

UNIDAD 1 (PUEDEN VER DE ESTE TEXTO O DEL LIBRO DE JOYANES AGUILAR)

SOFTWARE DE UNA COMPUTADORA

INTRODUCCIÓN

El software de un sistema informático está constituido por el conjunto de programas ejecutables en dicho sistema y todo lo relacionado con los mismos. Dentro del software se incluyen: el sistema operativo, las interfaces de usuario, los lenguajes de programación, las herramientas o utilidades, las aplicaciones de cualquier especialidad, tipo o contenido, etc.

Como parte integrante del software de un sistema, se han señalado las herramientas o utilidades aquí puede ser incluido software tan variado como programas de hoja de cálculo, tratamiento de textos, aplicaciones de uso común, los gestores de bases de datos, los paquetes de todo en uno, los paquetes de gestión, así como otros paquetes de utilidades.

Las computadoras tienen la capacidad de realizar muy diversas tareas siempre que tengan el software adecuado. Los ordenadores permiten realizar tareas que antes necesitaban un personal muy especializado en diversos campos (mecanografía, delineación, analistas financieros, programadores, etc.) para poder llevarlas a cabo. Actualmente, la gran mayoría de esas tareas pueden ser realizadas mediante un ordenador personal, el software adecuado y una persona entrenada mínimamente en ese software.

Una primera clasificación del software nos permite diferenciar dos grandes categorías: software de sistema y software de aplicación.

SOFTWARE DE SISTEMA

Llamamos software de sistema al conjunto de programas que se encargan de controlar el funcionamiento de los programas que se ejecutan y de la gestión interna de los recursos físicos de la computadora. Como es natural, el sistema operativo forma parte del software de sistema pero, además, se incluyen aquí el software de programación y el software de diagnóstico y mantenimiento.

Software de programación

Está formado por los programas y utilidades que facilitan la construcción de aplicaciones de usuarios. Aquí incluiríamos a los intérpretes, los compiladores, los montadores, los módulos de gestión de ficheros, los cargadores, etc. Algunos ejemplos de utilidades son:

- Los compiladores traducen un programa escrito en lenguaje de alto nivel a un lenguaje ensamblador.
- Los ensambladores traducen el lenguaje ensamblador a lenguaje máquina, pero todavía no son ejecutables, ya que hay llamadas a módulos que se desconoce donde están.
- Los montadores se encargan de la unión de todos los módulos, generando un nuevo fichero ejecutable.
- El cargador se encarga de llevar el programa ejecutable a memoria y prepararlo para su ejecución.

- El distribuidor carga en el contador de programa la dirección física donde se encuentra la primera instrucción.

- Utilidades de rastreo o depuración de errores. Son utilidades que nos permiten ejecutar los programas de diversas formas (línea a línea, detenerse en alguna línea, etc) para hacer un seguimiento de las variables y así poder encontrar posibles errores.

Software de diagnóstico y mantenimiento

Es el software utilizado por el personal encargado de la puesta a punto de los equipos. Con este software se pretende localizar averías de un periférico o encontrar el mal funcionamiento de un paquete software.

Un ejemplo de prueba de chequeo de la memoria puede consistir en escribir en todas las posiciones de memoria un valor determinado, posteriormente se leen todas estas posiciones y se comprueba donde no coincida el valor leído con el escrito. Esto nos determina las posiciones que se encuentran en un mal estado. Un sistema parecido se puede usar para comprobar la memoria masiva.

SOFTWARE DE APLICACIÓN

El software de aplicación lo forman los programas que controlan el funcionamiento de la computadora para realizar una tarea específica (esta tarea es denominada normalmente aplicación). Dentro de este tipo de software se incluyen el software estándar y el software a medida.

El software estándar o herramientas informáticas hace referencia a aquellas aplicaciones de uso general especialmente diseñadas para su lanzamiento al mercado. Estas aplicaciones pueden ser utilizadas por gran número de usuarios y sobre diferentes sistemas. Algunas de estas aplicaciones de uso común son el tratamiento de textos, las hojas de cálculo, la gestión de base de datos, comunicaciones, gráficos, los paquetes integrados, etc. Por su extensión, desarrollamos este tipo de software en la siguiente sección.

El software a medida está constituido por aquellas aplicaciones específicas que se refieren a actividades más especializadas. En este caso, una aplicación de este tipo es desarrollada para un/unos usuario/s concreto/s y para un sistema específico. Aquí se incluyen los programas realizados por los propios usuarios, una aplicación de control del tráfico en el área de Londres, un sistema experto para el reconocimiento de yacimientos de minerales, un programa para llevar la contabilidad y la gestión de clientes de una empresa concreta, etc.

LENGUAJES DE PROGRAMACION. TRADUCTORES

INTRODUCCIÓN

En los últimos tiempos, el hardware de computadoras ha sufrido un desarrollo vertiginoso. No obstante, el aprovechamiento de los avances tecnológicos no es óptimo si no se dispone del software adecuado. Con este fin, se han diseñado diversos tipos de lenguajes de programación,

unos de propósito general y otros de aplicación específica en alguna de las áreas del ámbito informático.

Un **lenguaje de programación** es un conjunto de símbolos junto a un conjunto de reglas para combinar dichos símbolos que se usan para expresar programas. Los lenguajes de programación, como cualquier tipo de lenguaje, se componen de un léxico (conjunto de símbolos permitidos o vocabulario), una sintaxis (reglas que indican cómo *realizar las construcciones del lenguaje*) y una *semántica* (reglas que permiten determinar el significado de cualquier construcción del lenguaje).

Para que una computadora pueda procesar un programa escrito en un determinado lenguaje de programación, es necesario realizar previamente una *traducción* del programa al lenguaje que entienda dicha computadora según una serie de fases.

LENGUAJE MÁQUINA

El **lenguaje máquina** es el único lenguaje que entiende directamente dicho ordenador. Por esta razón, su estructura está totalmente adaptada a los circuitos de la máquina y muy alejada de la forma de expresión y análisis de los problemas propia de los humanos. Esto hace que la programación en este lenguaje resulte tediosa y complicada, requiriéndose un conocimiento profundo de la arquitectura física del ordenador. Frente a esto, el código máquina hace posible que el programador utilice la totalidad de los recursos que ofrece el ordenador, obteniéndose programas muy eficientes (es decir, que aprovechan al máximo los recursos existentes) en tiempo de ejecución y en ocupación de memoria.

Las principales características del lenguaje máquina son las siguientes:

- Las instrucciones se expresan en el *alfabeto binario* (están codificadas en binario como cadenas de ceros y unos), pudiéndose utilizar códigos intermedios (octal y hexadecimal). Esta característica hace que un programa en lenguaje máquina sea difícil de entender y, como consecuencia, difícil de modificar.
- Los datos se referencian por medio de las direcciones de memoria donde se encuentran (no aparecen nombres de variables o de constantes).
- Las instrucciones realizan operaciones muy simples. El programador debe ingeniárselas para expresar cada una de las operaciones que desea realizar en términos de las instrucciones elementales de las que dispone.
- El lenguaje máquina depende y está íntimamente ligado a la CPU del ordenador. Esto hace que los programas en lenguaje máquina no sean transferibles de un modelo de ordenador a otro (un programa en lenguaje máquina sólo se puede ejecutar en el procesador para el cual está destinado), es decir, que exista baja *portabilidad*.
- En un programa en lenguaje máquina no pueden incluirse comentarios que faciliten la legibilidad del mismo.

