

---

---

# Asignatura Programación

## Apuntes de clase

### El proceso de programación

#### Introducción

El mundo real es naturalmente complejo y en muchas ocasiones los problemas a resolver resultan difíciles de sintetizar. Cada problema planteado requiere de un análisis minucioso y de la determinación clara y concreta del objetivo.

La especificación del problema debe ser unívoca, para ello es necesario expresar el problema en el lenguaje correcto, formalmente riguroso y que no permita ambigüedades.

Si se pudiera reducir los problemas del mundo real a expresiones matemáticas, como ocurre con algunos problemas de la Física tendríamos resuelto este tema de la ambigüedad. A modo de ejemplo, para calcular la velocidad que lleva un cuerpo con movimiento rectilíneo uniforme, podemos alcanzar con aplicar la ecuación:  $\text{velocidad} = \text{espacio} / \text{tiempo}$ ; pero este no es el caso, las áreas de la aplicación de la informática no siempre permiten que un problema se reduzca a combinaciones de expresiones algebraicas, por lo tanto, hay que considerar otras maneras de expresar nuestras soluciones.

En cuanto a la notación elegida: es importante la forma en la que se describirán los algoritmos. Se sugiere usar el lenguaje natural, el español y las estructuras algorítmicas fundamentales.

#### La programación

Un programa es un conjunto de instrucciones, ejecutables sobre una computadora, que permite cumplir una función específica. La escritura de un programa que represente una solución ejecutable por computadora es la piedra basal de las Ciencias de la Computación.

Se puede perder mucho dinero y tiempo si la tarea no fue hecha con prolijidad. Alguna vez, ¿ha desperdiciado mucho tiempo escribiendo el programa equivocado? o ¿ha tratado de realizar un cambio en un programa escrito por otra persona? ¿Cuánto tiempo perdió?

Situaciones como estas les ocurren a los programadores todo el tiempo, porque no tienen disciplina en su práctica de la programación y les ocasionan pérdidas de tiempo, económicas, de motivación. La mayoría de los programadores ignoran temas como prueba, depuración, alternativas de diseño y estilo, puesto que no trabajaron con ellos en su etapa de aprendizaje. Los aprenden penosamente a medida que van adquiriendo experiencia, y algunos otros nunca los llegan a aprender, lo cual no es rentable.

La propuesta de esta 1ª unidad de trabajo, es mostrar como debe ser el proceso de programación de manera tal que esta modalidad sea adoptada como una disciplina de trabajo y se puedan evitar males posteriores.

Independiente del lenguaje en que se escriba, la tarea del programador es hacer lo mejor posible con las herramientas disponibles. Un buen programador puede sobreponerse a un lenguaje pobre o a un sistema operativo torpe, pero ni siquiera un excelente lenguaje de programación salvará a un mal programador.

La tarea de programar consiste en encontrar la solución a un problema, de manera tal que la misma sea expresada como un conjunto de comandos o instrucciones que puedan ser ejecutadas por una computadora. El resultado es un programa escrito en un lenguaje de programación.

Existen distintas metodologías para escribir programas. La metodología propuesta a lo largo de este curso, es un procedimiento formal que deberá dominar, que hará que su tarea de programar le resulte más sencilla.

## Etapas en la resolución de problemas con computadora

Para escribir un programa deberá tener en cuenta las etapas siguientes:

### Análisis del problema

La primera etapa es la definición del problema. Generalmente el problema a resolver está mal definido. Considere que el problema es encontrar el número de teléfono de una persona. Surgen cuestiones tales como: ¿Cuál es el nombre de la persona? ¿Dónde vive esa persona? ¿Dónde están todos los números de teléfonos? Estas preguntas deben tener respuesta antes de empezar a escribir la solución.

El problema es analizado en el contexto del mundo real para interpretar sus aspectos esenciales y el objetivo que persigue. Como resultado de este análisis el problema queda definido. Es muy importante que en esta definición esté expresado en forma clara cuales son los datos, las transformaciones que pueden sufrir y los resultados esperados.

### Diseño del algoritmo

Un algoritmo es un conjunto de reglas para llevar a cabo un procedimiento que permita resolver un problema, ya sea a mano o en una máquina. Una vez que representa el problema, se busca la solución y puede ocurrir que se encuentren varios algoritmos como solución del mismo.

Para buscar el número de teléfono se puede buscar secuencialmente, lo cual daría lugar a una solución torpe y lenta. Otra alternativa es aprovechar las entradas de índice de la parte superior de cada página recorriendo la guía hasta encontrar la página correcta. Luego se buscará secuencialmente en la página. Este algoritmo es difícil de describir, pero más rápido.

