DESARROLLO DE SOFTWARE

Unidad Didáctica 1

Contenidos

Bloques:

I. **El software del ordenador**
   A. Clasificación del Software
   B. Software de Programación
II. Fases de desarrollo de una aplicación informática
   II. Ciclo de vida del software
   III. Modelos de ciclo de vida

El software del ordenador

Bloque I

El software del ordenador

Algunas definiciones de la palabra **Software:**

- Según la RAE, software es *el conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora*
- Según el estándar 729 del IEEE, es *el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación*

El software del ordenador

- En cualquier caso, el software es todo aquello que se refiere a los programas y datos almacenados en un ordenador:
  - **Programas** encargados de dar instrucciones para realizar tareas con el hardware o para comunicarnos con otro software
  - **Datos** necesarios para la ejecución de los programas

Clasificación del Software

Bloque I - A

1. Según el tipo de tarea que realizan
2. Según el método de distribución
3. Según la licencia de distribución

Para ampliar:

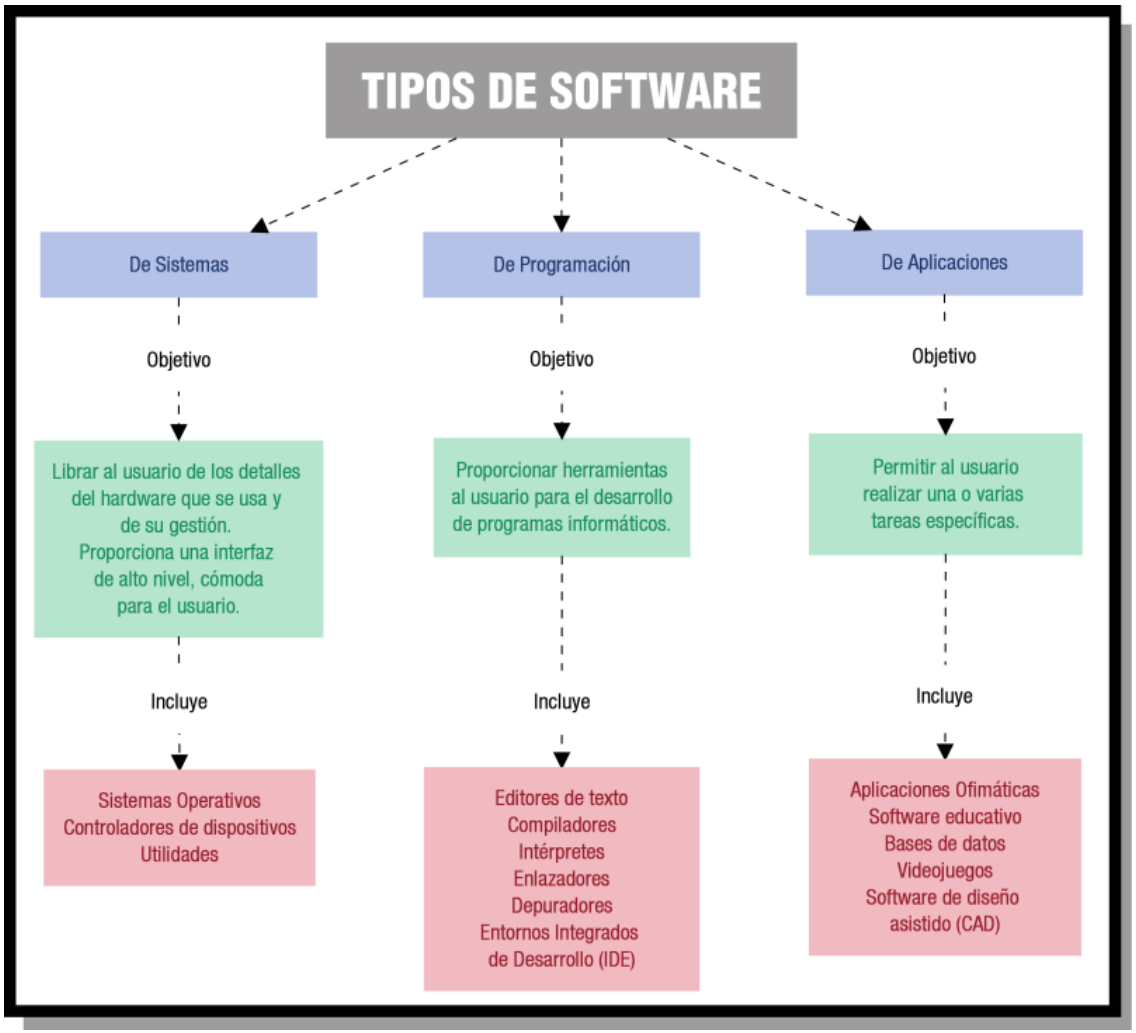
http://www.tiposdesoftware.com/

Clasificación del software

- Según el tipo de tarea que realizan:
  - De sistema
  - De aplicación
  - De programación o desarrollo

Clasificación del software

- Según el tipo de tarea que realizan:



Según el tipo de tarea

- **Software de sistema:** es aquel que permite que el HW funcione
- Lo forman los programas que permiten la administración de la parte física o los recursos del ordenador
- Es el que interactúa entre el usuario y los componentes HW del ordenador

  - Ejemplo: Sistemas operativos, los controladores de dispositivos, las herramientas de diagnóstico, las de corrección y optimización, etc.

