ENTORNOS DE DESARROLLO

UNIDAD 1

TAREA ED01

DAVID MEDINA GARCIA

1) (1 p) Explica en qué aspectos está **relacionada una aplicación informática con el hardware**, y pon algún **ejemplo** de cómo **optimizar** el dicho software a la hora de programar.

El hardware y el software son las dos partes interdependientes de un sistema informático completo y su relación se refleja principalmente en los siguientes aspectos.

- La interdependencia del hardware y el software. El hardware es la base material sobre la que funciona el software, y el funcionamiento normal del software es la única forma de que funcione. El sistema informático debe estar equipado con un sistema de software completo para funcionar con normalidad y dar pleno rendimiento a las diversas funciones de su hardware.

- No existe un límite estricto entre el hardware y el software. Con el desarrollo de la tecnología informática, en muchos casos, ciertas funciones de la computadora se pueden realizar mediante hardware o software. Por lo tanto, el hardware y el software no tienen una interfaz absolutamente estricta en cierto sentido.

- Desarrollo coordinado de hardware y software. El software informático se desarrolla con el rápido desarrollo de la tecnología del hardware, y el desarrollo y la mejora continuos del software promueve la actualización del hardware. Los dos están estrechamente entrelazados y desarrollados, y ninguno es indispensable.

***Corrección de los siguientes apartados:**
a/b/c están mal, sustituir por lo siguiente.

Procesador: código eficiente, sin bucles innecesarios;
Memoria: elección oportuna del tipo de variables; cuidar la recursividad excesiva;
Disco Duro: Elección oportuna del tipo de datos a la hora del guardado en la BBDD, controlar accesos a disco.

a. Optimizar el uso del procesador.

Hay varios métodos que ayudaran a aumentar el rendimiento de tu unidad de memoria. A continuación se citan algunos ejemplos:

- Desactiva la conexión P2P de las actualizaciones.
- Liberar espacio del disco duro.
- Desactiva las notificaciones que no necesites.
- Quita programas del inicio del sistema.
- Desactiva animaciones y otros elementos gráficos.
- Quita los plugins del navegador.
- Riesgos.

b. Optimizar el uso de memoria.

Hay varios métodos que ayudaran a aumentar el rendimiento de tu unidad de memoria. A continuación se citan algunos ejemplos:

- Cierra las aplicaciones que más **RAM** consumen.
- Deshabilita aplicaciones que se inician con el PC.
- Limpia tu navegador.
- Cierra las webs que más están consumiendo.
- Elimina las aplicaciones que no utilices.

c. Optimizar el uso de disco duro (almacenamiento y número de accesos)

Hay varios métodos que ayudaran a aumentar el rendimiento de tu disco duro. A continuación se citan algunos ejemplos:

- Escanea y limpia tu disco duro con regularidad.
- Desfragmenta tu disco duro de vez en cuando.
- Vuelve a instalar tu sistema operativo Windows cada pocos meses.
- Desactiva la función de hibernación.
- Convierte tus discos duros a NTFS desde FAT32.

2) (0,5 p) Pon un **ejemplo** de software a **medida** y otro de software **estándar**.

Un Software **a medida** consiste en la creación de un soporte único según las especificaciones del cliente. Es decir, se diseña teniendo en cuenta las necesidades de un negocio en específico. Ejemplos de ello son las aplicaciones que tienen un negocio concreto (Restaurantes, ópticas, peluquerías, etc)

El software **estándar** es un software genérico que puede resolver múltiples necesidades de una empresa. Buenos ejemplos de ello es el paquete office (Word, Excel, etc) o los propios navegadores como Chrome o Firefox.

