, en colaboración con los usuarios que van a usar la aplicación (clientes).

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

Guía técnica.

¿A quién va dirigida la guía técnica?

Al personal técnico en informática (analistas y programadores).

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

Guía de uso.

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

Guía técnica.

¿Qué fases de un proyecto es necesario documentar?

Todas.

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

Guía de uso.

¿A quién va dirigida la guía de uso?

A los usuarios que van a usar la aplicación (clientes).

¿Cuál es el objetivo de la guía de uso?

Dar a los usuarios finales toda la información necesaria para utilizar la aplicación.

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

Guía de uso.

## 5.6 Explotación

¿En qué consiste la fase de explotación?

La explotación es la fase en que los usuarios finales conocen la aplicación y comienzan a utilizarla.

¿En qué consisten las beta test?

Son pruebas que se realizan en los propios equipos del cliente y bajo cargas normales de trabajo.

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

En este punto se deberá proporcionar soporte al usuario cuando la solicite.

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

Puede hacerla el cliente, puede hacerse por sí misma o el equipo de desarrollo.

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

Aquellos que la han fabricado.

## 5.7 Mantenimiento

¿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.?

Adaptativos.

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

Evolutivos.

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

Falso.

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

Perfectivos.

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

Correctivos.

# 6 Lenguajes de programación

Enumera los 4 tipos de lenguajes de programación en orden evolutivo

Lenguaje máquina, lenguaje ensamblador, lenguaje de alto nivel, lenguaje visual.

Define 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 de la computadora pueda entender y ejecutar

¿Qué tipo de lenguaje de programación utiliza nemotécnicos?

El ensamblador.

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

El máquina.

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

El de alto nivel.

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

El lenguaje visual.

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

El máquina.

## 6.1 Concepto y características

¿Cuáles son los dos tipos de lenguajes de bajo nivel?

El lenguaje máquina y el lenguaje ensamblador.

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

Lenguaje interpretado.

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

Lenguajes de bajo nivel.

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

Lenguajes compilados.

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

Sintaxis.

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

Lenguajes de propósito específico.

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

Lenguajes de programación de sistemas.

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

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

Lenguaje mixto.

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

Lenguajes imperativos.

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

Alfabeto.

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

Lenguajes interpretados.

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

Lenguaje interpretado.

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

Lenguajes de alto nivel.

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

Lenguaje imperativo.

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

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

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

Lenguajes de propósito general.

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

Lenguajes de programación de sistemas.

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

Lenguajes de script.

¿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.

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

Lenguajes imperativos.

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

Lenguaje de programación orientada a objetos.

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

Lenguaje compilado.

## 6.2 Lenguajes de programación estructurados

¿Qué tres tipos de sentencia permite la programación estructurada?

• Sentencias secuenciales.

• Sentencias selectivas (condicionales).

• Sentencias repetitivas (iteraciones o bucles).

¿Cuáles son las ventajas de la programación estructurada?

• Los programas son fáciles de leer, sencillos y rápidos.

• El mantenimiento de los programas es sencillo.

• La estructura del programa es sencilla y clara.

¿Cuáles son las desventajas de la programación estructurada?

• Todo el programa se concentra en un único bloque (si se hace demasiado grande es difícil manejarlo).

• No permite reutilización eficaz de código, ya que todo va "en uno". Es por esto que a la programación estructurada le sustituyó la programación modular, donde los programas se codifican por módulos y bloques, permitiendo mayor funcionalidad.

¿Hacía que evolucionó la programación estructurada?

Hacia la programación modular.

## 6.3 Lenguajes de programación orientados a objetos

¿El lenguaje Pascal es estructurado u orientado a objetos?

Estructurado.

¿De qué están compuestos los programas orientados a objetos?

Se componen de objetos independientes entre sí que colaboran para realizar acciones.

¿El lenguaje Java es estructurado u orientado a objetos?

Orientado a objetos.

¿Qué ventajas presenta la programación orientada a objetos?

• El código es reutilizable.

• Si hay algún error, es más fácil de localizar y depurar en un objeto que en un programa entero.

¿El lenguaje C es estructurado u orientado a objetos?

Estructurado.

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

Mediante los atributos.

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

Clase.

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

Con los métodos.

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

Orientado a objetos.

# 7. Herramientas de apoyo al desarrollo del software

¿Qué es un IDE?

Son entornos de programación.

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

Puede funcionar con varios lenguajes.

¿Qué es un framework?

Un framework es una estructura de ayuda al programador, en base a la cual podemos desarrollar proyectos sin partir desde cero.

¿Qué ventajas presenta el uso de un framework?

Desarrollo rápido de software.

Reutilización de partes de código para otras aplicaciones.

Diseño uniforme del software.