Según el tipo de tarea

- **Software de aplicación:** lo forman los programas que nos ayudan a realizar tareas específicas en cualquier campo susceptible de ser automatizado o asistido
  - Este software hace que el ordenador sea una herramienta útil para el usuario
  - Ejemplo: las aplicaciones de control y automatización industrial, las aplicaciones ofimáticas, el software educativo, el software médico, las aplicaciones de contabilidad, software para llevar la gestión de un videoclub, software Multimedia, etc.

Según el tipo de tarea

- **Software de programación o desarrollo:** es el que proporciona al programador herramientas para ayudarle a escribir programas informáticos y a usar diferentes lenguajes de programación de forma práctica

  - Entre ellos se encuentran los entornos de desarrollo integrados (IDE), que agrupan las herramientas anteriores, normalmente en un entorno visual, de forma que el programador no necesite introducir múltiples comandos para compilar, interpretar, depurar, etc.
  - Habitualmente, cuentan con una avanzada interfaz gráfica de usuario (GUI)

Clasificación del software

- Según el método de distribución:
  - Shareware
  - Freeware
  - Adware
  - De pago

Según el método de distribución

- **Shareware:** es una modalidad de distribución para que el usuario pueda evaluar de forma gratuita el producto por un tiempo especificado
  - Para adquirir una licencia de software que permita el uso de manera completa se requiere de un pago (muchas veces modesto)
    - También existe el llamado "shareware de precio cero". Esta modalidad es poco común
  - Por ejemplo: los compresores de archivos Winzip, WinRAR; herramientas del sistema como PC File, ZoneAlarm; edición de imágenes como Paint Shop Pro; antivirus como Virus Scan, etc.

Según el método de distribución

- **Freeware:** Es un software que se distribuye sin cargo
  - A veces se incluye el código fuente pero no es lo usual
  - El freeware suele incluir una licencia de uso, que permite su redistribución pero con algunas restricciones:
    - no modificar la aplicación en sí,
    - ni venderla y
    - dar cuenta de su autor
  - Contrariamente a lo que se cree, los programas de software libre NO necesariamente son freeware

Según el método de distribución

- **Adware:** suelen ser programas shareware que de forma automática descargan publicidad en un ordenador cuando lo ejecutamos o lo instalamos
  - Hemos de estar atentos a la hora de instalarlos porque a veces se puede evitar su descarga
  - Al comprar la licencia del programa se elimina la publicidad

Según el método de distribución

- **Software de pago**
  - **al por Menor**

    El software al por menor o retail software es un método de distribución en el que el usuario final puede adquirir una aplicación comprándolo en una tienda de computadoras

    El software al por menor tiene una presentación y contiene un CD, instrucciones de instalación y manual de uso

    Ejemplo Microsoft Office
  - **Software OEM**

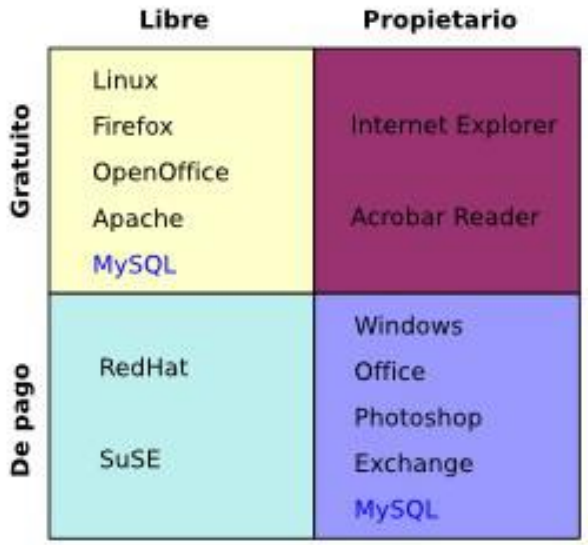
    EL software comercial se distribuye de forma masiva a los fabricantes de computadoras, los cuales instalan la aplicación en el equipo para poder venderlo

    Ejemplo: Un sistema operativo

Clasificación del software

Según la licencia bajo la cual se distribuye:

- Software libre
- Software propietario
- Software de dominio público



Según el tipo de licencia

- Una **licencia de software** es un contrato que se establece entre el desarrollador de un software, sometido a propiedad intelectual y a derechos de autor, y el usuario, en el cual se definen con precisión los derechos y deberes de ambas partes
  - Es el desarrollador, o aquel a quien este haya cedido los derechos de explotación, quien elige la licencia según la cual distribuye el software