3) (0,5 p) Busca una **imagen o pantallazo** de un programa sencillo en lenguaje máquina, otro en ensamblador (indicando para qué procesador es) y otro de alto nivel. Así verás la diferencia entre ellos. (Por ejemplo, el típico HOLA MUNDO en cada uno de ellos)

- **Lenguaje maquina**

  Se trata de un lenguaje de bajo nivel compuesto por cadenas de bits y que es entendido por el procesador. Reúne las instrucciones que recibe una máquina a la hora de llevar los procesos para los que haya sido programada.

```
b8    21 0a 00 00   #moving "!\n" into eax
a3    0c 10 00 06   #moving eax into first memory location
b8    6f 72 6c 64   #moving "orld" into eax
a3    08 10 00 06   #moving eax into next memory location
b8    6f 2c 20 57   #moving "o, W" into eax
a3    04 10 00 06   #moving eax into next memory location
b8    48 65 6c 6c   #moving "Hell" into eax
a3    00 10 00 06   #moving eax into next memory location
b9    00 10 00 06   #moving pointer to start of memory location into ecx
ba    10 00 00 00   #moving string size into edx
bb    01 00 00 00   #moving "stdout" number to ebx
b8    04 00 00 00   #moving "print out" syscall number to eax
cd    80            #calling the linux kernel to execute our print to stdout
b8    01 00 00 00   #moving "sys_exit" call number to eax
cd    80            #executing it via linux sys_call
```

- **Lenguaje ensamblador**

  El lenguaje ensamblador trabaja con nemónicos, que son grupos de caracteres alfanuméricos que simbolizan las órdenes o tareas a realizar. La traducción de los nemónicos a código máquina entendible por el microcontrolador la lleva a cabo un programa ensamblador.

Ejemplo para el procesador INTEL 8086

```
section .text
   global _start

_start:
   mov  edx,len
   mov  ecx,msg
   mov  ebx,1
   mov  eax,4
   int  0x80

   mov  eax,1
   int  0x80

section .data
msg db 'Hola, mundo!', 0xa
len equ $ - msg
```

- **Lenguaje de alto nivel**

  Es un lenguaje de programación con una fuerte abstracción de los detalles de la computadora. Puede utilizar elementos del lenguaje natural, haciéndolo más fácil de usar o puede automatizar áreas importantes de los sistemas informáticos (por ejemplo, gestión de memoria), lo que hace que el proceso de desarrollo de un programa sea más simple y sencillo.

JAVA

```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Prints the string to the
console.
    }
}
```

4) (1 p) Explica con **tus palabras** las diferentes maneras que hay de **traducción** de programas (compilado, interpretado y virtual), diciendo las principales **características** de cada una de ellas y sus **ventajas** y **desventajas**.

- **Compilado**

  Necesitan de un programa traductor (compilador) para convertir el código fuente en código máquina. Este tipo de programas se ejecutan de forma más rápida que los interpretados o los virtuales. Además del compilador, existe un programa llamado enlazador o linker que permite unir el código objeto del programa con el código objeto de las librerías. Sus principales **ventajas e inconvenientes** son:

  Ventajas

  - Dado que el programa ya se ha traducido, la ejecución es más rápida
  - Las instrucciones son enviadas directamente al procesador
  - Mayor eficiencia de ejecución del programa
  - Puede ejecutarse independientemente del entorno del lenguaje.

  Inconvenientes

  - Los errores de compilación impiden que se compile el código.
  - Un lenguaje compilado genera un file binario no modificable
  - Se requieren dos pasos separados para ejecutar el programa desde el código fuente
  - El programa solo se puede ejecutar en ciertas máquinas y sistemas operativos

- **Interpretado**

  No se genera código objeto. El intérprete es un programa que tiene que estar cargado en memoria y se encarga de leer cada una de las instrucciones, interpretarlas y luego ejecutarlas. Las instrucciones se traducen «on the fly» y solamente se traducen las instrucciones que se van ejecutando en vez de interpretar todo el programa. Sus principales **ventajas e inconvenientes** son:

Ventajas

- La independencia de plataforma en los lenguajes interpretados (se pueden usar en cualquier S.O.).
- La reflexión y uso reflexivo del evaluador
- Generación funcional de primer orden, y orden sin necesidad de especificar metadata
- Posibilidad de generación de código in-situ, sin necesidad de recurrir a una compilación
- Contiene distintos tipos Dinámicos
- Facilidad en la depuración
- Gestión de memoria automática

Inconvenientes

- La ejecución del programa por medio de un intérprete es usualmente mucho menos eficiente que la ejecución de un programa compilado. No es eficiente en tiempo porque, o cada instrucción debe pasar por una interpretación en tiempo de ejecución, o como en más recientes implementaciones, el código tiene que ser compilado a una representación intermedia antes de cada ejecución
- La máquina virtual es una solución parcial al problema de la eficiencia del tiempo pues la definición del lenguaje intermedio es mucha más cercana al lenguaje de máquina y por lo tanto más fácil de ser traducida en tiempo de ejecución
- Es necesario un intérprete en la máquina local para poder hacer la ejecución posible

- **Virtual o Mixto**

Son lenguajes más portables que los lenguajes compilados puesto que el código que se genera tras la compilación es un código intermedio o bytecode. Este código puede ser a su vez interpretado por una máquina virtual instalada en cualquier equipo. Tienen una ejecución lenta pero su versatilidad de poder ejecutarse en cualquier entorno los hace muy apreciados. Sus principales **ventajas e inconvenientes** son:

Ventajas

- Puede ejecutarse en cualquier entorno
- Mejora la eficiencia de producción de software

Inconvenientes

- Tienen una ejecución lenta
- Necesita el intérprete de Bytecode (Java Virtual Machine)

5) (0,5 p) Pon un gráfico o imagen representativo de cada uno de los procesos anteriores.

LENGUAJE COMPILADO

LENGUAJE INTERPRETADO

LENGUAJE VIRTUAL O MIXTO

6) (0,5 p) ¿Qué sabes de la traducción **Just in Time**? Explícala con tus palabras y pon ejemplo de lenguajes que utilicen esa traducción.

El compilador **JIT** (Just-In-Time) es un componente del entorno de ejecución que mejora el rendimiento de aplicaciones Java™ compilando códigos de bytes en código de máquina nativo en tiempo de ejecución. **JIT** solo compilará las funciones que se vayan a utilizar en ese momento, guardando el resultado en una caché. A medida que vamos utilizando el programa, cuando nos encontramos con una nueva función que aún no se ha compilado, esta se compila de nuevo. Pero, cuando nos encontramos con una función que ya se ha utilizado, en lugar de compilarla de nuevo se busca en la caché, ahorrando una importante cantidad de tiempo.

Algunos ejemplos de programas que usan **JIT** son **JAVA**, **C#** o **C++.**

7) (1 p) Elige **3 lenguajes de programación de cada tipo** *compilados*, *interpretados* y *virtuales* (también llamados mixtos o compilados-interpretados), es decir 9 en total. Indica de cada uno de ellos sus principales características y para qué suelen utilizarse.

- **Compilados**

- C++: Es un lenguaje de programación orientado a objetos. Se utiliza para realizar programación estructurada de alto nivel y rendimiento, como sistemas operativos, videojuegos y aplicaciones en la nube.

Características

- Su sintaxis es heredada del lenguaje C.
- Programa orientado a objetos (POO).
- Permite la agrupación de instrucciones.
- Es portátil y tiene un gran número de compiladores en diferentes plataformas y sistemas operativos.
- Permite la separación de un programa en módulos que admiten compilación independiente.
- Es un lenguaje de alto nivel.

- Visual Basic: Su objetivo es diseñar, de forma productiva, aplicaciones de tipos con seguridad y orientadas a objetos para móviles, web y Windows.

Características

- Separación de la creación de la interfaz gráfica y el código.