Portabilidad de aplicaciones de un computador a otro, ya que los bytecodes que se generan a partir del lenguaje fuente podrán ser ejecutados sobre cualquier máquina virtual.

¿Qué desventajas presenta el uso de un framework?

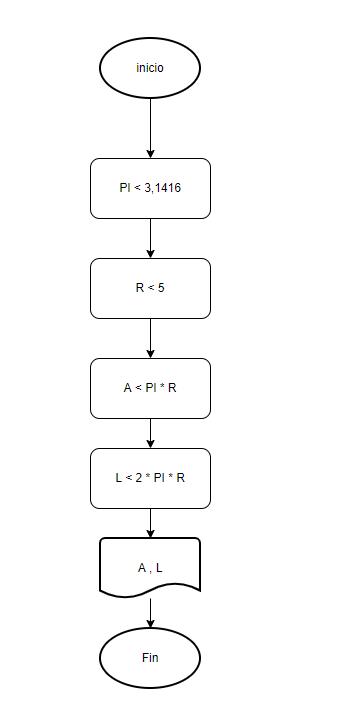
Gran dependencia del código respecto al framework utilizado (sin cambiamos de framework, habrá que reescribir gran parte de la aplicación).

La instalación e implementación del framework en nuestro equipo consume bastantes recursos del sistema.

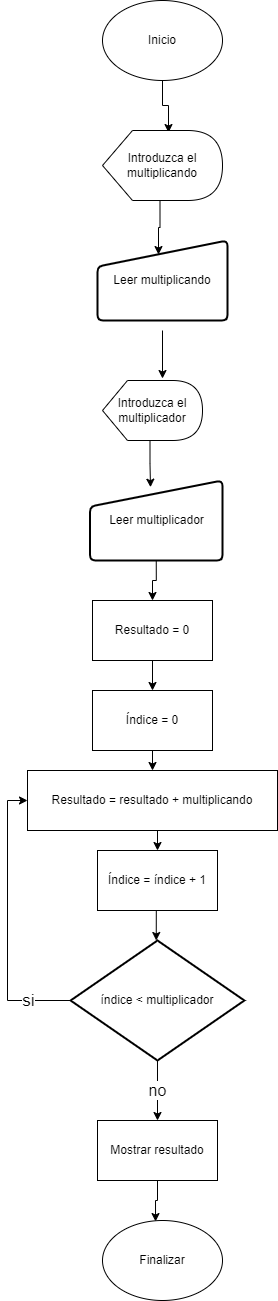
Pon dos ejemplos de frameworks

.NET, Spring de Java.