Según el tipo de licencia

- El **software libre** es aquel en el cual el autor cede una serie de libertades básicas al usuario, en el marco de una licencia, que establece las siguientes libertades:
  1. Libertad de autorizar el programa con cualquier fin en cuantos ordenadores se desee
  2. Libertad de estudiar cómo funciona el programa y adaptar su código a necesidades específicas; para ello, como condición previa, es necesario poder acceder al código fuente
  3. Libertad de distribuir copiar a otros usuarios (con o sin modificaciones)
  4. Libertad de mejorar el programa (ampliarlo, añadir funciones) y de hacer públicas y distribuir al público las modificaciones; para ello, como condición previa, es necesario poder acceder al código fuente

Según el tipo de licencia

- El **software propietario** es aquel que, habitualmente, se distribuye en formato binario, sin posibilidad de acceso al código fuente según una licencia en la cual el propietario, por regla general, prohíbe alguna o todas las siguientes propiedades:
  - la redistribución,
  - la modificación,
  - copia,
  - uso en varias máquinas simultáneamente,
  - transferencia de la titularidad,
  - difusión de fallos y errores que se pudiesen descubrir en el programa, etc.
- Se dan diferentes variantes: Freeware y Shareware

Según el tipo de licencia

- El **software de dominio público** es aquel que carece de licencia o no hay forma de determinarla porque se desconoce al autor
  - El propietario abandona los derecho que lo acreditan como titular, o bien
  - Expira el plazo de propiedad

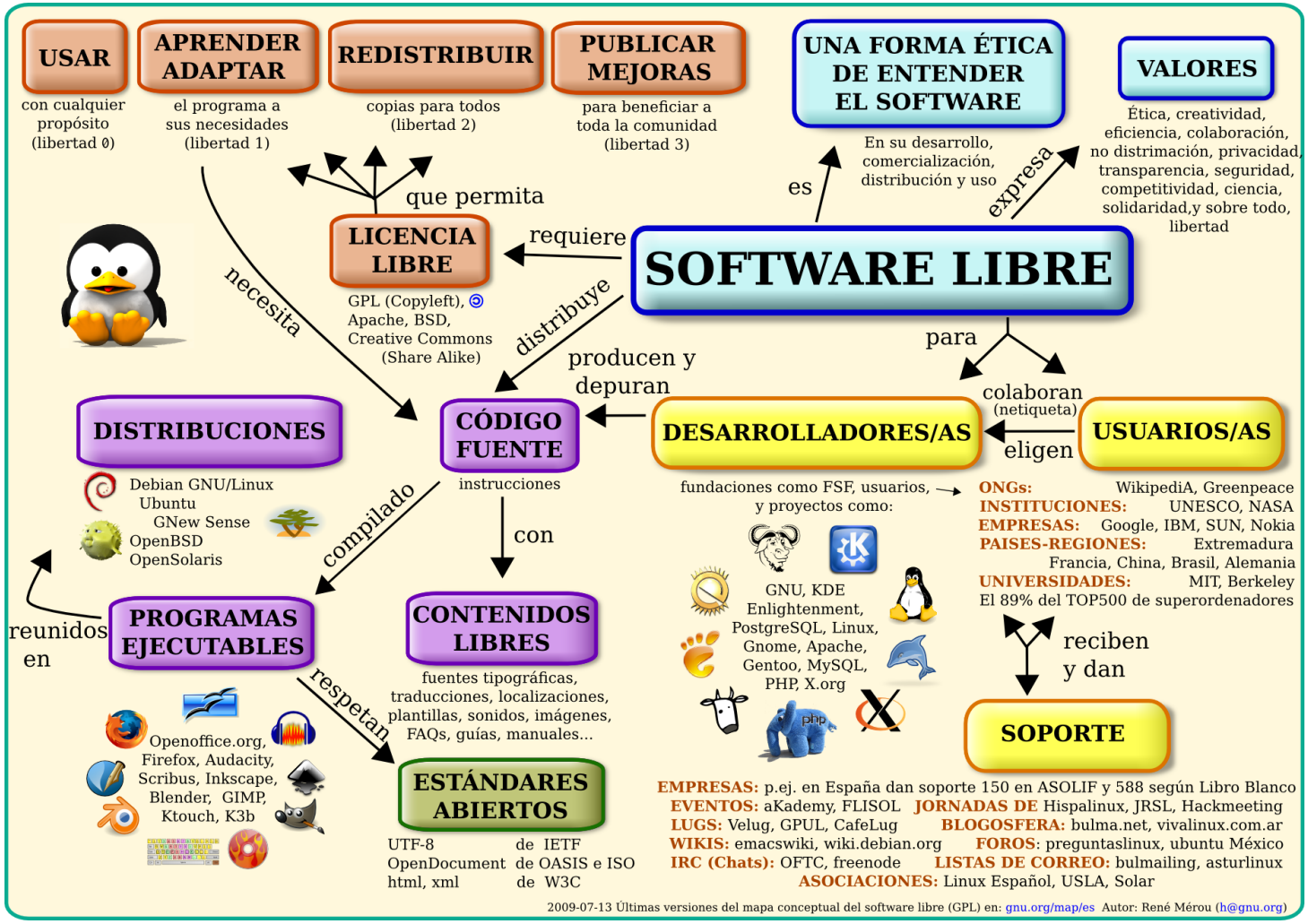
- Todo el mundo lo puede utilizar
- Se puede desarrollar una oferta propietaria sobre la base de un código de dominio público