- Una barra de herramientas, con los controles (se podían añadir muchos o crear de propios) necesarios para la creación de formularios. Sólo había que arrastrar y redimensionarlos dentro de la interfaz. Los controles comunes eran labels, textbox, button, checkbox, picturebox, combobox, frame, timer, etc.

- Un explorador de proyectos, para ver todos los formularios, controles, código, etc de nuestro proyecto y un panel de propiedades, dependiendo del formulario o control seleccionado. De esta manera se podía modificar sus características en modo diseño.

- En la parte inferior una Ventana inmediato, que mientras se depuraba podías escribir directamente funciones simples o el valor contenido de las variables, y nos ayuda a la hora de corregir errores en nuestro código.

- **C:** Se desarrollan tanto aplicaciones como sistemas operativos (Windows, Linux, Mac, Unix) a la vez que forma la base de otros lenguajes más actuales como Java, C++ o C# y está presente en la mayoría de bases de datos.

Características

- Estructura de C - Lenguaje estructurado.

- Programación de nivel medio (beneficiándose de las ventajas de la programación de alto y bajo nivel).

- No depende del hardware, por lo que se puede migrar a otros sistemas.

- Objetivos generales. No es un lenguaje para una tarea específica, pudiendo programar tanto un sistema operativo, una hoja de cálculo o un juego.

- Ofrece un control absoluto de todo lo que sucede en el ordenador.

- Organización del trabajo con total libertad.

- Los programas son producidos de forma rápida y son bastante potentes.

- Rico en tipo de datos, operadores y variables en C.

- **Interpretados**

- PHP: Es un lenguaje de programación destinado a desarrollar aplicaciones para la web y crear páginas web, favoreciendo la conexión entre los servidores y la interfaz de usuario.

Características

- Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- Es considerado un lenguaje fácil de aprender.
- El código fuente escrito en PHP es invisible al navegador web y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando módulos (extensiones).
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos. Incluso aplicaciones como Zend framework, empresa que desarrolla PHP, están totalmente desarrolladas mediante esta metodología.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones (desde PHP5).
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar, aun haciéndolo, el programador puede aplicar en su trabajo cualquier técnica de programación o de desarrollo que le permita escribir código ordenado, estructurado y manejable.

- **Python**: ES muy utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos, inteligencia artificial (IA) y aprendizaje automático.

Características

- **Programación orientada a objetos**: Se organiza en clases y objetos, por lo cual es posible representar conceptos de la vida diaria en el lenguaje. Esto lo hace más eficiente a largo plazo, ya que es posible reutilizar dichos conceptos en cualquier momento.

- **Lenguaje interpretado**: No es necesario complicar el código fuente a código máquina, pues ya se cuenta con un intérprete que ejecutará el programa directo del código.

- **Multiplataforma**: Es posible ejecutar este lenguaje de programación en muchos sistemas operativos (Windows, Linux, Mac, etc.) y dispositivos.

- **Tipado dinámico**: Con Python no es necesario que las variables se tipifiquen, pues estas adoptan un tipo en función del valor asignado.

- **Fuertemente tipado**: Los valores ingresados no cambian de un momento a otro, y en caso de hacerlo, se debe marcar. Esto previene errores a largo plazo, pues los señala desde el inicio.

- **Código abierto**: Python permite que cualquier desarrollador contribuya al código, de esta forma el código base sigue creciendo y se adapta a las necesidades de cada programador.

- **Fácil de aprender**: Este lenguaje de programación es bastante amigable, tanto para los programadores que van iniciando, hasta los que tienen experiencia en otros lenguajes.

- **Soporte para GUI**: Python es compatible con distintas Interfaces Gráficas de Usuario (GUI), así que el código puede tener distintas apariencias, lo cual aumenta la visibilidad.

- **Portabilidad**: Permite que el código pueda trasladarse de un sistema a otro sin hacerle algún cambio.

- JavaScript: Se usa para añadir características interactivas a tu sitio web, (por ejemplo, juegos, eventos que ocurren cuando los botones son presionados o los datos son introducidos en los formularios, efectos de estilo dinámicos, animación, y mucho más).

