# UNIVERSIDAD NACIONAL DE CUYO

# FACULTAD DE INGENIERÍA

# PROGRAMACIÓN I

Primer Año
Licenciatura en Ciencias de la Computación

# Guía Teórica - Unidad Nº 1

# Introducción

## 1. Computación e Informática

Computación es sinónimo de informática y como tal, se refiere a la tecnología desarrollada para el tratamiento automático de la información mediante el uso computadora. Computación, se refiere a la acción y efecto de computar, realizar una cuenta, un cálculo matemático.

*¿Qué es una Computadora?*

Una computadora es un dispositivo electrónico que recibe y procesa datos para convertirlos en información conveniente y útil que posteriormente se envían a las unidades de salida. Una computadora es capaz de ejecutar cálculos y tomar decisiones a velocidades millones o cientos de millones más rápidas que puedan hacerlo los seres humanos. En el sentido más simple una computadora es "un dispositivo" para realizar cálculos o computar.

Las computadoras se construyen y se incluyen en todo tipo de dispositivos: automóviles, aviones, trenes, relojes, televisores, etc. A su vez estas máquinas pueden enviar, recibir, almacenar, procesar y visualizar información de todo tipo: números, texto, imágenes, gráficos, sonidos, video, etc.

Los dos componentes principales de una computadora son: *hardware* y *software*. El *hardware* es la computadora en sí misma, es decir, su composición física o los dispositivos asociados con una computadora (circuitos electrónicos, cables, gabinete, teclado, etcétera). Sin embargo, para ser útil una computadora necesita además del equipo físico, un conjunto de instrucciones dadas. El conjunto de instrucciones que indican a la computadora aquello que deben hacer se denomina *software* o programas y se escriben por programadores. En esta materia nos centraremos en la enseñanza y aprendizaje de la programación o proceso de escribir programas.

Los datos y la información se pueden introducir en la computadora por una entrada y luego se procesan para producir una salida (resultados) como se ilustra en la *Figura 1*. La computadora se puede considerar como una unidad en la que se introducen ciertos datos (entrada de datos/información), se procesan y se produce un resultado (datos de salida o información). Los datos de entrada y los datos de salida pueden ser, realmente, de cualquier tipo: texto, dibujos, sonido, imágenes, etc. El sistema más sencillo para comunicarse una persona con la computadora es mediante un teclado, una pantalla (monitor) y un mouse. Hoy en día existen otros dispositivos muy populares tales como escáneres, micrófonos, cámaras de vídeo, teléfonos inteligentes, iPad, reproductores de música MP3, iPod, etc.

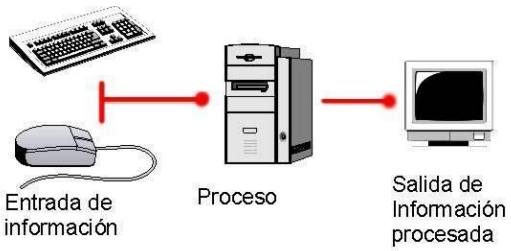


*Figura 1: Proceso de Información en una Computadora*

*Organización Física de una Computadora*

La *Figura 2* muestra la integración de los componentes que conforman una computadora cuando se ejecuta un programa; las flechas conectan los componentes y muestran la dirección del flujo de información.