Según el tipo de licencia

- La licencia más utilizada en los productos y desarrollos de software libre y de fuentes abiertas es la Licencia Publica General de GNU más conocida por su nombre en inglés **GNU General Public licence** (o simplemente sus siglas *GNU GPL)*
  - Esta licencia da derecho al usuario a usar y modificar el programa con la obligación de hacer públicas las versiones modificadas
- Más información sobre licencias de software libre compatibles con GPL:

  http://www.gnu.org/licenses/license-list.es.html#SoftwareLicenses

Según el tipo de licencia



Software de Programación

Bloque I – B

1. Concepto de programa
2. Lenguajes de programación
   - Clasificación y características
     - Según nivel de abstracción
     - Según la forma de ejecución
       - Tipos de código
       - Compilador e Interprete
       - Máquinas virtuales
         - La máquina virtual de Java
     - Según el paradigma de programación
3. Herramientas utilizadas en programación

Lenguajes de programación

- Un programa informático es un conjunto de **instrucciones escritas en un lenguaje de programación** que se ejecutan secuencialmente aplicadas sobre un conjunto de datos

Lenguajes de programación

- Un lenguaje de programación es un conjunto de caracteres, las reglas para la combinación de esos caracteres y las reglas que definen sus efectos cuando los ejecuta un ordenador
- Consta de los siguientes elementos:
  - Un **alfabeto o vocabulario (léxico):** formado por el conjunto de símbolos permitidos
  - Una **sintaxis:** son las reglas que indican cómo realizar las construcciones con los símbolos del lenguaje
  - Una **semántica:** son las reglas que determinan el significado de cualquier construcción del lenguaje

Clasificación y características

- Se pueden clasificar atendiendo a diversos criterios
- Según su **nivel de abstracción:**
  - Lenguajes de bajo nivel
  - Lenguajes de alto nivel
- Según la **forma de ejecución:**
  - Lenguajes compilados
  - Lenguajes interpretados
  - Máquinas Virtuales

Clasificación y características

- Según el **paradigma de programación:**
  - Lenguajes imperativos / estructurados
  - Lenguajes orientados a objetos
  - Lenguajes funcionales
  - Lenguajes lógicos

Según nivel de abstracción

- **Lenguajes de bajo nivel**
  - Son lenguajes de programación que se acercan al comportamiento del ordenador
  - *lenguaje máquina*
    - es entendible directamente por la máquina
    - Las instrucciones están formadas por cadenas de ceros y unos, es decir, utilizan el alfabeto binario (0 y 1)
    - Los programas son específicos para cada procesador
  - *lenguaje ensamblador*
    - Es difícil de aprender y es específico para cada procesador
    - Un programa escrito en este lenguaje necesita ser traducido a lenguaje máquina para poder ejecutarse

Según nivel de abstracción

- **Lenguajes de alto nivel**
  - Son más fáciles de aprender porque están formados por palabras del lenguaje natural, como el inglés.
  - Para poder ejecutarlos en el ordenador se necesita un programa (intérprete o compilador) que traduzca las instrucciones escritas en este lenguaje a instrucciones en lenguaje máquina que el ordenador pueda entender
  - Son independientes de la máquina. No dependen del HW del ordenador y no requieren ningún conocimiento de código máquina por parte del usuario que lo utiliza
  - Algunos de estos lenguajes son:
    - Basic, C++, C#, Clipper, COBOL
    - Fortran, Java, PHP, Perl, PL/SQL, etc.

Tipos de código

El código de un programa pasa por diferentes estados desde que se escribe hasta que se ejecuta en el ordenador:

- **Código fuente:** es el código de instrucciones que la computadora deberá realizar escrito por los programadores
  - Se utiliza un lenguaje de programación de alto nivel apropiado para el problema que se trata de resolver
  - Para escribir el código se parte de los diagramas de flujo o pseudocódigos diseñados en la etapa de diseño
  - Este código no es directamente ejecutable por el ordenador