Estas limitaciones son resueltas en parte por el lenguaje ensamblador y prácticamente en su totalidad por los lenguajes simbólicos de alto nivel.

LENGUAJE ENSAMBLADOR

El **lenguaje ensamblador** constituye el primer intento de sustitución del lenguaje máquina por uno más cercano al usado por los humanos. Este acercamiento a las personas se plasma en las siguientes aportaciones:

- Uso de una *notación simbólica o nemotécnica* para representar los códigos de operación. De esta forma, se evitan los códigos numéricos, tan difíciles de manejar. Normalmente, los códigos nemotécnicos están constituidos por abreviaturas de las operaciones en inglés (dado el origen anglosajón de los fabricantes). Así, por ejemplo, la suma se representa en la mayoría de los ensambladores por ADD.
- *Direccionamiento simbólico*. En lugar de utilizar direcciones binarias absolutas, los datos pueden ser identificados con nombres como COSTE, TOTAL, X,Y,Z, etc.
- Se permite el *uso de comentarios* entre las líneas de instrucciones, haciendo posible la redacción de programas más legibles.

Aparte de esto, el lenguaje ensamblador presenta la mayoría de los inconvenientes del lenguaje máquina, como son su repertorio muy reducido de instrucciones, el rígido formato de las instrucciones, la baja portabilidad y la fuerte dependencia del hardware. Por otro lado, mantiene la ventaja del uso óptimo de los recursos hardware, permitiendo la obtención de un código muy eficiente.

Este tipo de lenguajes hacen corresponder a cada instrucción en ensamblador una instrucción en código máquina. Esta traducción es llevada a cabo por un programa traductor denominado **ensamblador**.

Para solventar en cierta medida la limitación que supone poseer un repertorio de instrucciones tan reducido, se han desarrollado unos ensambladores especiales denominados macroensambladores. Los lenguajes que traducen los macroensambladores disponen de macroinstrucciones cuya traducción da lugar a varias instrucciones máquina y no a una sola. Por ejemplo, existen macroinstrucciones para multiplicar, dividir, transferir bloques de memoria principal a disco, etc.

Dado que el lenguaje ensamblador está fuertemente condicionado por la arquitectura del ordenador que soporta, los programadores no suelen escribir programas de tamaño considerable en ensamblador. Más bien usan este lenguaje para afinar partes importantes de programas escritos en lenguajes de más alto nivel. El lenguaje ensamblador sigue siendo importante, ya que da al programador el control total de la máquina, y como resultado genera un código compacto, rápido y eficiente.

LENGUAJES DE ALTO NIVEL

Los esfuerzos encaminados a hacer la labor de programación independiente de la máquina dieron como resultado la aparición de los lenguajes de programación de alto nivel. Estos lenguajes, más evolucionados, usan palabras y frases relativamente fáciles de entender y proporcionan también facilidades para expresar alteraciones del flujo de control de una forma bastante sencilla e intuitiva.

Las características fundamentales de los lenguajes de alto nivel son las siguientes:

- Son *independientes de la arquitectura física* de la computadora. Esto permite utilizar los mismos programas en computadoras de arquitecturas diferentes (portabilidad) y, además, no es necesario conocer el hardware específico de la máquina.
- Al igual que ocurre con el lenguaje ensamblador, la ejecución de un programa en lenguaje de alto nivel requiere de una traducción del mismo al lenguaje máquina de la computadora donde va a ser ejecutado.
- Por lo general, *una sentencia en un lenguaje de alto nivel da lugar, al ser traducida, a varias instrucciones en lenguaje máquina*. Además, determinados componentes del programa pueden ser referenciados con un nombre simbólico para facilitar la comprensión de las personas.
- Utilizan *notaciones cercanas a las usadas por las personas en un determinado ámbito*. Se pretende la mayor aproximación posible al lenguaje natural o al lenguaje algebraico. Para ello, las instrucciones vienen expresadas mediante texto, es posible introducir comentarios en las líneas de las instrucciones y la escritura de programas está usualmente basada en reglas parecidas a las humanas. Todas estas características dan lugar a programas más legibles y de más fácil modificación y puesta a punto.
- Se suelen incluir *instrucciones potentes de uso frecuente* que son ofrecidas por el lenguaje de programación. Por ejemplo, se suelen ofrecer funciones matemáticas de uso común (seno, coseno, conversión de entero a real, etc.), operadores específicos de entrada/salida, operadores de tratamiento de cadenas de caracteres, etc.
- Existe gran cantidad de lenguajes de alto nivel actualmente en uso, y se han desarrollado diferentes versiones de algunos de ellos. Esta heterogeneidad constituye el principal problema que presentan estos lenguajes.
- Los lenguajes de alto nivel, a diferencia de los lenguajes máquina y ensamblador, no permiten aprovechar completamente los recursos internos de la máquina.

Todas estas características ponen de manifiesto un acercamiento a las personas y un alejamiento de la máquina. Por esta razón, los programas escritos en lenguaje de alto nivel no pueden ser directamente interpretados por la computadora, siendo necesario realizar previamente su traducción a lenguaje máquina. Para ello, se han de utilizar unos programas traductores (previamente desarrollados para cada computadora) que se encarguen de realizar dicho proceso de traducción. Hay dos tipos de traductores de lenguajes de alto nivel que vamos a considerar: los *compiladores* y los *intérpretes*.

TRADUCTORES. COMPILADORES E INTÉRPRETES

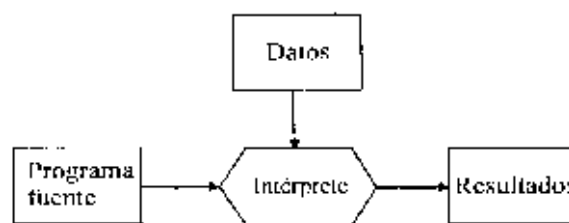
Como la computadora puede interpretar y ejecutar únicamente el código máquina, existen programas especiales, denominados traductores, que traducen programas escritos en un lenguaje de programación al lenguaje máquina de la computadora. Un **traductor** es un metaprograma que toma como entrada un programa (o parte de un programa) escrito en lenguaje simbólico denominado *programa fuente* y proporcionan como salida otro programa semánticamente equivalente, escrito en un lenguaje comprensible por el hardware de la

computadora denominado *programa objeto*. Aquí veremos dos tipos de traductores, los compiladores y los intérpretes, los cuales representan dos aproximaciones muy distintas a la tarea de permitir el funcionamiento de los programas escritos en un determinado lenguaje de programación de alto nivel.

