Principios de programación

El proceso de resolución de un problema del mundo real por medio de la computadora requiere de la escritura de un programa y su posterior ejecución. Un programa de computadora es una secuencia de instrucciones escritas en un lenguaje especializado. Dado que los programas son escritos por personas, la forma en que se resuelve cada problema puede variar de una persona a otra. Esto se debe a que, en realidad, la programación es un proceso que se apoya mucho en la creatividad.

La programación de computadoras, es, ante todo, un proceso mental. Un programa de computadora medianamente complejo no se resuelve sentándose inmediatamente en frente de la máquina y comenzando a escribir instrucciones de programa. Lo primero que hay que hacer es ingeniar y estructurar el proceso de solución.

Sea cual sea la solución implementada, existen una serie de pasos que deben seguirse. Algunos autores enumeran más pasos y otros menos. Algunos autores desglosan muchos pasos y otros autores incluyen algunos pasos dentro de otros. Pero, en general establecen siempre la misma secuencia de los que se conoce como el ciclo de vida de un programa.

En nuestro caso, y desglosados con mucho detalle, se pueden mencionar estos pasos del ciclo de vida de un programa:

1. Definición del problema.
2. Diseño de una solución computacional.
3. Creación del algoritmo y su descripción.
4. Codificación o implementación.
5. Pruebas y depuración.
6. Implantación.
7. Documentación.
8. Producción.
9. Mantenimiento.

1) Definición del problema (Análisis)

El primer paso es identificar un problema del mundo real y comprenderlo. Existe un problema, para una organización o para una persona, al que hay que darle solución por medios informáticos. Hay que identificarlo, definirlo, delimitarlo y comprenderlo. No confundamos el programa con el problema: el programa es la solución al problema por medios informáticos.

No podemos proporcionar una solución de algo que no conocemos y no hemos definido claramente. Como primer paso es fundamental entender el problema, para luego poder determinar cuáles serán las entradas, las salidas y los proceso.

a) **Entradas**: fundamentalmente las entradas determinan:

   i. Qué y cuántos valores debe recibir el programa para poder funcionar.
   ii. El orden en que se ingresarán los datos.
   iii. El rango en que se encuentra cada valor de entrada y de qué tipo debe ser.
   iv. En qué momento se deben proporcionar estos datos.
   v. Se debe tener presente si la definición del problema involucra restricciones particulares sobre los valores.
   vi. Se debe prever si se van o no a modificar dichos valores de entrada una vez que ya han sido introducidos.
   vii. Hay que saber si se tendrá validación de los datos. La validación de los datos es el proceso mediante el cual se realizan un conjunto de pruebas sobre los datos de entrada antes de empezar a procesarse para garantizar que estén correctos y/o completos.

viii. Es necesario pensar en los mensajes de error, en caso de que algo falle.
ix. El programa debe saber cuándo ha recibido todas las entradas.

b) **Salidas**: se requiere planear:
   i. Qué valores debe generar el programa cuando termine.
   ii. En qué formato se requiere ver las salidas (si tendrán encabezados, etc.) y en dónde (impresas, en pantalla, almacenadas, etc.)

c) **Proceso**: es necesario analizar qué tipo de manipulación deben recibir los datos de entrada para resolver el problema y producir a la salida los datos esperados.

2) Diseño de una solución computacional

Probablemente habrá diferentes soluciones para el problema. Hay que plantearlas y elegir una de ellas. Esto puede depender de costos, de disponibilidades de equipo, software y personal.

La solución elegida se diagrama utilizando herramientas de diseño de software. Se identifican bien las diferentes partes de que constará. Se determina la estructura del programa y la estructura de la base de datos.

Debe dividirse el programa en varias partes, bloques o módulos. Estos son pequeñas unidades que serán más fáciles de modificar cuando sea necesario. Estas partes están interconectadas y se suele hacer un diagrama de bloques que muestra la interconexión entre ellas. Usualmente a cada pequeña unidad se le llama *módulo*, *proceso* o *función*.

3) Creación del algoritmo y su descripción

El proceso de solución de un problema es deductivo, va de lo general a lo particular, paso a paso. Se puede definir un algoritmo como una serie de pasos que se siguen para la solución de un problema determinado. Se puede escribir en prosa, como una lista de pasos, como pseudo-código o como diagrama de flujo. Cada módulo que conforma el diseño procede a ser definido detalladamente por medio de un algoritmo.

Propiedades de los algoritmos:

- La limitación: un algoritmo debe ser finito, es decir, debe tener un número limitado de pasos. El número de pasos depende del grado de detalles que se quiera obtener.

- Ausencia de ambigüedad: Un algoritmo debe ser claro en cada uno de sus pasos y resolver un tipo de problema específico.

- Debe tener un flujo de control: las instrucciones se deben ejecutar en un orden particular para resolver el problema.

- Entradas y salidas claramente definidas: esto es, lo que se le proporciona al algoritmo como insumo y que se espera obtener al final de su ejecución.

- Eficacia: el algoritmo siempre hará lo que se espera de él.

4) Codificación o Implementación

Se debe traducir el algoritmo a un lenguaje de programación, tal como C, C++, PHP, Python, Java, C#, etc.

5) Pruebas y depuración

Existen básicamente tres tipos de pruebas en un programa:

- Pruebas alfa (alfa test): se debe probar cada módulo del programa por separado, asegurando que cada uno de estos funcione correctamente.

- Pruebas beta (beta test): se debe poner a funcionar todos los programas juntos y asegurarse que cada módulo opere apropiadamente ya unido e interactuando con otros. La palabra interactuando significa que los procesos o módulos de un programa pueden comunicarse entre sí, es decir, intercambian información.

- Pruebas gama (gamma test): se debe comprobar qué tanto funciona el programa para el usuario. Realizar pruebas para cualquier tipo de usuario del programa.

6) Implantación

Un técnico programador y/o analista recibe la aplicación y la pone en producción, es decir, la pone a disposición de los usuarios. En algunas ocasiones tal vez sea necesario hacer pequeñas modificaciones al entorno de trabajo. Por ejemplo, se asignan permisos para ciertos usuarios o se le quitan a otros.

7) Documentación

No hay software sin documentación, esta se considera parte del software. La hay dos tipos:

- Documentación de sistemas: se describe detalladamente qué y cómo se han resuelto los problemas en el programa, está escrita en lenguaje técnico pues está dirigida a personal informático. Esta documentación puede ser:

  - Interna: se escriben comentarios aclarativos dentro del código del programa.

  - Externa: se escriben párrafos descriptivos en un documento separado del programa.

- Documentación de usuario: se describe la forma en que el usuario debe utilizar el programa, debe estar escrita en lenguaje natural y accesible para cualquier persona.

8) Producción

Es la etapa en la que se pone el programa a trabajar, poniéndolo a disposición de los usuarios. Es decir, el programa se utiliza para resolver los problemas. El tipo de problemas que resuelve depende del tipo de flexibilidad que el programador le haya dado para adaptarse automáticamente a las diversas situaciones que se le pueden presentar.

9) Mantenimiento

El diseño de los programas no es perfecto, tarde o temprano presentan errores, o bien cambia la realidad o el ambiente para los cuales fueron creados. Por ello requerirán transformaciones de fondo. En alguna medida un programa refleja la

realidad existente, por lo cual debe ser revisado cada cierto período de tiempo.