Tipos de código

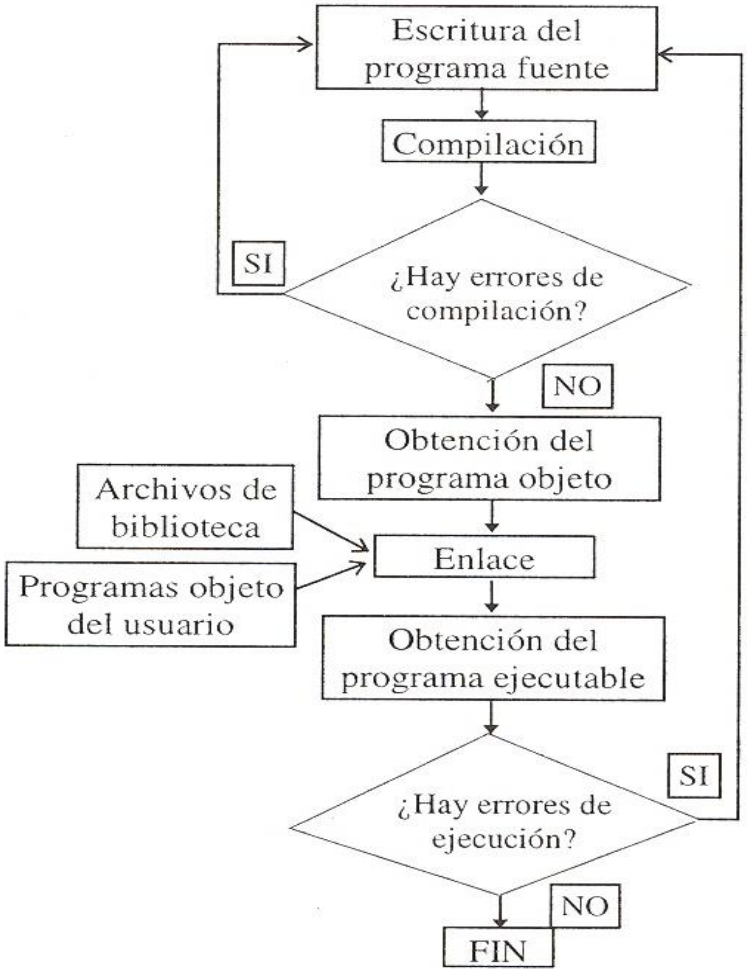
- **Código objeto:** es el resultado de traducir código fuente a un código equivalente formado por unos y ceros que aún no puede ser ejecutado directamente por la computadora
  - Es un código intermedio de bajo nivel (o bytecode)
- **Código ejecutable:** es el resultado de transformar el código objeto en código directamente ejecutable por la máquina

Según la forma de ejecución

- El proceso de traducción de código fuente a código objeto puede realizarse de dos formas:
- **COMPILADORES:**
  - Traducen completamente un programa fuente (el escrito en lenguaje de alto nivel) generando un programa objeto (semánticamente equivalente) escrito en lenguaje máquina
  - El compilador informa al usuario de los errores existentes en el programa fuente, pasándose a la creación del programa objeto sólo en caso de que no haya errores
  - El programa objeto se almacenará en un fichero que se podrá utilizar cuando se quiera, sin necesidad de volverse a hacer la traducción
  - Posteriormente, el programa enlazador o linker inserta en el código objeto las funciones de librería necesarias para producir el programa ejecutable
    - Por ejemplo, en un programa escrito en C si el fichero fuente hace referencia a funciones de una biblioteca o a funciones que están definidas en otros ficheros fuentes, entonces el enlazador combina estas funciones con el programa principal para crear un fichero ejecutable

Según la forma de ejecución

COMPILACIÓN:



Según la forma de ejecución

- **INTÉRPRETES:**
  - Permiten que un programa fuente vaya traduciéndose y ejecutándose directamente sentencia a sentencia por la computadora
  - El intérprete capta una sentencia fuente, la analiza y la interpreta, dando lugar a su ejecución inmediata. Por consiguiente no se crea ningún fichero objeto
    - Por ejemplo: PHP o JavaScript

Según la forma de ejecución

Según la forma de ejecución

Compiladores vs Interpretes:

- El intérprete es notablemente más lento que el compilador, ya que lleva a cabo la traducción a la vez que la ejecución
  - Además, esta traducción se hace siempre que se ejecuta el programa, mientras que el compilador sólo la lleva a cabo una vez
  - El intérprete elimina la necesidad de realizar una compilación después de cada modificación del programa
- La ventaja de los intérpretes es que hacen que los programas sean más portables
  - Un programa compilado en un ordenador con sistema operativo Windows no funcionará en un Macintosh, o en un ordenador con sistema operativo Linux, a menos que se vuelva a compilar el programa fuente en el nuevo sistema

Según la forma de ejecución

- La **Máquina Virtual** combina la compilación y la interpretación:
  - Un programa fuente se **compila** primero en un formato intermedio
  - Después una máquina virtual los **interpreta**

Según la forma de ejecución

- Las tareas principales de la **Máquina Virtual** son las siguientes:
  - Reservar espacio en memoria para los objetos creados y liberar la memoria utilizada
  - Comunicarse con el sistema huésped (sistema donde se ejecuta la aplicación) para ciertas funciones, como controlar el acceso a dispositivos HW
  - Vigilar el cumplimiento de las normas de seguridad de las aplicaciones