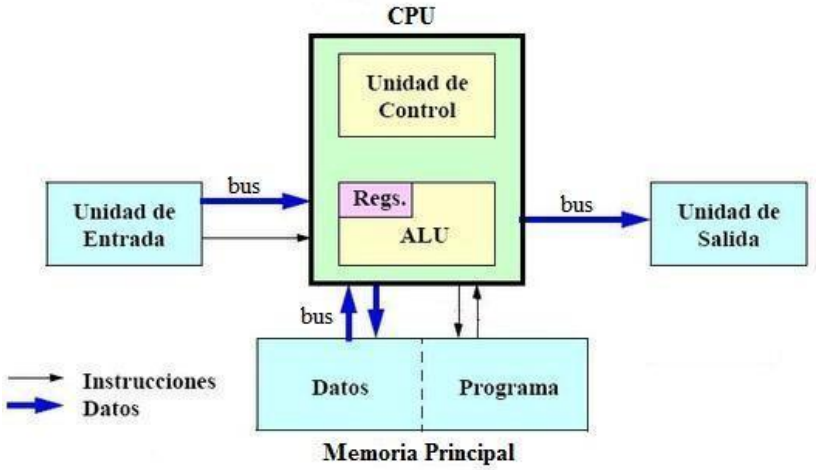


*Figura 2: Organización Física de una Computadora*

Las computadoras sólo entienden un lenguaje compuesto únicamente por ceros y unos. Esta forma de comunicación se denomina *sistema binario* digital y en el caso concreto de las computadoras, código o *lenguaje máquina*. Este lenguaje máquina utiliza secuencias o patrones de ceros y unos para componer las instrucciones que posteriormente reciben de los diferentes dispositivos de la computadora, tales como el microprocesador, las unidades de discos duros, los teclados, etc.

- *Unidades/Dispositivos de Entrada y Salida:* Los dispositivos de Entrada/Salida (E/S) permiten la comunicación entre la computadora y el usuario. Los *dispositivos de entrada*, sirven para introducir datos (información) en la computadora para su proceso. Los datos se leen de los dispositivos de entrada y se almacenan en la memoria central o interna. Algunos ejemplos son los teclados y mouse. Por otra parte, los *dispositivos de salida* permiten representar los resultados (salida) del proceso de los datos. El dispositivo de salida típico es la pantalla o monitor. Otros ejemplos son las impresoras donde los resultados se imprimen en papel.

- *Memoria Central o Principal:* Es uno de los componentes más importantes de una computadora y se utiliza para almacenar, de modo temporal información, datos y programas. En general, la información almacenada en memoria puede ser de dos tipos: las instrucciones de un programa y los datos con los que operan las instrucciones. Para que un programa se pueda ejecutar, debe ser situado en la memoria central, en una operación denominada carga (load) del programa. Después, cuando el programa se ejecuta (funciona), cualquier dato a procesar se debe llevar a la memoria mediante las instrucciones del programa. En la memoria central, hay también datos diversos y espacio de almacenamiento que necesita el programa cuando se ejecuta y así poder funcionar. Es una zona de almacenamiento organizada en centenares o millones de unidades de almacenamiento individual denominadas celdas. Una *celda o posición* de memoria es el elemento fundamental en el que se basa la memoria informática.
  La memoria principal consta de un conjunto de celdas de memoria (estas celdas o posiciones de memoria se denominan también palabras, aunque no "guardan" analogía con las palabras del lenguaje). Cada palabra puede ser un grupo de 8 bits, 16 bits, 32 bits o incluso 64 bits, en las computadoras más modernas y potentes. El bit es la unidad de información más pequeña que puede tratar una computadora y puede tomar los dos posibles valores del sistema binario: "cero o uno". Si la palabra es de 8 bits se conoce como byte. El término byte es muy utilizado en la jerga informática y, normalmente, las palabras de 16 bits se suelen conocer como palabras de 2 bytes, y las palabras de 32 bits como palabras de 4 bytes.

- *Unidad Central de Proceso (CPU):* La CPU es el hardware dentro de una computadora u otros dispositivos programables, que interpreta las instrucciones de un programa y realiza sus funciones de procesamiento de los datos, constituyendo el cerebro y corazón de la computadora o también su sistema nervioso. El procesador se encarga de un modo práctico de realizar numerosos cálculos y operaciones ordenadas por los diferentes programas instalados en la computadora. Dos componentes típicos de una CPU son la unidad aritmético lógica (ALU), que realiza operaciones aritméticas y lógicas, y la unidad de control (UC), que extrae instrucciones de la memoria, las decodifica y las ejecuta, llamando a la ALU cuando sea necesario. Cada computadora tiene al menos una CPU para interpretar y ejecutar las instrucciones de cada programa, realizar las manipulaciones de datos aritméticos y lógicos, y comunicarse con todas las restantes partes de la máquina indirectamente a través de la memoria.