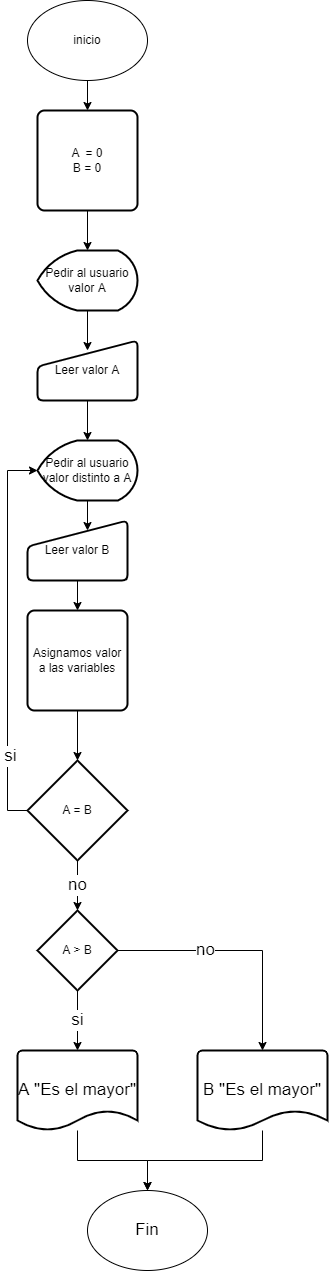
# 8 Diagramas de flujo



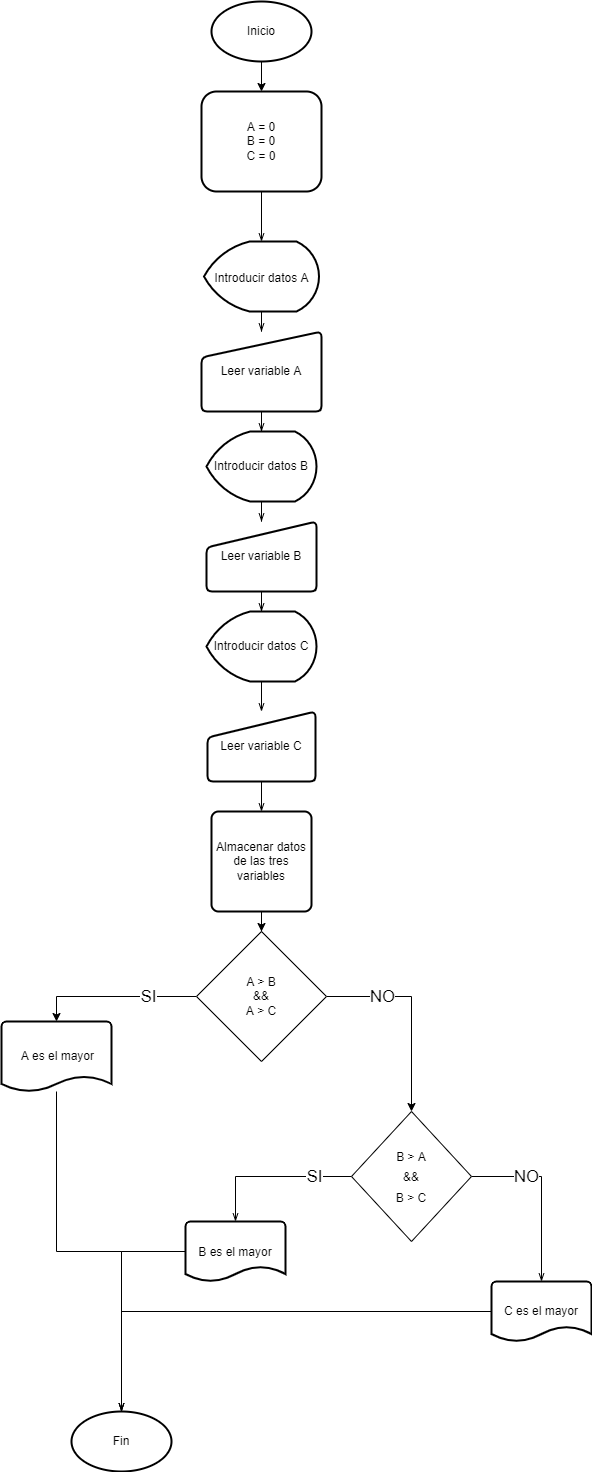
## 8.1 Ejercicio diagrama flujo final apuntes



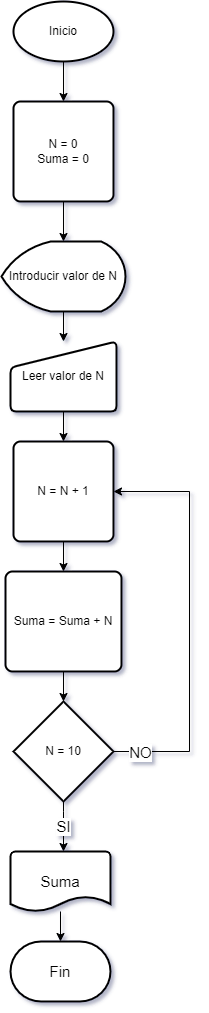
## 8.2 Ejercicio diagrama flujo final apuntes



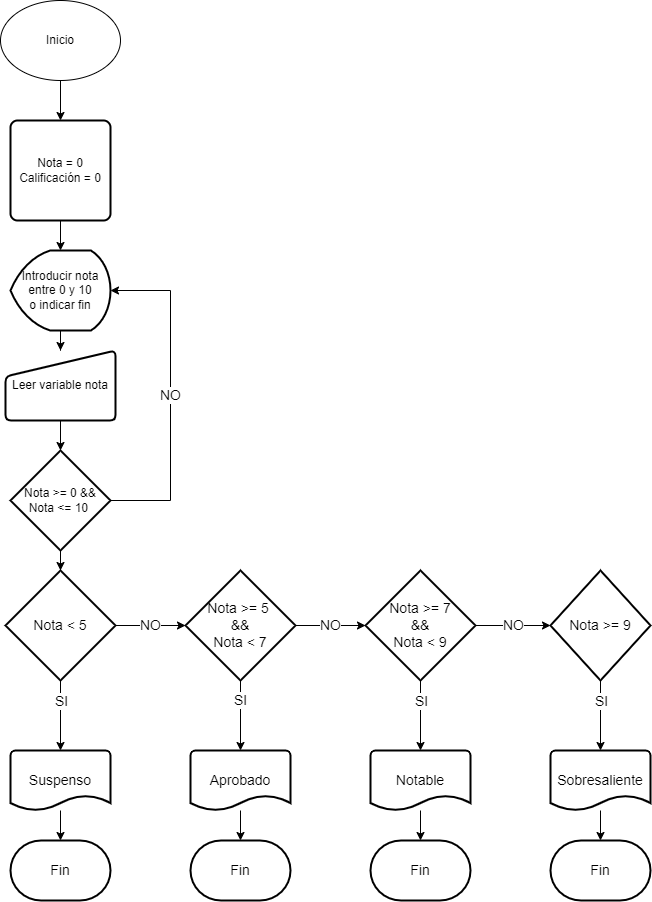
## 8.3 Ejercicio diagrama flujo final apuntes