Según la forma de ejecución

- La principal **desventaja de los lenguajes basados en máquinas virtuales es** que son más lentos que los lenguajes completamente compilados, debido a la sobrecarga que genera tener una capa de software intermedia entre la aplicación y el HW del ordenador
- Esta desventaja no es crítica

Según el paradigma de programación

Atendiendo a la forma de trabajar de los programas y a la filosofía con la que fueron concebidos:

- Paradigma imperativo / estructurado
- Paradigma de objetos
- Paradigma funcional
- Paradigma lógico

Según el paradigma de programación

- El **paradigma imperativo** / **estructurado** debe su nombre al papel dominante que ejercen las sentencias imperativas, es decir aquellas que indican llevar a cabo una determinada operación que modifica los datos guardados en memoria.
- Algunos de los lenguajes imperativos son C, Basic, Pascal, Cobol ...
- La técnica seguida en la programación imperativa es la **programación estructurada**
- La idea es que cualquier programa, por complejo y grande que sea, puede ser representado mediante tres tipos de estructuras de control:
  - Secuencia
  - Selección
  - Iteración
- Propone desarrollar el programa con la técnica de diseño descendente (*top-down*):
  - Se trata de modular el programa creando porciones más pequeñas de programas con tareas específicas, que se subdividen en otras subprogramas, cada vez más pequeños.
  - La idea es que estos subprogramas típicamente llamados funciones o procedimientos deben resolver un único objetivo o tarea

Según el paradigma de programación

- El **paradigma de objetos** conocido como Programación Orientada a Objetos (POO o OEP en inglés) es una estrategia de construcción de programas basado en una abstracción del mundo real
  - Estos objetos son una representación directa de algo del mundo real, como un libro, una persona, un pedido, un empleado ...
- Algunos de los lenguajes de programación orientada a objetos son C ++, Java, C # ...
- Un objeto es una combinación de datos (llamadas atributos) y métodos (funciones y procedimientos) que nos permiten interactuar con él
- En este tipo de programación los programas son conjuntos de objetos que interactúan entre ellos a través de mensajes (llamadas a métodos)
- La programación orientada a objetos se basa en la integración de 5 conceptos:
  - abstracción, encapsulación, modularidad, jerarquía y polimorfismo

Según el paradigma de programación

- El **paradigma funcional** está basado en un modelo matemático. La idea es que el resultado de un cálculo es la entrada del siguiente y así sucesivamente hasta que una composición produzca el resultado deseado.
- Los creadores de los primeros lenguajes funcionales pretendían convertirlos en lenguajes de uso universal para el procesamiento de datos en todo tipo de aplicaciones, pero, con el paso del tiempo, se ha utilizado principalmente en ámbitos de investigación científica y aplicaciones matemáticas
- Uno de los lenguajes más típicos es el Lisp

Según el paradigma de programación

- El **paradigma lógico** tiene como característica principal la aplicación de las reglas de la lógica para inferir conclusiones a partir de datos
- Un programa lógico contiene una base de conocimiento sobre la que se llevan a cabo consultas. La base de conocimiento está formada por hechos, que representan la información del sistema expresada como relaciones entre los datos y reglas lógicas que permiten deducir consecuencias a partir de combinaciones entre los hechos y, en general, otras reglas
- Uno de los lenguajes es el Prolog

Herramientas de programación

- Para llevar a cabo la codificación y prueba de los programas se suelen utilizar *entornos de programación*
- Estos entornos nos permiten realizar diferentes tareas:
  - Crear, editar y modificar el código fuente del programa
  - Compilar, montar y ejecutar el programa
  - Examinar el código fuente
  - Ejecutar el programa en modo depuración
  - Realizar pruebas del programa de forma automática
  - Generar documentación
  - Gestionar los cambios que se van haciendo en el programa (control de versiones)
  - Etc.

Herramientas de programación

- A estos entornos de programación se les suele llamar *Entornos de Desarrollo Integrado* o ***IDE (Integrated Development Environment)***

- Los IDEs están diseñados para maximizar la productividad del programador
- Un IDE es un programa informático formado por un conjunto de herramientas de programación que facilitan las tareas de creación, modificación, compilación, implementación y depuración de software
- La mayoría de los IDEs actuales proporcionan un entorno de trabajo visual formado por ventanas, barras de herramientas, paneles laterales para presentar la estructura en árbol de los proyectos o del código del programa que estamos editando, pestañas adicionales con el resultado de aplicar determinadas herramientas, etc.