*Proceso de ejecución de un programa*

El *programa* se debe transferir primero desde la memoria secundaria a la memoria principal antes de que pueda ser ejecutado. Los datos se deben proporcionar por alguna fuente. La persona que utiliza un programa (usuario) puede proporcionar datos a través de un *dispositivo de entrada*. Los datos pueden proceder de un archivo, o pueden proceder de una máquina remota vía una conexión de red o bien desde Internet.

Los datos se almacenan en la *memoria principal* de una computadora a la cual se puede acceder y manipular mediante la *unidad central de proceso (CPU).* Los resultados de esta manipulación se almacenan de nuevo en la memoria principal. Por último, los resultados (la información) de la

memoria principal se pueden visualizar en un *dispositivo de salida*, guardar en un almacenamiento secundario o enviarse a otra computadora conectada con ella en red.
Los paquetes de datos se mueven continuamente entre la CPU y todos los demás componentes (memoria RAM, disco duro, etc.). Estas transferencias se realizan a través de *buses*. Los *buses* son los canales de datos que interconectan los componentes del PC; algunos están diseñados para transferencias pequeñas y otros para transferencias mayores.

## 2. Resolución de Problemas

El programador de computadora es antes que nada una persona que resuelve problemas, por lo que para llegar a ser un programador eficaz se necesita aprender a resolver problemas de un modo riguroso y sistemático. Para poder resolver un problema hay que comprender su contexto. El *contexto de un problema* implica la realidad cercana a un determinado problema, se refiere a todo aquello que rodea, ya sea física o simbólicamente, a un acontecimiento. A partir del contexto, por lo tanto, se puede interpretar o entender un hecho para luego proponer una solución, por ejemplo, a través de un algoritmo.

### *Algoritmo*

Un *algoritmo* es un método para resolver un problema. La palabra *algoritmo* se deriva de la traducción al latín de la palabra Alkhô-warîzmi, nombre de un matemático y astrónomo árabe que vivió durante el siglo IX y alcanzó gran reputación por el enunciado de las reglas paso a paso para sumar, restar, multiplicar y dividir números decimales; la traducción al latín del apellido en la palabra algorismus derivó posteriormente en algoritmo. Alkhô-warîzmi escribió un tratado sobre manipulación de números y ecuaciones en el siglo IX. Un algoritmo es un método para resolver un problema mediante una serie de pasos precisos, definidos y finitos.

Los algoritmos son independientes tanto del lenguaje de programación en que se expresan como de la computadora que los ejecuta. En cada problema el algoritmo se puede expresar en un lenguaje diferente de programación y ejecutarse en una computadora distinta; sin embargo, el algoritmo será siempre el mismo. Así, por ejemplo, en una analogía con la vida diaria, una receta de un plato de cocina se puede expresar en español, inglés o francés, pero cualquiera que sea el lenguaje, los pasos para la elaboración del plato se realizarán sin importar el idioma del cocinero.

En la ciencia de la computación y en la programación, los algoritmos son más importantes que los lenguajes de programación o las computadoras. Un lenguaje de programación es tan sólo un medio para expresar un algoritmo y una computadora es sólo un procesador para ejecutarlo. Tanto el lenguaje de programación como la computadora son los medios para obtener un fin: conseguir que el algoritmo se ejecute y se efectúe el proceso correspondiente.

Dada la importancia del algoritmo en la ciencia de la computación, un aspecto muy importante será el diseño de algoritmos*. ¡A la enseñanza y práctica de esta tarea nos dedicaremos en esta materia!*

*Las características fundamentales que debe cumplir todo algoritmo son:*

- debe ser preciso e indicar el orden de realización de cada paso.
- debe estar bien definido. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez. Esto en matemática se adjetiviza como "determinístico".
- debe ser finito. Si se sigue un algoritmo, se debe terminar en algún momento; o sea, debe tener un número finito de pasos.