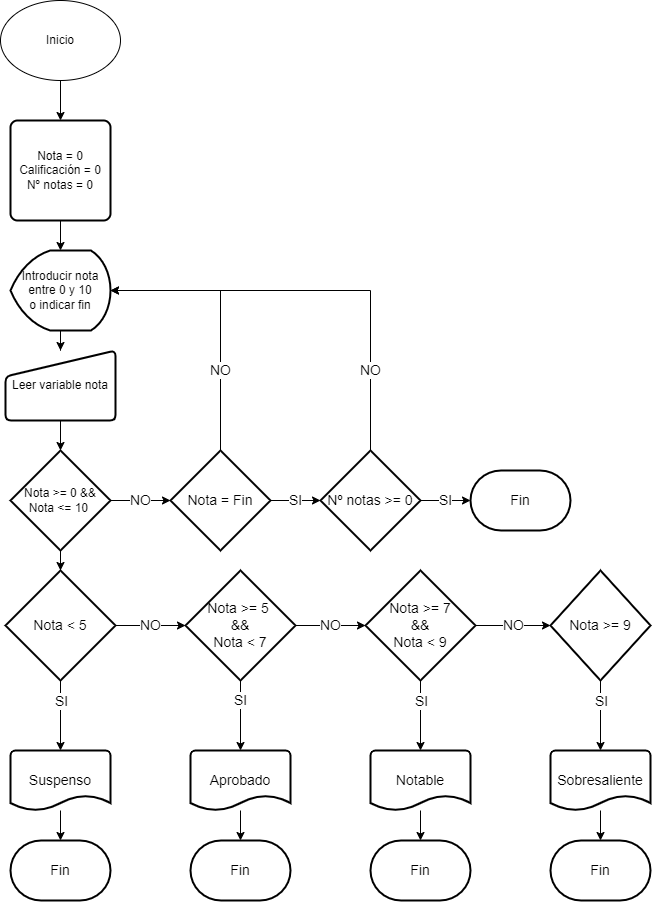
## 8.4 Ejercicio diagrama flujo final apuntes



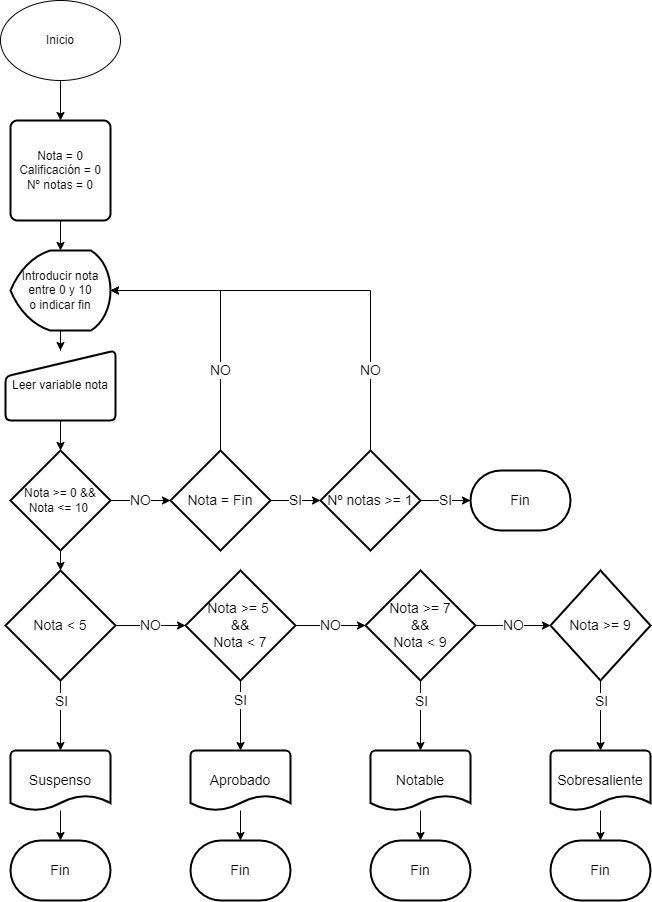
## 8.5 Ejercicio diagrama flujo final apuntes



## 8.6 Ejercicio diagrama flujo final apuntes



## 8.7 Ejercicio diagrama flujo final apuntes



## 8.8 Ejercicio diagrama flujo final apuntes