Un **compilador traduce** completamente un programa fuente, generando un programa objeto (semánticamente equivalente) escrito en lenguaje máquina. Como parte importante de este proceso de traducción, el compilador informa al usuario de la presencia de errores en el programa fuente, pasándose a la creación del programa objeto sólo en el caso de que no hayan sido detectados errores en el programa fuente (por lo general, suele cancelarse la compilación cuando se detectan errores). El programa fuente suele estar contenido en un fichero, y el programa objeto puede almacenarse como otro fichero en memoria masiva para ser ejecutado posteriormente, sin necesidad de volver a realizar la traducción. Una vez traducido un programa, su ejecución es independiente de su compilación.

Un **intérprete** permite que un programa fuente escrito en un determinado lenguaje vaya traduciéndose y ejecutándose directamente sentencia a sentencia por la computadora. El intérprete capta una sentencia fuente, la analiza y la interpreta, dando lugar a su ejecución inmediata. Por consiguiente, en este caso no se crea ningún archivo o programa objeto almacenable en memoria masiva para posibles ejecuciones futuras.

Si utilizamos un intérprete para traducir un programa cada vez que necesitamos ejecutar el programa tenemos que volver a analizarlo, ya que no se genera un archivo objeto. En cambio, con un compilador, aunque sea más lenta, la traducción sólo debe realizarse una vez.

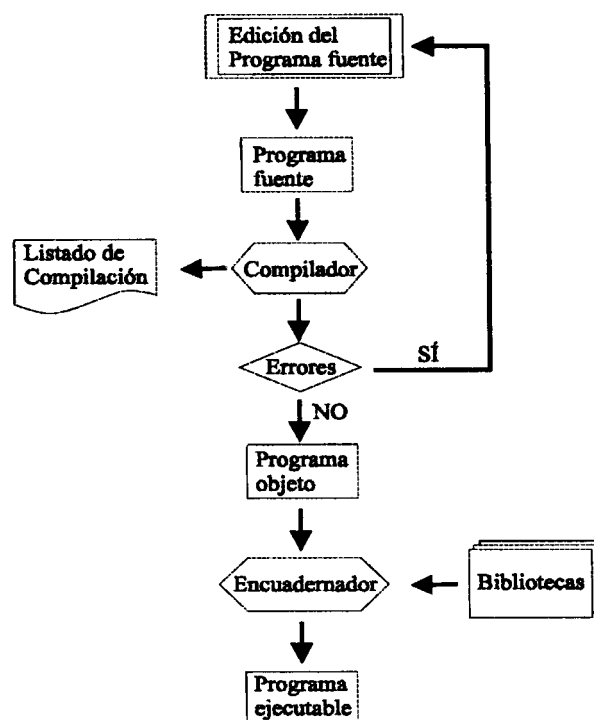


La principal ventaja de los intérpretes frente a los compiladores es que resulta más fácil localizar y corregir errores de los programas (depuración de programas), ya que la ejecución de un programa bajo un intérprete puede interrumpirse en cualquier momento para conocer los valores de las distintas variables y la instrucción fuente que acaba de ejecutarse. Por este motivo, los intérpretes resultan más pedagógicos para aprender a programar, ya que el alumno puede detectar y corregir más fácilmente sus errores (son famosos los intérpretes del lenguaje de programación BASIC, un lenguaje (muy usado para enseñar a programar)).

EL PROCESO DE COMPILACIÓN

Por ser los compiladores el tipo de traductor más utilizado en la actualidad desarrollamos el proceso de compilación, que consiste en la traducción de un programa fuente, escrito en lenguaje de alto nivel, a su correspondiente programa objeto, escrito en lenguaje máquina, dejándolo listo para la ejecución con poca o ninguna preparación adicional.

Previamente al proceso de compilación de un programa, se crea el mencionado programa fuente utilizando cualquier aplicación disponible con capacidades de edición de textos. Para llevar a cabo la compilación de un programa, dicho programa debe residir en memoria principal simultáneamente con el compilador. El resultado de 1ª compilación puede dar lugar a la aparición de errores, en cuyo caso no se genera el programa objeto, sino que se realiza lo que se denomina **listado de compilación**, que no es más que un informe indicando la naturaleza y situación de los errores en el programa fuente. El listado de compilación permite ver los errores detectados por el compilador para volver al programa editor, corregirlos y empezar de nuevo el proceso. Una vez finalizada la compilación y obtenido el programa objeto, es necesario someterlo a un proceso de montaje donde se enlazarán los distintos módulos que lo componen, en caso de tratar con programas que poseen subprogramas (los cuales pueden ser compilados separadamente). Además, se incorporan las denominadas **rutinas de biblioteca** en caso de solicitarlas el propio programa. Este proceso de montaje es realizado por un programa denominado **montador o encuadernador**, que también recibe el nombre de *editor de enlace* o *linker*.



La compilación es un proceso complejo que consume a veces un tiempo muy superior a la propia ejecución del programa. Este proceso consta en general de dos etapas fundamentales: la etapa de análisis del programa fuente y la etapa de síntesis del programa objeto. Cada una de estas etapas conlleva la realización de varias fases.

El compilador utiliza internamente una **tabla de símbolos** para introducir determinados datos que necesita. Esta tabla interviene prácticamente en todas las fases del proceso de compilación; así mismo, el compilador posee un módulo de **tratamiento de errores** que permite determinar las reacciones que se deben producir ante la aparición de cualquier tipo de error.

Los **tipos de errores** que puede tener un programa son los siguientes:

- *Errores lexicográficos.* Son errores que se producen por la aparición de cadenas que no se ajustan al patrón de ningún símbolo elemental del lenguaje. Estos errores son detectados por el escáner en el tiempo en que se ejecuta el análisis lexicográfico.

- *Errores sintácticos.* Son aquellos errores que aparecen cuando se violan las reglas de sintaxis del lenguaje.

- *Errores semánticos.* Son detectados generalmente durante la fase de generación de código intermedio y se detectan cuando aparece una sentencia sintácticamente correcta, pero que carece de sentido dentro del contexto del programa fuente.

- *Errores lógicos.* Son los debidos a la utilización de un algoritmo o expresión incorrecta para el problema que se trata de resolver. Se detectan en la fase de pruebas de un programa por medio de la utilización de una estrategia de pruebas bien estudiada.

- *Errores de ejecución.* Son errores relacionados con desbordamientos, operaciones matemáticamente irresolubles, etc. Ejemplos de este tipo de errores son: división por 0, cálculo de la raíz cuadrada de un número negativo, leer en un fichero no abierto o sin información, salir o exceder el rango de una tabla, bucles infinitos, etc.

En ocasiones, se indican al programador determinados errores que pueden existir, pero que no perjudican al resto del proceso de compilación e incluso pueden permitir el funcionamiento del programa final. Estos mensajes de error se denominan **advertencias o warnings**. Un ejemplo típico de warning puede venir dado por la declaración de una variable que no se usa en ningún bloque del programa.

CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN

Los lenguajes de programación se pueden clasificar de acuerdo a diversos criterios. El criterio más simple que se puede considerar hace referencia a la **proximidad** del lenguaje con la máquina o con el lenguaje natural. De acuerdo a este criterio, existen tres niveles de lenguajes de programación:

- Lenguajes de bajo nivel: Lenguajes máquina.
Lenguajes de nivel medio: Lenguajes ensambladores y macroensambladores.
- Lenguajes de alto nivel: A los que ya hemos hecho referencia.

Dado que los lenguajes de programación, en cierto modo, han sufrido un desarrollo paralelo a la evolución de las computadoras, podemos clasificar los lenguajes de programación atendiendo a su **desarrollo** histórico desde el comienzo de las computadoras. Esta clasificación distingue cinco generaciones de lenguajes:

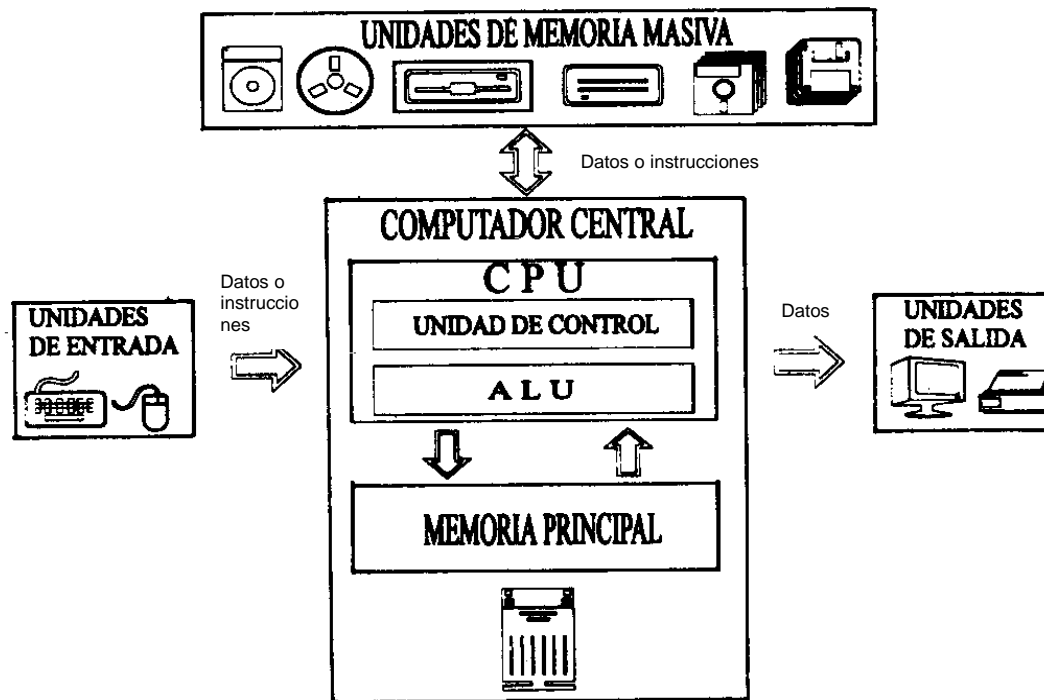
- Primera generación: Lenguajes máquina.
- Segunda generación: Ayudas a la programación como son los ensambladores.
- Tercera generación: Lenguajes de alto nivel imperativos que siguen vigentes en la actualidad, como PASCAL, MODULA-2, FORTRAN, COBOL, C y ADA.

- Cuarta generación (4G): Lenguajes o entornos de programación orientados básicamente a aplicaciones de gestión y bases de datos, como SQL, NATURAL, etc.

- Quinta generación: Lenguajes orientados a aplicaciones en inteligencia artificial, como LISP y PROLOG.

HARDWARE: ESTRUCTURA FUNCIONAL DE UNA COMPUTADORA

El hardware de un sistema informático está compuesto por todos los elementos del mismo con entidad física, es decir, los cables, los circuitos, los dispositivos electromecánicos, etc. Aquí se presenta una clasificación, desde un punto de vista funcional, de los componentes de un ordenador. Un esquema de la estructura de un ordenador típico, se presenta en la siguiente figura:



La computadora central

Es el elemento más importante de la computadora, ya que maneja todo el procesamiento, coordinando y realizando todas las operaciones del sistema informático. Podemos distinguir, a su vez, dos unidades funcionales dentro de la computadora central: la unidad de memoria principal y la unidad central de procesamiento (CPU):

- **Memoria principal, central o interna:** Es el elemento encargado de almacenar los programas y los datos necesarios para que el sistema informático lleve a cabo alguna tarea. Para que un programa pueda ser ejecutado en una computadora, al menos parte del mismo debe encontrarse en memoria principal, junto con los datos que deban ser procesados. Estas memorias presentan gran rapidez y se componen de celdas direccionadas, de forma que cada operación de lectura o escritura en memoria exige la especificación de la dirección sobre la cual se va a realizar dicha operación. Existen dos tipos de memoria principal: la memoria RAM, que permite realizar tanto operaciones de lectura como de escritura y es volátil (si se desconecta el ordenador, se pierde toda la información almacenada), y la memoria ROM, que sólo permite lecturas y es permanente (no necesita ser alimentada con corriente para mantener la información almacenada).

- Unidad central de procesamiento (CPU): También denominada procesador, es el elemento encargado del control y ejecución de las operaciones del sistema. Se puede considerar como el cerebro del ordenador y está compuesto, a su vez, de dos unidades:

- La unidad de control: Es el elemento encargado de coordinar todas las actividades de la computadora. Para ello, se comunica con todas las demás unidades e interpreta y ejecuta ordenadamente las instrucciones del programa en curso.

- La unidad aritmético-lógica (ALU): Está constituida por los circuitos electrónicos necesarios para la realización de operaciones elementales de tipo aritmético (suma, resta, multiplicación, etc.) y lógico (comparaciones, operación OR, operación AND, etc.).

Unidades de entrada

Son aquellos dispositivos encargados de aceptar datos de entrada e instrucciones del exterior y transformarlos en señales binarias eléctricas susceptibles de ser procesadas directamente por el ordenador. Ejemplos típicos de unidades de entrada son el teclado y el mouse (ratón).

ESTRUCTURA DE UNA COMPUTADORA

CONEXIÓN ENTRE LAS UNIDADES DE UNA COMPUTADORA

Las unidades que forman la computadora central (memoria, unidad de control y unidad aritmético-lógica) se encuentran alojadas en lo que se denomina placa base. A través de ella, se ponen en contacto las distintas partes de un ordenador.

La conexión entre los elementos de una computadora se realiza a través de buses (conjunto de hilos que proporcionan un camino para el flujo de datos entre los distintos elementos y que transmiten simultáneamente información en paralelo).