La definición de un algoritmo debe describir tres partes: Entrada, Proceso y Salida. En el algoritmo de receta de cocina citado anteriormente se tendrá:

- *Entrada*: Ingredientes y utensilios empleados.
- *Proceso*: Elaboración de la receta en la cocina.
- *Salida*: Terminación del plato (por ejemplo, cordero).

*Ejemplo de Algoritmo 1:*
Un cliente ejecuta un pedido a una fábrica. La fábrica examina en su banco de datos la ficha del cliente, si el cliente es solvente entonces la empresa acepta el pedido; en caso contrario, rechazará el pedido. Redactar el algoritmo correspondiente.
Los pasos del algoritmo son:

*1. Inicio.*
*2. Leer el pedido.*
*3. Examinar la ficha del cliente.*
*4. Si el cliente es solvente, aceptar pedido;*
*en caso contrario, rechazar pedido.*
*5. Fin.*

*Ejemplo de Algoritmo 2:*
Realizar la suma de todos los números pares entre 2 y 1.000. El problema consiste en sumar 2 + 4 + 6 + 8 ... + 1.000. Utilizaremos las palabras SUMA y NÚMERO (variables, serán definidas más adelante) para representar las sumas sucesivas (2+4), (2+4+6), (2+4+6+8), etc. La solución se puede escribir con el siguiente algoritmo:

*1. Inicio.*
*2. Establecer SUMA a 0.*
*3. Establecer NÚMERO a 2.*
*4. Sumar NÚMERO a SUMA. El resultado será el nuevo valor de la suma (SUMA).*
*5. Incrementar NÚMERO en 2 unidades.*
*6. Si NÚMERO es menor o igual a 1.000 bifurcar al paso 4;*
*en caso contrario, escribir el último valor de SUMA y terminar el proceso.*
*7. Fin.*

*Metodología de Resolución de Problemas*

La metodología necesaria para resolver problemas mediante programas, se denomina m*etodología de la programación.* Los pasos necesarios a llevar a cabo se ilustran en la *Figura 3.*

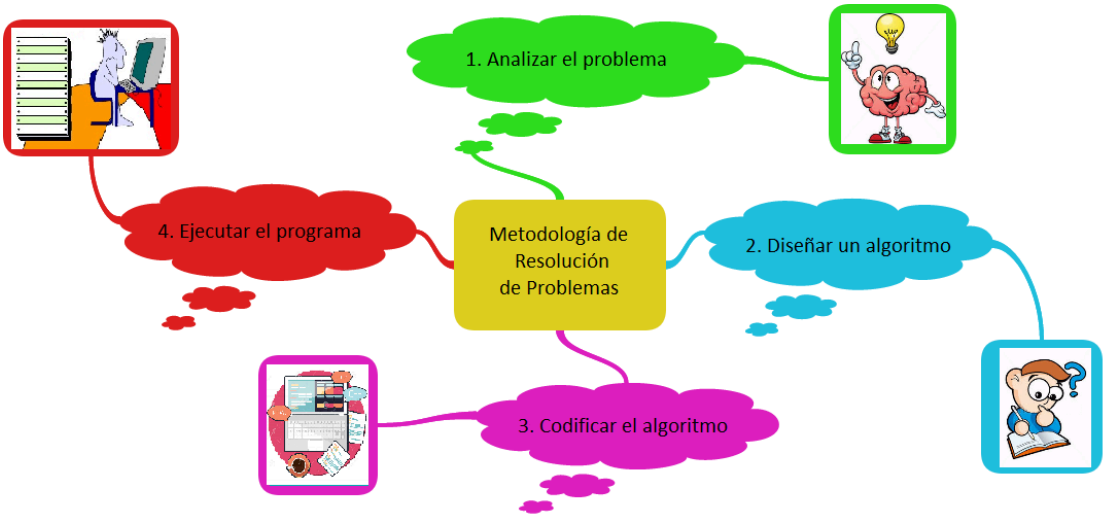


*Figura 3: Metodología de Resolución de Problemas*

A continuación, se describen cada uno de los *pasos en la resolución de un problema*:

a) *Analizar el problema:* Los programas de computadora tienen como finalidad resolver problemas específicos y el primer paso consiste en definir con precisión el problema hasta lograr la mejor comprensión posible. Una forma de realizar esta actividad se basa en *formular claramente el problema*, especificar los *resultados* que se desean obtener, identificar la *información disponible* (datos), determinar las *restricciones* y definir los *procesos* necesarios para convertir los datos disponibles (materia prima) en la información requerida (resultados). Para poder identificar y definir bien un problema es conveniente responder a las siguientes preguntas:

- ¿Qué información es importante?
- ¿Qué información *no* es relevante?
- ¿Cuáles son los datos de entrada?
- ¿Cuál es el resultado (salida) deseado?
- ¿Qué método produce la salida deseada?
- ¿Qué información me falta para resolver el problema?
- Requisitos o requerimientos adicionales y restricciones a la solución.

b) *Diseñar un algoritmo:* En la etapa de análisis del proceso de programación se determina qué hace el programa. En la etapa de diseño se determina cómo hace el programa la tarea solicitada, es decir, se describe la secuencia ordenada de pasos -sin ambigüedades- que conducen a la solución de un problema dado. Los métodos más eficaces para el proceso de diseño se basan en el conocido divide y vencerás. Es decir, la resolución de un problema complejo se realiza dividiendo el problema en subproblemas y a continuación dividiendo estos subproblemas en otros de nivel

más bajo, hasta que pueda ser implementada una solución en la computadora. Este método se conoce técnicamente como *diseño descendente* (top-down) o modular. El proceso de romper el problema en cada etapa y expresar cada paso en forma más detallada se denomina *refinamiento sucesivo*. Las ventajas más importantes del diseño descendente son:

- El problema se comprende más fácilmente al dividirse en partes más simples denominadas módulos.
- Las modificaciones en los módulos son más fáciles.
- La comprobación del problema se puede verificar fácilmente.

En conclusión, el proceso que convierte los resultados del análisis del problema en un diseño modular con refinamientos sucesivos que permitan una posterior traducción a un lenguaje se denomina diseño del algoritmo. El diseño del algoritmo es independiente del lenguaje de programación en el que se vaya a codificar posteriormente. La *Figura 4* muestra como el problema de encontrar la superficie y la longitud de un círculo se puede dividir en tres problemas más simples o subproblemas.

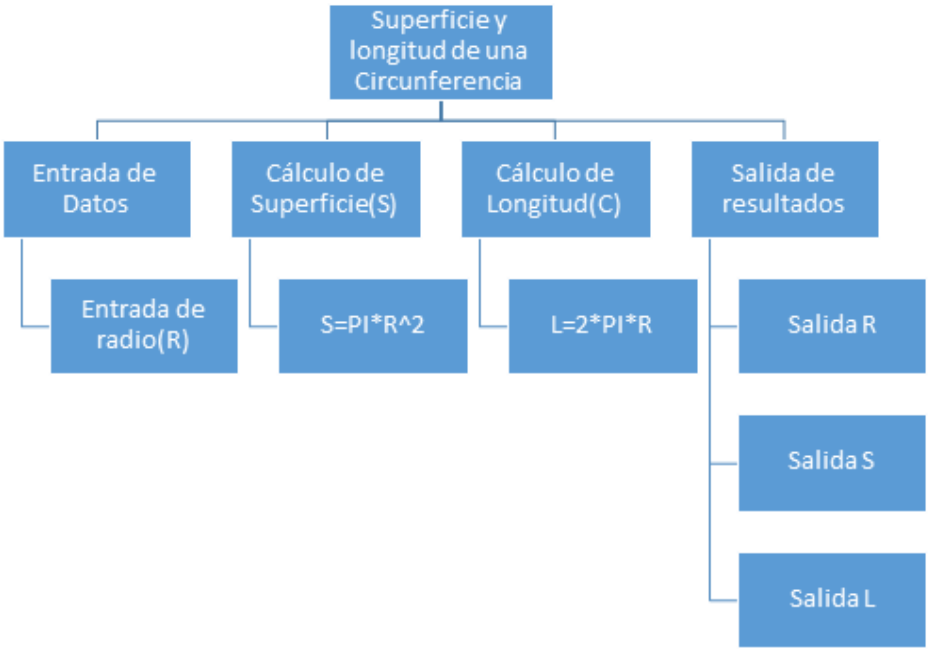


*Figura 4: Refinamiento de un Algoritmo*

Tras los pasos anteriores (diseño descendente y refinamiento por pasos) es preciso representar el algoritmo mediante una determinada herramienta de programación, como por ejemplo un diagrama de flujo o pseudocódigo.

