ESTILOS DE PROGRAMACIÓN.

La programación ha ido evolucionando a lo largo de la historia, de una programación secuencial se paso a la estructurada, y posteriormente a la POO.

Esta evolución en el mundo de la programación va enfocada al objetivo, de la posibilidad de que, un usuario, pueda utilizar una determinada aplicación para obtener un determinado rendimiento.

Podemos entender como estilos de programación “a los métodos que existen para mejorar la calidad de los programas de computación”.

Para que un programa sea considerado como bueno, debe cumplir una serie de requisitos:

* Que funcione.
* No debe tener dificultades. Deben ser controlados todos los posibles errores que se puedan dar.
* Debe estar bien documentado. La documentación de una aplicación es uno de los elementos más importantes del producto, ya que una buena documentación permite realizar futuras actualizaciones de forma rápida a los programadores. Existen dos tipos de documentaciones:
  + Externas: Generalmente, de forma impresa se adjunta al propio proyecto. Existen dos tipos:
    - Cuaderno de carga: Dirigida al programador, es un documento que contendrá todo lo relacionado con la aplicación, dirigido al programador.
    - Manual de usuario: Donde de una forma didáctica, el usuario debe entender el manejo de la aplicación.
  + Interna: Hace referencia al comentario de código.
* Debe ser eficiente, es decir, debe realizar lo que se le exige al programa.

FASES DE CONSTRUCCION DE UN PROGRAMA:

La creación de la documentación exige de un programador, de tener un orden, una estructura planificada, que permita realizar una serie de fases que facilite la comprensión de cómo se ha hecho la aplicación, la cual va dirigida a otro programador. Las fases de construcción de un programa pueden contener los siguientes elementos:

1. **Fase 0**. Introducción: Se produce una descripción de la empresa, de los estamentos que han intervenido, etc. Lo dice la empresa si quiere ponerlo. En un cuaderno de carga la introducción puede contener elementos como:
   1. Con que software se ha realizado la aplicación.
   2. Que programas y versiones se han utilizado como apoyo de la aplicación.
   3. Los requisitos mínimos y recomendables que necesita la aplicación.
   4. Resolución optima de la aplicación para que se visualice bien.
   5. Como se instala y desinstala la aplicación.
   6. Etc.

En definitiva, en la introducción se plantean todos los elementos que son imprescindibles antes de instalar la aplicación.