Herramientas de programación

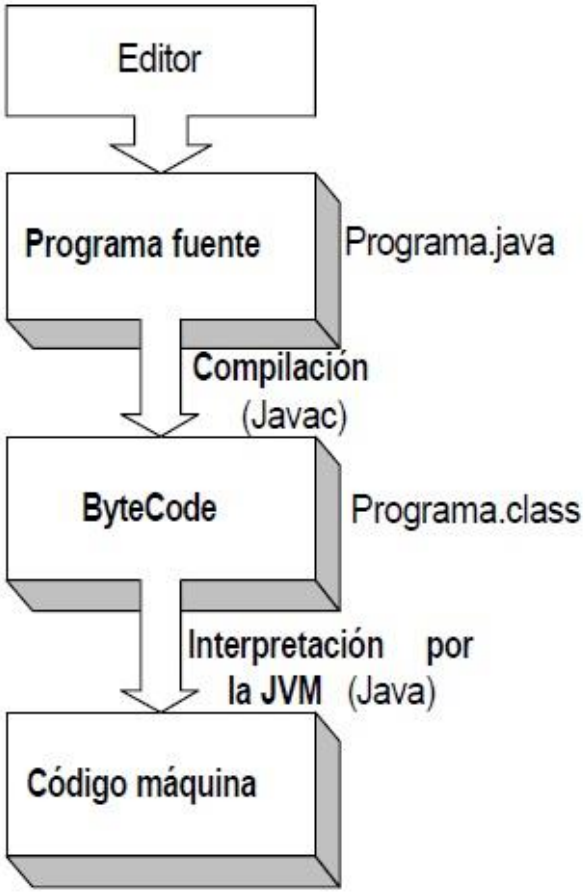
- Algunos ejemplos de IDE
  - Netbeans, Eclipse y Jcreator para los lenguajes Java
  - Visual Studio para los lenguajes C#, C++ i Visual Basic

Java

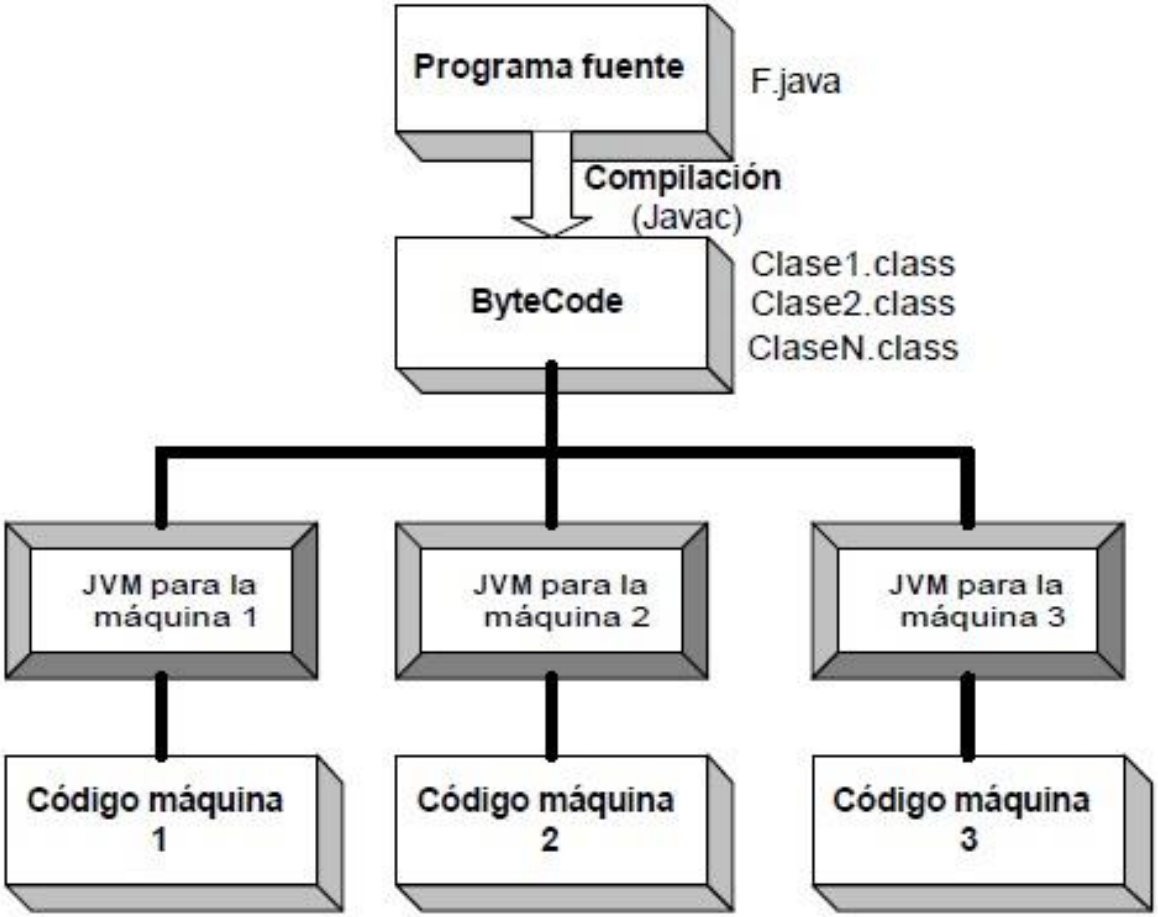
Con detalle

La Máquina virtual de Java (JVM)