- *Diagrama de Flujo:* Un diagrama de flujo (flowchart) es una de las técnicas de representación de algoritmos más antigua y a la vez más utilizada, aunque su empleo ha disminuido considerablemente. Un diagrama de flujo es la representación gráfica del algoritmo o proceso. En estos diagramas a través del uso de flechas, denominadas líneas de flujo, se indica la secuencia en que se debe ejecutar el algoritmo.

- *Pseudocódigo*: El *pseudocódigo* es una herramienta de programación en la que las instrucciones se escriben en palabras similares al inglés o español, que facilitan tanto la escritura como la lectura de programas. En esencia, el pseudocódigo se puede definir como un lenguaje de especificaciones de algoritmos. El uso de tal lenguaje hace el paso de codificación final (esto es, la traducción a un lenguaje de programación) relativamente fácil. El pseudocódigo se considera un primer borrador, dado que tiene que traducirse posteriormente a un lenguaje de programación. El pseudocódigo no puede ser ejecutado por una computadora. La *ventaja* del pseudocódigo es que, en su uso, en la planificación de un programa, el programador se puede concentrar en la lógica y en las estructuras de control y no preocuparse de las reglas de un lenguaje específico. Es también fácil modificar el pseudocódigo si se descubren errores o anomalías en la lógica del programa, mientras que en muchas ocasiones suele ser difícil el cambio en la lógica, una vez que está codificado en un lenguaje de programación.

c) *Traducir el algoritmo en un lenguaje de programación (codificación):* La codificación es la escritura en un lenguaje de programación de la representación del algoritmo desarrollada en las etapas precedentes. Dado que el diseño de un algoritmo es independiente del lenguaje de programación utilizado para su implementación, el código puede ser escrito con igual facilidad en un lenguaje o en otro. Para realizar la conversión del algoritmo en programa se deben sustituir las palabras reservadas en español por sus homónimos en inglés, y las operaciones/instrucciones indicadas en lenguaje natural por el lenguaje de programación correspondiente.

d) *Ejecución, verificación y depuración del programa por la computadora:* Después de traducir el algoritmo en un lenguaje de programación, el programa resultante debe ser probado y verificado. La verificación de un programa es el proceso de ejecución del programa con una amplia variedad de datos de entrada, llamados datos de prueba, que determinarán si el programa tiene o no errores (“bugs”). Para realizar la verificación se debe desarrollar una amplia gama de datos de prueba: valores normales de entrada, valores extremos de entrada que comprueben los límites del programa y valores de entrada que comprueben aspectos especiales del programa. La depuración es el proceso de encontrar los errores del programa y corregir o eliminar dichos errores.

## 3. Lenguajes de programación

Como se mencionó anteriormente, para que un procesador realice un proceso se le debe suministrar en primer lugar un algoritmo adecuado. El procesador debe ser capaz de interpretar el algoritmo, lo que significa:

- comprender las instrucciones de cada paso,
- realizar las operaciones correspondientes.

Cuando el procesador es una computadora, el algoritmo se ha de expresar en un formato que se denomina programa, ya que el pseudocódigo o el diagrama de flujo no son comprensibles por la computadora, aunque pueda entenderlos cualquier programador. Un programa se escribe en un lenguaje de programación y las operaciones que conducen a expresar un algoritmo en forma de programa se llama programación. Así pues, los lenguajes utilizados para escribir programas de computadoras son los lenguajes de programación y *programadores* son los escritores y diseñadores de programas. El proceso de traducir un algoritmo en pseudocódigo a un lenguaje de

programación se denomina *codificación*, y el algoritmo escrito en un lenguaje de programación se denomina *código fuente*.

