TEMA 1 – DAM – PROGRAMACIÓN

Generalmente, la primera razón que mueve a una persona hacia el aprendizaje de la programación es utilizar el ordenador como herramienta para resolver problemas concretos. Como en la vida real, la búsqueda y obtención de una solución a un problema determinado, utilizando medios informáticos, se lleva a cabo siguiendo unos pasos fundamentales.

| **Resolución de problemas** | |
| --- | --- |
| **En la vida real...** | **En Programación...** |
| Observación de la situación o problema. | **Análisis del problema:**requiere que el problema sea definido y comprendido claramente para que pueda ser analizado con todo detalle. |
| Pensamos en una o varias posibles soluciones. | **Diseño o desarrollo de algoritmos:**se establece una solución al problema sin entrar en detalles tecnológicos. Se aplican diferentes técnicas y principios para establecer de forma detallada los pasos a seguir para resolver el problema. |
| Aplicamos la solución que estimamos más adecuada. | **Resolución del algoritmo elegido en la computadora:** consiste en convertir el algoritmo en programa, ejecutarlo y comprobar que soluciona verdaderamente el problema. |

**Corrección y eficacia:** si resuelve el problema adecuadamente.

**Eficiencia:** si lo hace en un tiempo mínimo y con un uso óptimo de los recursos del sistema.

Para conseguirlo, cuando afrontemos la construcción de la solución tendremos que tener en cuenta los siguientes conceptos:

**Abstracción:** se trata de realizar un análisis del problema para descomponerlo en problemas más pequeños y de menor complejidad, describiendo cada uno de ellos de manera precisa. Divide y vencerás, esta suele ser considerada una filosofía general para resolver problemas y de aquí que su nombre no sólo forme parte del vocabulario informático, sino que también se utiliza en muchos otros ámbitos.

**Encapsulación:** consiste en ocultar la información que manejan de los diferentes elementos que forman el sistema. La forma de manejar es información no debe influir en el resto de elementos del sistema.

**Modularidad**: un proyecto software será dividido en módulos independientes, dependiendo de su tamaño, donde cada uno de ellos tendrá su función correspondiente. Los demás módulos del sistema podrán utilizar su funcionalidad sin necesidad de conocer cómo funciona internamente.

**¿Qué es un algoritmo?**

**Algoritmo:** secuencia ordenada de pasos, descrita sin ambigüedades, que conducen a la solución de un problema dado.

* Debe ser**preciso** e indicar el orden de realización paso a paso.
* Debe estar **definido**, si se ejecuta dos o más veces con los mismos datos de entrada, debe obtener el mismo resultado cada vez. Además, debe dar una respuesta a cualquier dato de entrada.
* Debe ser **finito**, debe tener un número finito de pasos.

Los algoritmos son independientes de los lenguajes de programación y de las computadoras donde se ejecutan. Un mismo algoritmo puede ser expresado en diferentes lenguajes de programación y podría ser ejecutado en diferentes dispositivos.

La diferencia fundamental entre **algoritmo** y **programa** es que, en el segundo, los pasos que permiten resolver el problema, deben escribirse en un determinado lenguaje de programación para que puedan ser ejecutados en el ordenador y así obtener la solución.

**Los lenguajes de programación** son sólo un medio para expresar el algoritmo, es decir, establece una serie de normas sintácticas y semánticas para expresarlo.

**Paradigma de programación:** es un modelo básico para el diseño y la implementación de programas. Este modelo determinará como será el proceso de diseño y la estructura final del programa.

Cuando se establece un patrón para la creación de aplicaciones nos estamos acercando al significado de la palabra paradigma. Si establecemos una serie de normas y principios que recojan experiencia y buenas prácticas de otros desarrolladores para su uso en la resolución de problemas, estaremos creando un paradigma de programación.

---------------------------------------------------------------------------------

**El proceso de creación de software** puede dividirse en diferentes fases:

* **Fase de resolución del problema.**
* **Fase de implementación.**
* **Fase de explotación y mantenimiento.**

RESOLUCIÓN:

* 1. **Análisis:** Los contactos entre el analista/programador y el cliente/usuario serán numerosos, de esta forma podrán ser conocidas todas las necesidades que precisa la aplicación.

En esta fase debemos aprender a analizar la documentación de la empresa , investigar, observar todo lo que rodea el problema y recopilar cualquier información útil.

* 1. **Diseño:** Durante la fase de diseño, se plantea la aplicación a desarrollar como una única operación global, y se va descomponiendo en operaciones más sencillas, detalladas y específicas. El nivel de descomposición dependerá del tamaño del problema. En cada nivel de refinamiento, las operaciones identificadas se asignan a módulos separados.

Hay que tener en cuenta que antes de pasar a la implementación del algoritmo, hemos de asegurarnos que tenemos una solución adecuada. Para ello, todo diseño requerirá de la realización de la **prueba o traza** del programa.

IMPLEMENTACION:

1. **Codificación o construcción**

Esta etapa consiste en transformar o traducir los resultados obtenidos a un determinado lenguaje de programación.