- ¿Java es Compilado o Interpretado?
  - Aunque estrictamente hablando es interpretado, necesita de un proceso previo de compilación
  - Una vez "compilado" el programa, se crea un fichero que almacena lo que se denomina ***bytecodes*** (pseudocódigo prácticamente al nivel de código máquina)
  - Para ejecutarlo, es necesario un "intérprete", la **JVM (Java Virtual Machine) ó Máquina Virtual Java**
  - De esta forma, es posible compilar el programa en una estación UNIX y ejecutarlo en otra con Windows utilizando la máquina virtual Java para Windows
  - Esta JVM se encarga de leer los bytecodes y traducirlos a instrucciones ejecutables directamente en un determinado microprocesador, de una forma bastante eficiente



La Máquina Virtual Java (JVM)



La Máquina Virtual Java (JVM)

- Un mismo programa fuente compilado en distintas plataformas o sistemas operativos, genera el mismo fichero en bytecode
- La JVM realiza la traducción de ese bytecode a código nativo de la máquina sobre la que se ejecuta
- Existe una versión distinta de esta JVM para cada plataforma. Esta JVM se carga en memoria y va traduciendo “al vuelo”, los bytecodes a código máquina
- La JVM no ocupa mucho espacio en memoria

La Máquina Virtual Java (JVM)

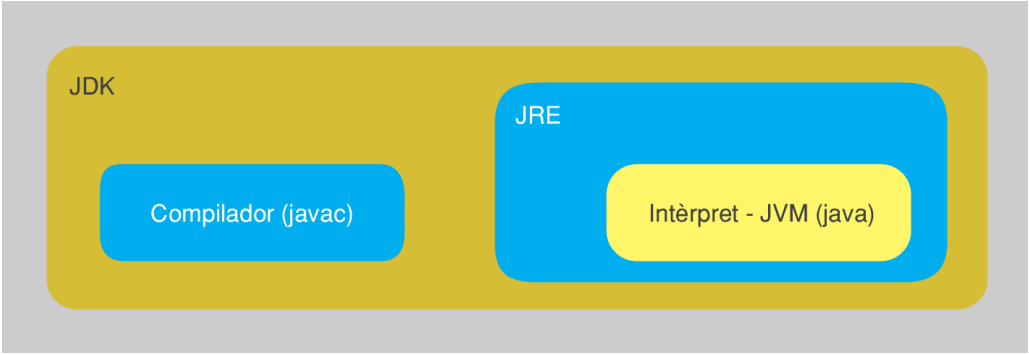
- El código fuente del programa está escrito en ficheros de texto plano que tienen la extensión **.java**
- La compilación del fichero, mediante el compilador Java **javac**, genera un fichero (o varios) con la extensión **.class** (siempre y cuando no haya errores)
- Un fichero .class contiene código en un lenguaje intermedio cercano al lenguaje máquinas pero independiente del ordenador y el SO en que se ejecuta, este código se llama **bytecode**
- La Máquina Virtual de Java toma y traduce el bytecode en código binario para el procesador que se utiliza para ejecutar el programa
- Como está disponible en muchos SO diferentes, los ficheros pueden ser ejecutados en distintas plataformas: MS Windows, Solaris, Linux o Mac OS.

Java Runtime Environment (JRE)

- Java Runtime Environment o JRE es un conjunto de utilidades que permite la **ejecución** de programas Java
- Está formado básicamente por:
  - Una Máquina Virtual Java o JVM
    - Es el programa que ejecuta el código Java previamente compilado (bytecode)
  - Un conjunto de bibliotecas Java para proporcionar los servicios que pueda necesitar la aplicación
    - (El API de JAVA formada por librerías de clases estándar)
  - Otros componentes…

El entorno de desarrollo JDK

- Un usuario sólo necesita el JRE para **ejecutar** las aplicaciones desarrolladas en lenguaje Java
- Para **desarrollar** nuevas aplicaciones en Java la herramienta básica es el JDK (*Java Developer's Kit*) o Kit de Desarrollo Java
- Incluye entre otros:
  - Un compilador
  - Un JRE (máquina virtual JVM y librerías)



- El Kit de desarrollo puede obtenerse en la dirección siguiente:

  http://www.oracle.com/technetwork/java/javase/downloads/index.html

El entorno de desarrollo JDK

- En el entorno para Windows está formado por un fichero ejecutable que realiza la instalación, creando toda la estructura de directorios
- El kit contiene básicamente:
  - El compilador: javac.exe
  - El depurador: jdb.exe
  - El intérprete: java.exe y javaw.exe
  - El visualizador de applets: appletviewer.exe
  - El generador de documentación: javadoc.exe
  - Un desensamblador de clases: javap.exe
  - El generador de archivos fuentes y de cabecera (.c y .h) para clases nativas en C: javah.exe