En la realidad la computadora no entiende directamente los lenguajes de programación, sino que se requiere un programa que traduzca el código fuente a otro lenguaje que sí entiende la máquina directamente, pero muy complejo para las personas; este lenguaje se conoce como *lenguaje máquina* y el código correspondiente *código máquina*.
Los programas que traducen el código fuente escrito en un lenguaje de programación (como por ejemplo C++ o Java) a código máquina se denominan *traductores*. El proceso de conversión de un algoritmo escrito en pseudocódigo hasta un programa ejecutable comprensible por la máquina, se muestra en la *Figura 5*.

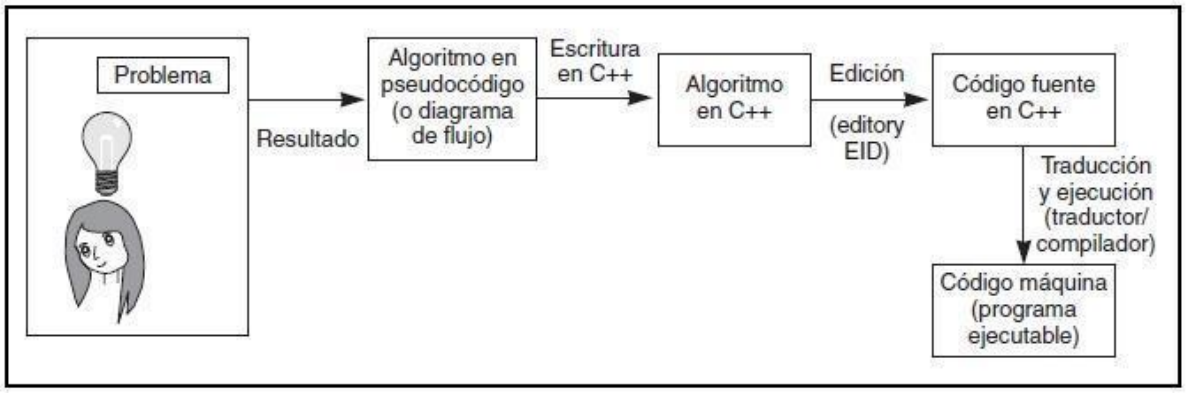


*Figura 5: Proceso de transformación de un algoritmo en pseudocódigo en un programa ejecutable*

Los *lenguajes de programación* se utilizan para escribir programas. Los programas de las computadoras modernas constan de secuencias de instrucciones que se codifican como secuencias de dígitos numéricos que podrán entender dichas computadoras. El sistema de codificación se conoce como *lenguaje máquina* que es el lenguaje nativo de una computadora. Desgraciadamente la escritura de programas en lenguaje máquina es una tarea tediosa y difícil ya que sus instrucciones son secuencias de 0 y 1 (patrones de bit, tales como 11110000, 01110011...) que son muy difíciles de recordar y manipular por las personas. En consecuencia, se necesitan lenguajes de programación "amigables con el programador" que permitan escribir los programas para poder "charlar" con facilidad, lo que será preciso traducir los programas resultantes a lenguajes de máquina antes de que puedan ser ejecutadas por ellas. Cada lenguaje de programación tiene un conjunto o "juego" de instrucciones (acciones u operaciones que debe realizar la máquina) que la computadora podrá entender directamente en su código máquina o bien se traducirán a dicho código máquina.

*Instrucciones básicas de un lenguaje de programación:*

- Instrucciones de entrada/salida. Instrucciones de transferencia de información entre dispositivos periféricos y la memoria central, tales como "leer de..." o bien "escribir en...".

- Instrucciones de cálculo. Instrucciones para que la computadora pueda realizar operaciones aritméticas.
- Instrucciones de control. Instrucciones que modifican la secuencia de la ejecución del programa.