Características

- **Lenguaje de script centrado en objetos:** Los lenguajes centrados en objetos se utilizan principalmente para características como el polimorfismo, que es una cualidad de tomar un objeto en muchas formas.

- **Tecnología de punta del cliente:** La tecnología edge del cliente en Java Script permite al cliente tener control total sobre el contenido que se actualiza en los servidores.

- **Validación de la entrada del usuario:** permite a los usuarios interactuar con el cliente mediante el llenado de formularios a través de páginas web.

- **Otra declaración y si**: Las declaraciones IF y Else se utilizan para realizar operaciones lógicas.

- **Intérprete centrado:** permite al usuario obtener la salida sin el uso del compilador.

- **Capacidad para realizar la función incorporada:** tiene muchas funciones incorporadas como isNAN (), Number (), parseFloat () y parseInt () etc.

- **Formato sensible a mayúsculas y minúsculas:** Distingue entre mayúsculas y minúsculas

- **Peso ligero y delicado:** los códigos escritos en JavaScript no incluyen variables y solo utilizan objetos para realizar las operaciones.

- **Declaraciones en bucle**: se usa para realizar las mismas operaciones repetidamente.

- **Manejo de eventos:** tiene la capacidad de controlar las operaciones actualizadas en los servidores.

➢ **Mixtos**

- Java: Se utiliza principalmente para desarrollar aplicaciones web. También destaca en las aplicaciones móviles (Android).

Características

- **Es simple:** Java ofrece la funcionalidad de un lenguaje potente, derivado de C y C++, pero sin las características menos usadas y más confusas de estos, haciéndolo más sencillo.

- **Orientado a objetos:** El enfoque orientado a objetos (OO) es uno de los estilos de programación más populares. Permite diseñar el software de forma que los distintos tipos de datos que se usen estén unidos a sus operaciones.

- **Es distribuido:** Java proporciona una gran biblioteca estándar y herramientas para que los programas puedan ser distribuidos.

- **Independiente a la plataforma:** Esto significa que programas escritos en el lenguaje Java pueden ejecutarse en cualquier tipo de hardware, lo que lo hace portable.

- **Recolector de basura:** Cuando no hay referencias localizadas a un objeto, el recolector de basura de Java borra dicho objeto, liberando así la memoria que ocupaba. Esto previene posibles fugas de memoria.

- **Es seguro y sólido:** Proporcionando una plataforma segura para desarrollar y ejecutar aplicaciones que, administra automáticamente la memoria, provee canales de comunicación segura protegiendo la privacidad de los datos y, al tener una sintaxis rigurosa evita que se quiebre el código, es decir, no permite la corrupción del mismo.

- **Es multihilo:** Java logra llevar a cabo varias tareas simultáneamente dentro del mismo programa. Esto permite mejorar el rendimiento y la velocidad de ejecución.

- Elixir: Es un lenguaje de programación dinámico y funcional, está considerado como un lenguaje cuya finalidad es poder desarrollar todo tipo de aplicaciones escalables y mantenibles. Es decir, se dedica a no dejar obsoleto ninguno de los programas o aplicaciones que se desarrolla bajo este lenguaje.

Características

- Está basado en los lenguajes de programación Erlang y Ruby.

- Su sintaxis es fácil de entender.

- Puede utilizarse libremente para cualquier proyecto, independientemente de su propósito.

- Es un lenguaje funcional (sus funciones son puras y los valores inmutables).

- Es un lenguaje distribuido (Beam está pensado para trabajar en redes y poder distribuir el software y la carga a través de los distintos nodos que la forman).

- Es un lenguaje concurrente (no requiere de un orden para su ejecución).

- Es un lenguaje resiliente (gran resistencia a fallos y errores, garantizando un alto nivel de disponibilidad).