1. **Fase 1**. Definición o planteamiento del problema: Se trata de indicar que es lo que hace nuestra aplicación, explicando que es lo que queremos obtener. Si es necesario, debemos indicar de que datos es los que vamos a partir y los que vamos a obtener. En definitiva, que es lo que hace esa aplicación.
2. **Fase 2**. Análisis del problema: Una vez planteado que es lo que se quiere hacer, el programador tiene que indicar como lo va a hacer. Para ello es necesario realizar un estudio exhaustivo de todos los elementos que van a intervenir. Estos elementos se pueden dividir en tres categorías:
   1. Diagramas de proceso: Hace referencia al análisis de todos los elementos que intervienen en el proyecto de forma grafica, utilizando para ello todas las herramientas existentes, por ejemplo modelo entidad-relación, planteamiento de las clases, organigramas, etc.
   2. Diseño de registros, variables y mensajes: Consiste en analizar, si los hubiere, todos los ficheros utilizados, indicando sus registros, nombres de campo, etc. Además se deberán incluir todas las variables que intervienen en la aplicación. Los mensajes deben salir como una imagen en los documentos.
   3. **Condición/es para la solución (IMPORTANTE)**: Cuando tenemos claro que es lo que hay que hacer, cuando analizamos que herramientas o estructuras de programación vamos a necesitar para resolver el problema, es imprescindible redactar como vamos a solucionar el problema. Si el problema tiene más de una solución se deberán plantear todos ellos, optando por cual es la solución más optima del problema (Escribir como se ha hecho el programa).
3. **Fase 3.** Programación a la solución del problema: Una vez que sabemos que hay que hacer y cómo hay que hacerlo, se programa el problema. Esto tiene dos partes:
   1. Organigramas: Esquema de lo que vamos a hacer (algoritmo de planificación, diagramas de flujo, no hacer algoritmos de planificación).
   2. Codificación: Hacer código fuente del programa (Imprimir todo el código para el proyecto).
4. **Fase 4.** Fase de edición, puesta a punto y pruebas: Entendemos por edición la posibilidad de editar mi código fuente o programa fuente (el que yo escribo), con la intención de linkarlo y obtener así el programa objeto que es el que entiende el ordenador. Es aquí donde se comprueba el perfecto comportamiento del mismo. El paso siguiente es realizar la puesta a punto junto con una serie de pruebas, que recibe el nombre de juego de ensayo, consistiendo en someter a nuestro programa fuente a una serie de situaciones con la intención de buscar posibles errores. Este juego de ensayo debe quedar perfectamente preparado para buscar en todos los aspectos errores. Existen una serie de técnicas que pueden ayudar a ese juego de ensayos, permitiendo controlar todos los aspectos de la programación, estas técnicas o conjunto de pruebas reciben dos nombres:
   1. Técnica de la caja blanca: Se basa en un minucioso examen de los procedimientos para verificar que todas las funciones y procedimientos funcionan correctamente, que la entrada de datos es adecuada, salida de datos adecuada, manejo adecuado de BBDD y ficheros…. La caja blanca tiene una serie de técnicas:
      1. Prueba de camino básico: Consiste en que se garantiza (escribe) que la siguiente aplicación se ha ejecutado totalmente al menos una vez.
      2. Prueba de estructura de control: Consiste en realizar pruebas que garanticen el buen uso de las estructuras condicionales, bucles y flujo de datos, garantizando:
         1. Pruebas de estructuras condicionales
         2. Condiciones del flujo de programación.
         3. Bucles: Consiste en hacer un examen exhaustivo de todos los bucles del programa, distinguiendo entre bucles simples y bucles anidados. Para los bucles simples, analizando cual es el resultado de cada uno son:
            1. Ejecutar la estructura repetitiva saltándose el bucle.
            2. Ejecutar solo una vez la estructura repetitiva.
            3. Ejecutar más de una vez la estructura repetitiva.
            4. Ejecutar n-1 veces la estructura repetitiva siendo n es el límite superior de la estructura.
            5. Ejecutar completamente la estructura repetitiva.

Con las estructuras anidadas:

1. Comenzar con el bucle mas interno y analizar hacia los demás
2. Sumar uno a los bucles.
3. Probar todos los bucles para que estén bien formados.
4. Etc.

* 1. Técnica de la caja negra: Se encarga de buscar errores de interfaz, errores en las estructuras (tablas, BBDD, ficheros…), errores de conexión, errores de rendimiento, errores de inicio y fin del programa, etc. La caja negra tiene una serie de técnicas:
     1. Técnica de partición equivalente: Consiste en hacer un estudio detallado de todos los valores limites de todos los datos que intervienen en la aplicación.
     2. Análisis de valores límite: Consiste en analizar todos los valores límite de todas las estructuras utilizadas. Se utilizan con arrays dinámicos.
     3. Pruebas de comparación o software redundante: Consiste en ejecutar nuestra aplicación en diferentes ordenadores, con diferentes características, con la intención de obtener los mismos resultados. Habrá que tener en cuenta en que maquina se va a explotar nuestra aplicación, ya que habrá que adaptar el proyecto a sus características.

1. **Fase 5.** Fase de documentación: Consiste en recopilar toda la información del proyecto y si es necesario encuadernarla adecuadamente para archivarla.
2. **Fase 6.** TODAS LAS IDEAS QUE EL PROGRAMADOR ESTIME OPORTUNAS.

Documentar una aplicación es obligatorio, totalmente necesario, pero son las empresas quienes determinan como hacerlo. Por tanto no existe una forma estandarizada de documentar, ya que con la intención de ahorrar tiempo y espacio muchas empresas no obligan a la documentación, o utilizan otras técnicas de documentación, como por ejemplo, documentos Word, programas HTML o fichas que evitan una acumulación de papeles en la empresa.