Los buses que interconectan las distintas unidades funcionales de un ordenador pueden ser de tres tipos:

- Bus de datos: Transporta los datos que se transfieren entre unidades. Suele ser bidireccional, es decir, los mismos hilos se utilizan para transmitir información hacia adentro o hacia afuera de una unidad, pero siempre en instantes diferentes. Conviene matizar la diferencia entre el bus de datos interno y el bus de datos externo:

- *Bus de datos interno*: Se utiliza para transferir datos entre los elementos de la computadora central (CPU + memoria principal).

- *Bus de datos externo*: Se puede considerar como una prolongación del bus de datos interno. Pone en comunicación el procesador con el resto de las unidades (periféricos).

- Bus de direcciones: Transporta la dirección de la posición de memoria o del periférico que interviene en el tráfico de información (de dónde procede el dato o a dónde se dirige). Permite la comunicación entre el procesador y las celdas de la memoria RAM. Cuando el procesador quiere leer el contenido de una celda de memoria, envía por el bus de direcciones la dirección de la celda que quiere leer, recibiendo a través del bus de datos el contenido de la misma. El tamaño de este bus define la cantidad de memoria RAM que la CPU puede gestionar

(con 10 bits se pueden direccionar 2^{10} Bytes = 1.024 Bytes = 1KBytes; con 16 bits $\rightarrow 2^{16}$ Bytes = 65.536 Bytes = 65 KBytes; con 32 bits $\rightarrow 4.294.967.296$ = 4 GBytes).

- Bus de control: Contiene hilos que transportan las señales de control y las señales de estado, indicando la dirección de la transferencia de datos, controlando la temporización de eventos durante la transferencia, transmitiendo las señales de interrupción, etc.

MEMORIA

Tipos de memoria

La memoria es la unidad donde se almacenan tanto los datos como las instrucciones. Existen dos tipos básicos de memoria, diferenciados principalmente por su velocidad:

- Memoria principal, central o interna: Es la que actúa a mayor velocidad, estando ligada directamente a las unidades más rápidas de la computadora. Para que un programa pueda ser ejecutado, debe estar almacenado en la memoria principal. Está formada por multitud de celdas o posiciones (palabras de memoria) de un determinado número de bits y numeradas de forma consecutiva. A la numeración de las celdas se le denomina dirección de memoria y mediante esta dirección se puede acceder de forma directa a cualquiera de ellas, independientemente de su posición; por ello, se dice que la memoria principal es una memoria de acceso directo o memoria accesible por dirección.

- Memoria masiva auxiliar, secundaria o externa: Trata de solventar las deficiencias de la memoria principal en cuanto a volatilidad y pequeña capacidad de esta última. Aunque la memoria interna es muy rápida, no tiene gran capacidad para almacenar información. Para guardar información de forma masiva, se utiliza la memoria auxiliar (discos magnéticos, cintas magnéticas, discos ópticos, etc.). Además, la información almacenada en memoria secundaria permanece indefinidamente hasta que el usuario expresamente la borra. Otra ventaja de este tipo de memoria es el precio. En la memoria externa el coste por bit es notablemente inferior que en la memoria interna.

La memoria interna está formada por dos tipos de memoria:

- La memoria ROM (*Read Only Memory* - Memoria de sólo lectura): En la que sólo se permite leer y es permanente, es decir, al desconectar el ordenador, la información no se pierde. Algunos chips de ROM tienen su contenido grabado permanentemente desde el momento en que se fabricaron. Otros están inicialmente en blanco y pueden grabarse con el equipo apropiado. Estas son las memorias programables de sólo lectura o PROM (*Programmable Read Only Memory*). Algunas PROM pueden borrarse para programarse de nuevo empleando el equipo apropiado para este propósito. Éstas son las memorias programables de sólo lectura que pueden borrarse o EPROM (*Erasable Programmable Read Only Memory*). En cualquiera de estos casos, los chips de ROM, una vez instalados en un ordenador, sólo pueden leerse. Las instrucciones y los datos de la ROM permanecen allí una vez que se apaga el ordenador. Cualquier intento de escribir en la ROM no causa ningún efecto, excepto provocar un error que será detectado por el sistema operativo.

- La memoria RAM (Random Access Memory - Memoria de acceso aleatorio): En la que se puede leer y escribir. Esta memoria es volátil, al desconectar el ordenador la información almacenada en la RAM desaparece, de forma que al volver a conectar la máquina, la zona de memoria RAM se encuentra vacía.

Un factor importante para medir la potencia de la memoria es la velocidad de respuesta. Se tienen tres parámetros relacionados con la velocidad:

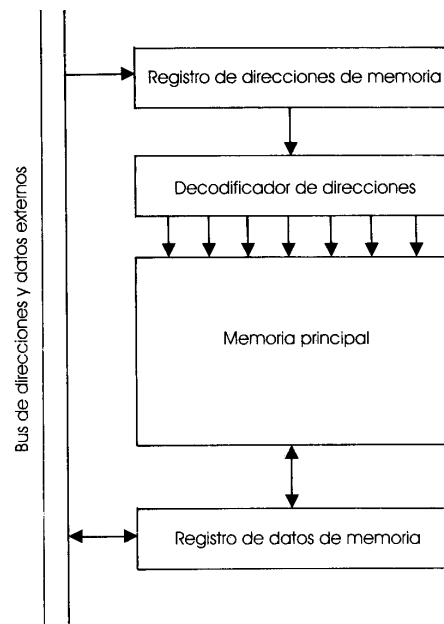
- *Tiempo de acceso*, t : Es el tiempo máximo que se tarda en leer o escribir el contenido de una posición de memoria.
- *Tiempo de ciclo*, t_c : Es el tiempo mínimo entre dos lecturas consecutivas.
- *Ancho de Banda*, AB: Es el número de palabras que se transfieren entre memoria y CPU por unidad de tiempo: $AB = 1/t_c$

Esquema general de una unidad de memoria

La principal función de la unidad de memoria consiste en gestionar los procesos que se encargan de almacenar y recuperar la información. El esquema general de una unidad de memoria es el siguiente:

- *Registro de dirección de memoria*: Antes de realizar una operación de lectura/escritura (L/E), se ha de colocar en este registro la dirección de la celda que va a intervenir en la operación. Dependiendo del número de bits que contenga el registro de dirección, se tendrá una determinada capacidad de memoria (si el registro de dirección es de 8 bits, se podrán codificar hasta $2^8=256$ direcciones de memoria distintas).
- *Decodificador de dirección o selector de memoria*: Se activa cada vez que se produce una orden de L/E, conectando la celda de memoria, cuya dirección se encuentra en el registro de dirección, con el registro de datos y posibilitando la transferencia de los datos en un sentido u otro.
- *Registro de datos*: En él se almacena el dato que se ha leído de memoria o el dato que se va a escribir en memoria.

También existen líneas de control mediante las cuales se transmiten órdenes procedentes de la unidad de control (señal de escritura/lectura, de funcionamiento, de estado).