- Es un lenguaje rápido al ejecutarse en Beam y aprovechar todos los núcleos de procesamiento que se encuentren disponibles.

- Trabaja con tipado dinámico (no hay que declarar y mantener los tipos de variables para cada operación).

- Scala: Es un lenguaje de programación multi-paradigma diseñado para expresar patrones comunes de programación de forma concisa, elegante y con tipos seguros. Integra sutilmente características de lenguajes funcionales y orientados a objetos.

Caracteristicas

- **Orientación a objetos**: Scala es un lenguaje de programación puro orientado a objetos, en el sentido de que cada valor es un objeto. El tipo y comportamiento de los objetos se describe por medio de clases y traits. La abstracción de clases se realiza extendiendo otras clases y usando un mecanismo de composición basado en mixins como un reemplazo limpio de la herencia múltiple.

- **Lenguaje funcional:** Scala también posee características propias de los lenguajes funcionales. Por ejemplo, las funciones son valores de primera clase, soportando funciones anónimas, orden superior, funciones anidadas y currificación, viene integrado de fábrica con la técnica de pattern matching para modelar tipos algebraicos usados en muchos lenguajes funcionales.

- **Tipificado estático:** Scala está equipado con un sistema de tipos expresivo que refuerza a que las abstracciones de tipos se usen en forma coherente y segura.

- **Extensibilidad:** Scala se diseñó teniendo en mente el hecho de que en la práctica el desarrollo de aplicaciones requiere a menudo de extensiones específicas del lenguaje. Para ello, se proporcionan una combinación única de mecanismos que facilitan agregar construcciones nuevas al lenguaje en forma de bibliotecas.

8) (0,5 p) ¿Cuáles son los **lenguajes de programación más utilizados hoy en día** y para qué se utilizan cada uno de ellos? (al menos 5)

-Python: ES muy utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos, inteligencia artificial (IA) y aprendizaje automático.

-C: Se desarrollan tanto aplicaciones como sistemas operativos (Windows, Linux, Mac, Unix) a la vez que forma la base de otros lenguajes más actuales como Java, C++ o C# y está presente en la mayoría de bases de datos.

-Java: Se utiliza principalmente para desarrollar aplicaciones web. También destaca en las aplicaciones móviles (Android).

-C++: Es un lenguaje de programación orientado a objetos. Se utiliza para realizar programación estructurada de alto nivel y rendimiento, como sistemas operativos, videojuegos y aplicaciones en la nube.

-C#: Se utiliza para crear sitios y aplicaciones web, así como generación de aplicaciones web ASP.NET, Servicios Web XML, aplicaciones de escritorio aplicaciones móviles.

-Visual Basic: Su objetivo es diseñar, de forma productiva, aplicaciones de tipos con seguridad y orientadas a objetos para móviles, web y Windows.

-JavaScript: Se usa para añadir características interactivas a tu sitio web, (por ejemplo, juegos, eventos que ocurren cuando los botones son presionados o los datos son introducidos en los formularios, efectos de estilo dinámicos, animación, y mucho más).

-PHP: Es un lenguaje de programación destinado a desarrollar aplicaciones para la web y crear páginas web, favoreciendo la conexión entre los servidores y la interfaz de usuario.

-SQL: Se usa para controlar todas las funciones que un sistema gestor de base de datos brinda a sus usuarios, proporcionando además un marco para crear la propia base de datos, gestionar su seguridad, actualizar sus contenidos, recuperar los datos y compartirlos entre diferentes usuarios.

9) (0,5 p) Indica la **diferencia** entre código fuente, código intermedio y código ejecutable.

- **Código Fuente:** Este código se encuentra en el programa, es escrito por alguien capaz de conocer su lenguaje, es decir un programador, pero este código no es directamente ejecutable, es más bien traducido a otro lenguaje o código.

- **Código Intermedio u Objeto:** Este código es la recopilación del código fuente, donde puede haber un lenguaje de bytes o de máquinas, que luego se distribuyen en varios archivos. No es un código ejecutable.