**Compilación:**Es el proceso por el cual se traducen las instrucciones escritas en un determinado lenguaje de programación a lenguaje que la máquina es capaz de interpretar, nomalmente código binario.

El proceso de compilación se puede llevar a cabo de dos formas:

* **Compilado**: El código se traduce todo a lenguaje máquina antes de ejecutarse (ej. C++).
* **Interpretado**: El código se ejecuta línea por línea en tiempo real (ej. Python).

1. **Prueba de ejecución y validación:** Para esta etapa es necesario implantar la aplicación en el sistema donde va a funcionar, debe ponerse en marcha y comprobar si su funcionamiento es correcto.

* **Documentación interna:** Encabezados, descripciones, declaraciones del problema y comentarios que se incluyen dentro del código fuente.
* **Documentación externa:** Son los manuales que se crean para una mejor ejecución y utilización del programa.

**Lenguaje de programación:**Conjunto de reglas sintácticas y semánticas, símbolos y palabras especiales establecidas para la construcción de programas. Es un lenguaje artificial, una construcción mental del ser humano para expresar programas.

Además, cada lenguaje de programación, al igual que otro tipo de lenguajes, se basa en una gramática.

**Gramática del lenguaje:** Reglas aplicables al conjunto de símbolos y palabras especiales del lenguaje de programación para la construcción de sentencias correctas.

Además, cada gramática dispone de:

1. **Léxico:**Es el conjunto finito de símbolos y palabras especiales, es el vocabulario del lenguaje.
2. **Sintaxis:** Son las posibles combinaciones de los símbolos y palabras especiales. Está relacionada con la forma de los programas.
3. **Semántica:** Es el significado de cada construcción del lenguaje, la acción que se llevará a cabo.

**Compilador:**Es un programa cuya función consiste en traducir el código fuente de un programa escrito en un lenguaje de alto nivel a lenguaje máquina. Al proceso de traducción se le conoce con el nombre de compilación.

**Intérprete:**Es un programa traductor de un lenguaje de alto nivel en el que el proceso de traducción y de ejecución se llevan a cabo simultáneamente, es decir, la instrucción se pasa a lenguaje máquina y se ejecuta directamente. No se genera programa objeto, ni programa ejecutable.

--------------------------------------------------------------------------

* **J2SE**: Entorno de Sun relacionado con la creación de aplicaciones y [applets](https://ikasaula.educacion.navarra.es/fponline/pluginfile.php/145558/mod_resource/content/11/PROG01_Contenidos_Imprimible/index.html#tdd1697a7-ea79-cab3-4cdb-d69f0afa8b75) en lenguaje Java para su ejecución en equipo y servidores.
* **J2EE**: Pensada para la creación de aplicaciones web Java empresariales y del lado del servidor.
* **Java FX**: Permite crear e implementar Aplicaciones de Internet Enriquecidas ([RIAs](https://ikasaula.educacion.navarra.es/fponline/pluginfile.php/145558/mod_resource/content/11/PROG01_Contenidos_Imprimible/index.html#tfda3b717-d151-48e1-0dac-86acb4155ce4)), cuya interfaz se comparta igual en distintas plataformas.
* **Java Embedded:**Plataforma creada para su ejecución en sistemas embebidos. Está muy relacionada con el conocido [Internet of Things](https://ikasaula.educacion.navarra.es/fponline/pluginfile.php/145558/mod_resource/content/11/PROG01_Contenidos_Imprimible/index.html#t10a2dcce-f707-fa73-3b53-605452487cd6) (Internet de las cosas).
* **Java Card**: tecnología que permite la ejecución de pequeñas aplicaciones Java en tarjetas inteligentes.

En Java se ha simplificado la gestión de memoria a través de la eliminación de la Aritmética de  , por lo que la incorporación del Garbage Collector evita que se produzca un crecimiento de los huecos libres en memoria, que recibe el nombre de  de memoria.

**Bytecode:**Son un conjunto de instrucciones en lenguaje máquina que no son específicas a ningún procesador o sistema de cómputo. Un intérprete de código de bytes (bytecodes) para una plataforma específica será quien los ejecute. A estos intérpretes también se les conoce como Máquinas Virtuales Java o intérpretes Java de tiempo de ejecución.

En el proceso de precompilación, existe un verificador de códigos de bytes que se asegurará de que se cumplen las siguientes condiciones:

* El código satisface las especificaciones de la Máquina Virtual Java.
* No existe amenaza contra la integridad del sistema.
* No se producen desbordamientos de memoria.
* Los parámetros y sus tipos son adecuados.
* No existen conversiones de datos no permitidas.

Para que un bytecode pueda ser ejecutado en cualquier plataforma, es imprescindible que dicha plataforma cuente con el intérprete adecuado, es decir, la máquina virtual específica para esa plataforma. En general, la Máquina Virtual Java es un programa de reducido tamaño y gratuito para todos los sistemas operativos.

Parte superior do formulário

En NetBeans, los archivos .**class** de un proyecto están alojados en la carpeta  y los **.Java** en la carpeta  .

Parte inferior do formulário

Los archivos correspondientes a las clases se alojan en la carpeta **build** y los archivos con el código fuente se alojan en la carpeta **src**.