Secuencia de pasos para leer/escribir un dato

Para la lectura de un dato almacenado en memoria, se siguen estos pasos:

- Se pasa la dirección al registro de dirección.
- Mediante el decodificador se accede a la dirección.
- Se pasa el dato que está en esa dirección al registro de datos.

Para la escritura de un dato en memoria, se siguen estos pasos:

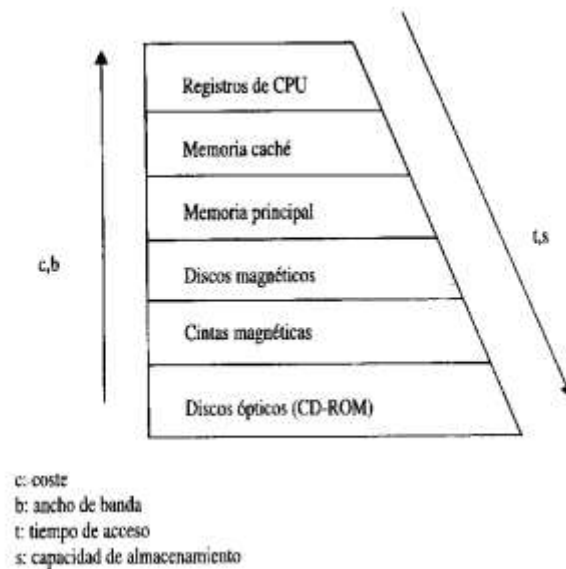
- Se transfiere la dirección en la que se va a escribir al registro de dirección.
- Se transfiere el dato al registro de datos.
- Se decodifica la dirección.
- Se pasa el contenido del registro de datos a la dirección que contiene el registro de dirección.

Jerarquía de memoria

Para que un programa pueda ser ejecutado, debe encontrarse en memoria principal. Puede ocurrir que el tamaño del programa sea mayor que el de la propia memoria principal. En estos casos se utiliza la técnica llamada memoria virtual, que consiste en guardar el programa y sus datos en memoria masiva y mantener en memoria principal únicamente la parte de ellos que está implicada en ese momento en la ejecución.

Otro problema que se plantea es que la CPU capta instrucciones y datos de la memoria principal, almacenando en ella los resultados de las operaciones. Sin embargo, la velocidad a la que opera la CPU es del orden de 10 veces superior a la de la memoria principal. Con esto se desperdiciaría mucho tiempo de CPU esperando que la memoria principal realice su función. En realidad, esto no suele ser así, sino que se introduce entre la CPU y la memoria principal una memoria llamada caché. La memoria caché es una pequeña memoria rápida que se coloca entre la memoria principal y la CPU, de forma que esta última se comunica directamente con ella y no con la

memoria principal. El problema de las memorias caché es que son más caras y tienen menos capacidad.



Las prestaciones de una memoria, sea del tipo que sea, se miden mediante cuatro parámetros:

- Capacidad de almacenamiento, s
- Tiempo de acceso, t
- Ancho de banda, b
- Costo, c

En general, en una memoria se cumple que a mayor velocidad, se tiene un mayor ancho de banda, mayor costo y menor capacidad. Así, los registros de la CPU son los más rápidos y los que tienen mayor ancho de banda, sin embargo son los más caros y los que tienen menor capacidad. A medida que descendemos en la pirámide de la figura, va aumentando el tiempo de acceso (la velocidad es menor) y la capacidad, y disminuyen el ancho de banda y el precio.

UNIDAD CENTRAL DE PROCESAMIENTO

La unidad central de procesamiento, también denominada procesador central o CPU (*Central Processing Unit*), es el verdadero cerebro de la computadora. Su misión consiste en controlar y coordinar o realizar todas las operaciones del sistema. Para ello, extrae, una a una, las instrucciones del programa ubicado en memoria principal, las analiza y emite las órdenes para su completa realización. Físicamente está formada por circuitos de naturaleza electrónica que se encuentran integrados en un chip denominado procesador. Funcionalmente, la unidad de procesamiento central está constituida por dos elementos: la unidad aritmético-lógica y la unidad de control.

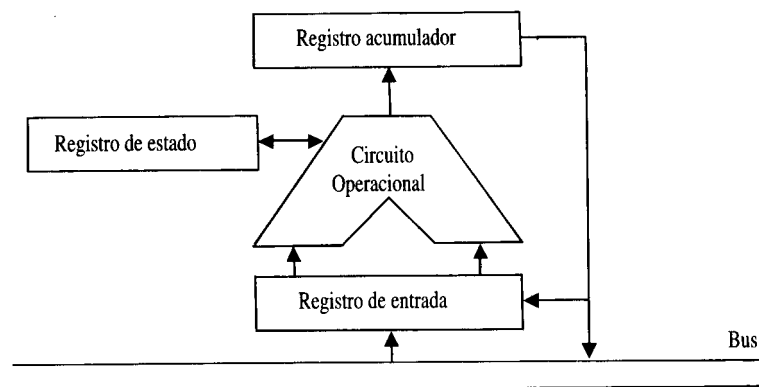
Unidad aritmético-lógica (unidad de procesamiento)

La ALU (*Arithmetic Logic Unit*) es la unidad encargada de realizar las operaciones elementales de tipo aritmético y lógico. Para comunicarse con otras unidades utiliza el bus de datos. La

operación a realizar por la ALU (suma, resta, desplazamientos, comparaciones, etc.) se decide mediante señales de control enviadas desde la unidad de control.

Los elementos que componen la ALU son los siguientes:

- *Circuito operacional (COP)*: Formado por los circuitos necesarios para la realización de las operaciones con los datos procedentes del registro de entrada. También acepta como entrada órdenes para seleccionar el tipo de operación que debe realizar.
- *Registro de entrada (RE)*: Contiene los datos u operandos que intervienen en una instrucción antes de que se realice la operación por parte del circuito operacional. También se emplea como almacenamiento de resultados intermedios o finales de las operaciones.
- *Registro de estado (RS)*: Engloba un conjunto de biestables (indicadores) en los que se deja constancia de condiciones que se dieron en la última operación realizada y que habrán de ser tenidas en cuenta en operaciones posteriores (indicadores de signo, de cero, de desbordamiento, etc.). Al registro de estado también se le conoce con el nombre de *palabra de estado*.
- *Registro acumulador (RA)*: Contiene los datos que se están tratando en cada momento. Almacena los resultados de las operaciones realizadas por el circuito operacional. Está conectado con los registros de entrada para realimentación en el caso de operaciones encadenadas. También tiene una conexión directa con el bus de datos para envío de resultados



a la memoria principal o a la UC.