Dado que un algoritmo describe la solución del problema, el modo en que se desarrolla el algoritmo es usando la metodología de descomposición arriba-abajo. Esto significa que se parte de una idea general y se va definiendo cada paso con mas detalle. El problema inicial se descompone progresivamente en módulos o partes que tendrán soluciones claras y definidas. En algunas etapas se pueden emplear algoritmos preexistentes. El programador es el administrador de estos módulos, y de la correcta asociación entre ellos obtendrá la solución final. Al ir desarrollando problemas más complejos se va construyendo un repertorio de técnicas que podemos asociar para resolver los problemas. También debe establecerse de que manera se comunicaran los módulos entre sí para intercambiar información.

Las herramientas elegidas para hacer los diseños de los algoritmos son las estructuras algorítmicas fundamentales.

### Codificación - Escritura de programas

Después de elegir el algoritmo se lo “traduce” al lenguaje de programación elegido. El proceso de escribir las instrucciones reales de un lenguaje de programación se llama codificación. La secuencia de tales sentencias se llama código. Es importante no realizar ninguna codificación hasta que el algoritmo esté casi perfectamente definido, para que estar obligados a un solo código y no sea necesario “cambiarse de mente”. La codificación es solo una etapa de la programación, la programación no debe confundirse con la codificación.

### Corrección, prueba y optimización

Cuando el programa se va a ejecutar, la computadora traduce este programa escrito en un lenguaje de programación a lenguaje de máquina. Esta traducción, denominada compilación, permite detectar los errores de sintaxis y corregirlos. En la etapa de prueba pueden aparecer los errores de ejecución, que implican un comportamiento anormal del programa y generalmente interrumpen la ejecución del mismo. Por ejemplo: Divisiones por cero, índices fuera de rangos, etc.

Hay otro tipo de errores más peligrosos y difíciles de corregir, son los errores de tipo lógicos. Ellos conducen a resultados inesperados y no detienen la ejecución, son errores en la expresión de la solución: se pensó una cosa pero se escribió otra.

Los errores de tipo lógico generalmente se descubren después de los errores de sintaxis y de ejecución. Si el programa es corto y de lógica sencilla, se recomienda para detectarlos un seguimiento a lápiz y papel. Si el programa es largo y ha sido dividido en módulos, se indica seguimiento por módulo con resultados intermedios que se irán controlando. Si el programa es grande y no está modularizado, será difícil encontrar los errores.

### **Verificación**

Se debe verificar que un programa que ya ha sido corregido desde un punto de vista sintáctico y lógico produzca los resultados deseados. Para ello, el programador diseñará diferentes conjuntos de datos de prueba, que hagan funcionar al programa en condiciones extremas a fines de medir su correctitud.

### **Documentación**

Es importante que el trabajo que se está desarrollando sea documentado. Los programas deben ser legibles y llevar consigo la documentación necesaria de manera tal que la interpretación del código no presente dificultades.

Como elementos de documentación interna del programa y para hacer los programas más legibles, se usan diversas técnicas:

- Uso de sangría o indentación: para poner de manifiesto en forma visual la dependencia funcional entre las instrucciones que componen un programa es conveniente usar sangría o indentación, esto es, comenzar la línea más adentro que la anterior.
- Uso de comentarios: distribuir a lo largo del código comentarios que expliquen los objetivos que persigue ese sector de instrucciones.

Cuando el programa es muy grande, se recomienda escribir la documentación adecuada para el usuario. Documentar en forma apropiada un programa permite modificarlo más fácilmente.

### **La buena programación y el buen estilo**

Programar bien es un arte y hay muchas opiniones acerca de cual es la mejor forma de escribir un programa. Un buen estilo de programación hace que el código sea fácil de leer para el programador y para los demás. Los principios del estilo de programación están basados en el sentido común: una lógica directa, expresión natural, lenguaje convencional, nombres con significado, comentarios útiles, estos son algunos de los elementos que enriquecen el estilo de programar.

Para determinar cuan correcto es un programa, se usarán tres criterios: correctitud, claridad, eficiencia.

Un programa correcto debe producir resultados correctos. Un programa claro debe ser fácil de entender. Un programa claro es fácil de escribir, depurar y mantener, por lo tanto, es más barato a lo largo de su vida que un programa obscuro.

### **Algunas recomendaciones.**

Para lograr claridad, se recomienda trabajar usando la metodología arriba-abajo, y dividir el problema en módulos o sectores con objetivos claros, con significado claro, con estructura lógica bien definida. Elegir los mejores recursos del lenguaje, usar las estructuras adecuadas, comentarios, líneas en blanco y sangría apropiada

Un programador no puede perder de vista la eficiencia de sus programas. La eficiencia se mide en función del espacio (cuanto ocupa el programa), tiempo de procesamiento (cuanto demora en su ejecución) y de la cantidad de datos.