- **Código Ejecutable:** Son unidades de programas, donde se realizan las instrucciones, que ya se han compilado, este código se encuentra listo para ser ejecutado en cualquier computadora.

10) (0,5 p) ¿Qué **fases** de **análisis** se realizan durante el proceso de compilación? Explica brevemente cada una de ellas.

-Análisis Léxico:

Se encarga de convertir el código fuente del usuario en una corriente de caracteres que al final se convierten en un conjunto de lexemas (secuencia de caracteres) con significados específicos que denominamos tokens.

-Análisis Sintáctico:

En esta fase, los componentes léxicos se agrupan en frases gramaticales que el compilador utiliza para sintetizar la salida.

-Análisis Semántico:

La fase de análisis semántico se intenta detectar instrucciones que tengan la estructura sintáctica correcta, pero que no tengan significado para la operación implicada.

11) (1 p) ¿Cuáles son las **fases de Desarrollo de software**? Define cada una de ellas en 2 o 3 líneas como máximo.

1. **Análisis:** Se intenta descubrir qué es lo que realmente se necesita y se llega a una comprensión adecuada de los requerimientos del sistema (las características que el sistema debe poseer).

2. **Diseño:** Se estudian posibles opciones de implementación para el software que hay que construir, así como decidir la estructura general del mismo.

3. **Codificación:** Se elige un determinado lenguaje de programación y se codifica toda la información anterior (indicar paso a paso usando un lenguaje de programación, las tareas que debe realizar el ordenador). Se obtiene el código fuente.

4. **Compilación:** En esta fase se hace una traducción de todo el código fuente con el objetivo de pasarlo a lenguaje máquina.

5. **Pruebas:** Se realizan una serie de pruebas en busca de los fallos cometidos en las etapas anteriores para corregirlos.

6. **Explotación:** Es la fase en que los usuarios finales conocen la aplicación y comienzan a utilizarla. Se realiza la instalación, puesta a punto y funcionamiento de la aplicación en el equipo final del cliente.

7. **Mantenimiento:** Es el proceso de control, mejora y optimización del software.

8. **Documentación:** Se hace necesario dar toda la información a los usuarios de nuestro software y para poder acometer futuras revisiones.

12) (1 p) Nombra los principales **perfiles** profesionales que participan en el desarrollo de software, indicando brevemente las funciones de cada uno de ellos.

JEFE DE PROYECTO

Es la persona que gestiona el buen funcionamiento del proyecto, quien controla y administra los recursos (tanto personales como económicos) con el fin de cumplir el plan y el objetivo definido.

ANALISTA DE SOFTWARE

Interviene en las primeras fases del proyecto donde se realizan las especificaciones de las necesidades o la problemática del cliente, desde lo general al detalle.

ARQUITECTO DE SOFTWARE

Tiene como misión crear, durante todo el proceso de desarrollo, la documentación que recoge los requisitos y será él quien centralice las decisiones técnicas sobre los problemas que irán surgiendo, asegurar la calidad, y mejorar continuamente la arquitectura.

DESARROLLADOR DE SOFTWARE

Este perfil conoce y es capaz de realizar todas las tareas de desarrollo, pero se ciñe a la implementación y delega otras funciones (como la de programación, el testeo, la supervisión o el mantenimiento) a otros miembros del equipo.

PROGRAMADOR

Es el encargado de traducir en código la especificación del sistema. A pesar de que el desarrollador también puede "picar código", los programadores se dedican exclusivamente a esto. Esta persona debe conocer los diferentes lenguajes de programación. Y además, se encarga de depurar los errores, implementar nuevas funcionalidades o mantener de forma general las aplicaciones cuando lo necesiten.

TESTER

Se encargará de asegurar que los requisitos definidos por el arquitecto de software se cumplen en la implementación del producto o servicio realizada por los desarrolladores y/o programadores. Para ello, será responsable de aplicar diferentes métodos de testeo junto a los programadores. Informará de todos los errores encontrados durante la fase de pruebas.