Además de estas instrucciones y dependiendo del procesador y del lenguaje de programación existirán otras que conformarán el conjunto de instrucciones y junto con las reglas de sintaxis permitirán escribir los programas de las computadoras. Los principales tipos de lenguajes de programación son:

a) *Lenguajes máquina:* es el único que entiende directamente la computadora, utiliza el alfabeto binario que consta de los dos únicos símbolos 0 y 1, denominados bits; físicamente, se materializan con tensiones comprendidas entre 0 y 4.0 voltios y entre 4 y 5 voltios, respectivamente. Para representar datos que contengan una información se utilizan una serie de unos y ceros cuyo conjunto indica dicha información.

b) *Lenguajes de bajo nivel (ensambladores):* consiste en un conjunto de mnemónicos que representan instrucciones básicas para los computadores, microprocesadores, microcontroladores y otros circuitos integrados programables. Implementa una representación simbólica de los códigos de máquina binarios y otras constantes necesarias para programar una arquitectura de procesador y constituye la representación más directa del código máquina específico para cada arquitectura legible por un programador. Cada arquitectura de procesador tiene su propio lenguaje ensamblador que usualmente es definida por el fabricante de hardware, y está basada en los mnemónicos que simbolizan los pasos de procesamiento (las instrucciones), los registros del procesador, las posiciones de memoria y otras características del lenguaje.

c) *Lenguajes de alto nivel:* se caracteriza por expresar el algoritmo de una manera adecuada a la capacidad cognitiva humana, en lugar de la capacidad ejecutora de las máquinas. Un lenguaje de alto nivel permite al programador escribir las instrucciones de un programa utilizando palabras o expresiones sintácticas muy similares al inglés. Por ejemplo, en C se pueden construir con ellas instrucciones como:

```
if( numero > 0 ) printf( "El número es positivo" )
```

que traducido al castellano es equivalente a decir que: si el número es mayor que cero, entonces, escribir por pantalla el mensaje: "El número es positivo".

*Traductores de lenguaje: el proceso de traducción de un programa*

El proceso de traducción de un programa fuente escrito en un lenguaje de alto nivel a un lenguaje máquina comprensible por la computadora, se realiza mediante programas llamados “traductores”. Los traductores de lenguaje son programas que traducen a su vez los programas fuente escritos en lenguajes de alto nivel a código máquina. Los traductores se dividen en intérpretes y compiladores.

a) *Intérpretes:* Un intérprete es un traductor que toma un programa fuente, lo traduce y, a continuación, lo ejecuta. El sistema de traducción consiste en: traducir la primera sentencia del programa a lenguaje máquina, se detiene la traducción, se ejecuta la sentencia; a continuación, se traduce la siguiente sentencia, se detiene la traducción, se ejecuta la sentencia y así sucesivamente hasta terminar el programa (ver *Figura 6*).

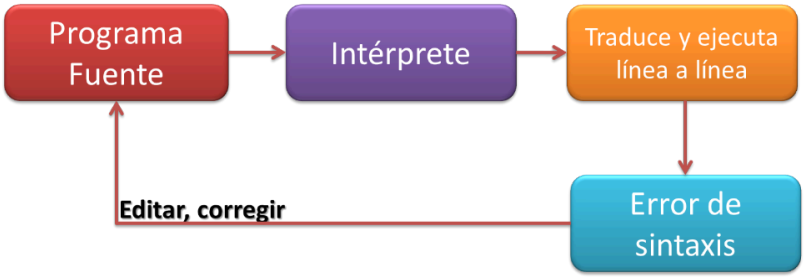


*Figura 6: Intérprete*

b) *Compiladores:* Un compilador es un programa que traduce los programas fuente escritos en lenguaje de alto nivel a lenguaje máquina. La traducción del programa completo se realiza en una sola operación denominada compilación del programa; es decir, se traducen todas las instrucciones del programa en un solo bloque. El programa compilado y depurado (eliminados los errores del código fuente) se denomina programa ejecutable porque ya se puede ejecutar directamente y cuantas veces se desee; sólo deberá volver a compilarse de nuevo en el caso de que se modifique alguna instrucción del programa. De este modo el programa ejecutable no necesita del compilador para su ejecución (ver *Figura 7*).


*Figura 7: Compilador*