Esquema de la unidad aritmético-lógica

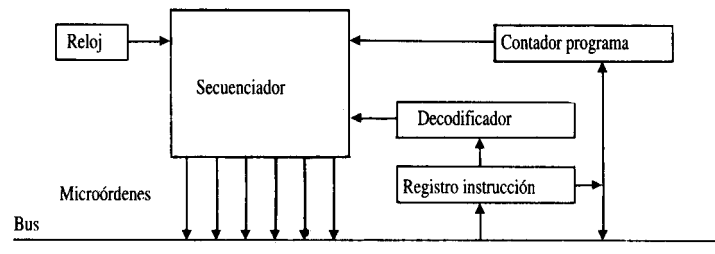
Unidad de control

La unidad de control (UC) se encarga de administrar todos los recursos de la computadora, controlando y dirigiendo la información a las distintas unidades en el momento adecuado mientras el procesador ejecuta cada una de las instrucciones de un programa. De forma más específica, las funciones de la UC son:

- Controlar la secuencia en que se ejecutan las instrucciones.
- Controlar el acceso del procesador (CPU) a la memoria principal.
- Regular las temporizaciones de todas las operaciones que ejecuta la CPU.
- Enviar señales de control y recibir señales de estado del resto de las unidades.

Para realizar estas funciones, la unidad de control consta de los siguientes elementos :

- Contador de programa (CP): Contiene en cada momento la dirección de memoria donde se encuentra la instrucción siguiente a ejecutar. Al iniciar la ejecución de un programa toma la dirección de su primera instrucción. Incrementa su valor en uno de forma automática cada vez que concluye una instrucción, salvo que la instrucción que esté ejecutando sea de salto o de ruptura.
- Registro de instrucción (RI): Dedicado a memorizar temporalmente la instrucción que la UC está interpretando o ejecutando en ese momento. El programa que se está ejecutando reside en memoria principal y la UC va buscando y captando las instrucciones secuencialmente para interpretarlas y generar las órdenes de ejecución. La captación de una instrucción implica leerla en la memoria y almacenarla en el registro de instrucción. La instrucción que se está ejecutando lleva consigo un código de operación (COP) y unos operandos o la dirección de los mismos.
- Decodificador (D): Es el que interpreta realmente la instrucción. Se encarga de extraer el código de operación de la instrucción en curso, lo analiza y emite las señales necesarias al resto de los elementos para su ejecución a través del secuenciador.
- Reloj (R): Proporciona una sucesión de impulsos eléctricos o ciclos a intervalos constantes que marcan los instantes en que han de comenzar los distintos pasos de que consta cada instrucción.
- Secuenciador (S): En este dispositivo se generan órdenes muy elementales (microórdenes), que sincronizadas por el reloj hacen que se vaya ejecutando poco a poco la instrucción que está cargada en el registro de instrucción.



Esquema de la unidad de control (extraído de Alcalde)

FUNCIONAMIENTO DE LAS COMPUTADORAS

Para que un programa pueda ser ejecutado por un ordenador, éste ha de estar almacenado en memoria principal. La unidad central de proceso tomará una a una sus instrucciones e irá realizando las tareas necesarias para completar la ejecución del programa. Se denomina ciclo de instrucción al conjunto de acciones que se llevan a cabo en la realización de una instrucción. Se compone de dos fases:

- Fase de búsqueda o captación: Se transfiere la instrucción que corresponde ejecutar desde la memoria principal a la unidad de control.
- Fase de ejecución: Consiste en la realización de todas las acciones que conlleva la propia instrucción.

Para iniciar la ejecución de un programa, se ubica en el contador de programa (CP) la dirección de memoria donde comienza dicho programa. La unidad de control envía una microorden para que el contenido del CP (la dirección de la primera instrucción) sea transferido al registro de dirección de memoria. El decodificador de memoria interpretará la dirección conectando la celda de memoria indicada en el registro de dirección de memoria con el registro de datos de memoria. Después de un tiempo determinado (tiempo de acceso a memoria), aparecerá en el registro de datos de memoria el contenido de la dirección indicada, es decir, la instrucción que va a ser procesada. A través del bus se transfiere la instrucción desde el registro de datos de memoria al registro de instrucción de la unidad de control (RI). A continuación, el decodificador de la unidad de control procede a interpretar la instrucción que acaba de llegar al RI e informa al secuenciador. Por último, el CP se incrementará automáticamente en uno, de tal forma que quede apuntando a la siguiente instrucción del programa en memoria. Si la instrucción en ejecución es de ruptura de secuencia (de salto o bifurcación), el CP se cargará con la dirección que corresponda. Hasta este punto se ha considerado sólo la fase de búsqueda, que es común a todas las instrucciones. Después tiene lugar la fase de ejecución, que es específica del código de operación de cada instrucción. El secuenciador envía una microorden a la ALU para que ejecute la operación de que se trate, almacenándose el resultado de la operación en el registro acumulador. Una vez concluida la ejecución de la instrucción en curso, la unidad de control vuelve a repetir el ciclo completo para cada una de las instrucciones que forman el programa en ejecución; es decir, capta una nueva instrucción (cuya dirección se encuentra en el CP y la transfiere al RI) y después la decodifica y la ejecuta. Este ciclo se repite iterativamente hasta que concluye la ejecución del programa.

PERIFÉRICOS

Las funciones principales de un ordenador son relativas a la entrada, proceso y salida de información, pudiendo resumirse en procesamiento y comunicaciones. Estas funciones son realizadas por unidades especializadas, encomendándose las comunicaciones al equipo periférico (dispositivos que relacionan el ordenador con el mundo exterior) y el proceso a la computadora central.

DEFINICIÓN Y OBJETIVOS

El núcleo principal de una computadora es la CPU, pero para que ésta funcione correctamente es necesario que los datos e informaciones estén soportados en un medio físico al que el propio ordenador tenga acceso. Estos elementos se denominan soportes de información. Un **medio o soporte de información** es un material físico empleado para almacenar datos de forma que la computadora pueda manejarlos o proporcionarlos a las personas de una manera inteligible (papel de impresora, disco magnético, etc.).

Asimismo, es necesario disponer de dispositivos conectados a la computadora capaces de leer la información en estos soportes o de escribirla en ellos, según se trate de realizar operaciones de lectura o escritura. A estos dispositivos se les denomina periféricos o dispositivos de entrada

y/o salida (dispositivos de E/S). Un **periférico** es una máquina empleada para transferir datos desde o hacia un determinado medio de información, generalmente para su almacenamiento o recuperación (impresora, unidad de disco magnético, etc.).

Se denomina periférico tanto a las unidades o dispositivos a través de los cuales la CPU se comunica con el mundo exterior, como a los sistemas que almacenan o archivan información.

Los periféricos que se pueden conectar a un ordenador se clasifican según las funciones que van a realizar en los siguientes grupos:

- **Unidades de entrada:** A través de las cuales poder dar a la computadora los programas que queremos que ejecute y los datos correspondientes.
- **Unidades de salida:** Con las que obtenemos los resultados de los programas ejecutados.
- **Unidades de memoria masiva auxiliar:** Que faciliten el funcionamiento y utilización del ordenador.