QUALITY ASSURANCE (QA)

Un QA se asegura de la calidad del software durante todas sus fases, no sólo en la fase de pruebas como un tester.

13) (0,5 p) Explica con tus palabras las **diferencias que ves entre fase de análisis y de diseño**. Valen ejemplos.

Durante la fase de **análisis** tratamos de determinar las funciones y características que queremos que posea nuestra aplicación (lo que queremos hacer), mientras que en la fase de **diseño** lo que queremos es establecer el cómo se llevan a cabo estas funciones (como lo queremos hacer).

14) (1 p) Explica al menos **2 metodologías** de desarrollo de software **clásicas** y otros **2 ágiles**. De las 4 metodologías que expliques tienes que indicar, principales características, y algún ejemplo de proyecto de software real que se ajustaría bien a ellas, razonando tu respuesta.

Metodologias clásicas

- **Prototipado**: Se basa en la construcción de un prototipo de software que se construye rápidamente para que los usuarios puedan probarlo y aportar feedback. Así, se puede arreglar lo que está mal e incluir otros requerimientos que puedan surgir. Es un modelo iterativo que se basa en el método de prueba y error para comprender las especificidades del producto.

  Características

  - Describe las fases principales de desarrollo de software.
  - Define las fases primarias esperadas de ser ejecutadas durante esas fases.
  - Ayuda a administrar el progreso del desarrollo del software
  - Provee un espacio de trabajo para la definición de un detallado proceso de desarrollo de software.

- **Espiral**: Es una combinación de los dos modelos anteriores, que añade el concepto de análisis de riesgo. Se divide en cuatro etapas: **planificación, análisis de riesgo, desarrollo de prototipo y evaluación del cliente.** El nombre de esta metodología da nombre a su funcionamiento, ya que se van procesando las etapas en forma de espiral. Cuanto más cerca del centro se está, más avanzado está el proyecto.

Características

- Es un modelo que puede combinarse con otros modelos de procesos de desarrollo (cascada y evolutivo).
- Es el mejor modelo que se utiliza para desarrollar grandes sistemas.
- El análisis de riesgo requiere la participación de personal con experiencia.

Metodologias agiles

- **Kanban**: Metodología de trabajo inventada por la empresa de automóviles Toyota. Consiste en dividir las tareas en porciones mínimas y organizarlas en un tablero de trabajo dividido en tareas pendientes, en curso y finalizadas. De esta forma, se crea un flujo de trabajo muy visual basado en tareas prioritarias e incrementando el valor del producto.

Características

- Visualizar el flujo de trabajo
- Limitar el trabajo que está en curso
- Gestionar el flujo de trabajo
- Hacer evidentes las políticas del proceso
- Implementación de ciclos de retroalimentación
- Mejora colaborativa
- Flexibilidad

- **Lean**: Está configurado para que pequeños equipos de desarrollo muy capacitados elaboren cualquier tarea en poco tiempo. Los activos más importantes son las personas y su compromiso, relegando así a un segundo plano el tiempo y los costes. El aprendizaje, las reacciones rápidas y potenciar el equipo son fundamentales.

Características

- Es una filosofía de trabajo.
- Su objetivo es la eliminación de todo tipo de desperdicio.
- Trata de conseguir la máxima eficiencia en todos los procesos.
- Aumenta la competitividad de las empresas.
- Se basa en la aportación de las personas relacionadas.

15) Realiza una **captura de pantalla**, en la que se vea la realización de este trabajo y de fondo todo el escritorio de tu ordenador junto con la identificación en el aula virtual abierta. La captura tiene que ser personal y el trabajo que debe aparecer que sea en el procesador de texto que hayas elegido. Luego todo el documento, con la captura incluida, deberás exportarlo a PDF para subirlo a la plataforma.