En algunas ocasiones, un mismo periférico incluye unidades de entrada y de salida. En este caso se denomina unidad mixta.

Los objetivos que deben cumplir los periféricos son los siguientes:

- Servir de medio de comunicación eficaz entre el usuario y la computadora, de forma que los datos de salida sean comprensibles para las personas y los datos e instrucciones de entrada lo sean para la computadora.
- Permitir el almacenamiento de informaciones necesarias para ser procesadas o que interesa guardar durante un período de tiempo.

Cada periférico suele estar formado por dos partes claramente diferenciadas en cuanto a su misión y funcionamiento:

- Una parte **mecánica:** Formada básicamente por dispositivos electromecánicos que se controlan a través de elementos electrónicos.
- Una **parte electrónica o controlador de periférico:** Que se encarga de interpretar las órdenes que le llegan de la CPU para la recepción (si es un dispositivo de salida) o transmisión (si es un dispositivo de entrada) de datos y de generar las señales de control para la activación de los elementos electromecánicos.

DISPOSITIVOS DE ENTRADA

Transmiten información desde el mundo exterior al procesador y a la memoria del ordenador mediante la transformación de los datos en señales eléctricas codificadas (código binario). Así, la CPU y la memoria reciben la información adecuadamente preparada para su tratamiento.

Lectora de tarjetas perforadas

Las tarjetas perforadas fueron, hasta finales de los años setenta, el soporte de información que más se utilizó. La tarjeta perforada es una cartulina dura, rectangular, en la que la información se representa con caracteres grabados por medio de perforaciones que realizan máquinas auxiliares denominadas perforadoras. Las tarjetas que se van a leer se depositan en las unidades

lectoras de tarjetas y pasan secuencialmente por un dispositivo de lectura que convierte la presencia o ausencia de perforación en un impulso eléctrico.

Teclado

Los teclados son similares a los de una máquina de escribir, correspondiendo cada tecla a uno o varios caracteres, funciones u órdenes. Para seleccionar uno de los caracteres, puede ser necesario pulsar simultáneamente dos o más teclas, una de ellas correspondiente al carácter (mayúsculas, minúsculas, Alt, etc.). El teclado dispone de un conjunto de teclas agrupadas en 4 bloques:

- **Teclado principal o alfanumérico:** Contiene los caracteres alfabéticos, numéricos y especiales, como en una máquina de escribir convencional, con alguno más.
- **Teclado numérico:** Es habitual que las teclas correspondientes a los dígitos decimales, signos de operaciones básicos y punto decimal estén repetidas para facilitar al usuario la introducción de datos numéricos.
- **Teclas de gestión de imagen o de control:** Sobre la pantalla se visualiza una marca o cursor (indicador de posición). También se suelen denominar con el nombre de teclas del cursor. El cursor indica la posición donde aparecerá el siguiente carácter que tecleemos. Las teclas de gestión de imagen permiten modificar la posición de dicho cursor en la pantalla.
- **Teclas de función:** Normalmente distribuidas en una hilera en la parte superior del teclado. El número más usual de teclas de función es 12 (F1, F2,..., F12). Son teclas cuyas funciones están definidas por el usuario o predefinidas por una aplicación. Así, la tecla F1 tiene funciones diferentes dependiendo de la aplicación que se esté ejecutando. En la mayoría de las aplicaciones Windows, por ejemplo, al pulsar la tecla F1 se abre una ventana de ayuda. Cuando se presiona una tecla, un pequeño chip dentro de la computadora o del teclado, llamado controlador del teclado, se percata de que una tecla ha sido presionada y coloca un código en parte de su memoria, denominada memoria temporal del teclado (buffer), que indica qué tecla fue seleccionada. El controlador envía una petición de interrupción a la CPU y cuando la CPU la acepta pasa el carácter del buffer a la CPU.

Lápiz óptico

Físicamente tiene la forma de una pluma o lápiz, de uno de cuyos extremos sale un cable que se conecta al monitor. El otro extremo tiene una abertura por la que puede pasar la radiación luminosa de la pantalla. El lápiz contiene un pulsador, transmitiéndose información hacia el monitor únicamente en el caso de estar presionado.

Detector de caracteres magnéticos

Se utiliza en talones y cheques bancarios. En estos documentos se imprimen unos caracteres con tinta magnetizable. El dispositivo que lee los cheques o talones contiene una microbobina que va barriendo el carácter y generando un potencial proporcional a la cantidad de tinta del carácter.

Detector de barras impresas (códigos de barras)

Los códigos de barras se están convirtiendo en la forma estándar de representar información en los productos de mercado. Cuando se fabrica un producto, se imprime en su envoltorio una etiqueta con información sobre el mismo según un código formado por un conjunto de barras separadas por zonas en blanco. La forma de codificar cada dígito decimal consiste en variar el grosor relativo de las barras negras y blancas.

Sistemas de adquisición de datos analógicos

La mayor parte de las variables físicas de la naturaleza (temperatura, luminosidad, etc.) son señales o funciones que varían continuamente con el tiempo. Estas señales, con sensores o detectores, pueden convertirse en señales eléctricas analógicas. Existen sensores específicos para cada magnitud (temperatura, presión, luminosidad, humedad, humo, sonido, sensores de señales fisiológicas, etc.). Una vez convertida la señal original en señal eléctrica, es necesario transformarla en datos aptos para ser tratados por la computadora (datos binarios). Esto se hace con unos circuitos electrónicos específicos denominados conversores analógico/digital (conversores A/D).

Escáner de imágenes

Es un sistema para digitalización de documentos basado en la exploración de documentos mediante procedimientos optoelectrónicos. El escáner transforma la información contenida en una página en una señal eléctrica que es transmitida al ordenador. El sistema considera a una página dividida en una fina retícula de celdas o puntos de imagen, que son iluminados por una fuente de luz. Esta luz se refleja en cada celda, y una malla de sensores optoelectrónicos convierte la luz reflejada en una carga eléctrica (en una señal analógica). Las señales analógicas obtenidas como consecuencia del barrido de la página, son digitalizadas por un conversor A/D, conformando así la imagen captada para poder ser almacenada y procesada. Cuanto más fina es la retícula considerada por el sistema, mayor resolución o mayor información se tiene sobre la figura y de mayor calidad será la imagen captada.

La página que lee un escáner siempre es una imagen gráfica, aunque el contenido sea una carta o un artículo de una revista. Es decir, la salida que genera el escáner es siempre un fichero en formato gráfico (PCX, TIFF, BMP, etc.). El usuario que ha utilizado el escáner para leer el documento original sólo podrá modificarlo utilizando un programa de tratamiento de imágenes.

Si lo que el usuario quiere es modificar el texto del documento original con un procesador de textos o un programa de autoedición, entonces necesitará un programa de reconocimiento óptico de caracteres (OCR) para convertir esa imagen que nos entrega el escáner en un fichero ASCII o en un formato propio de un procesador de textos.

El OCR está basado en el uso de un dispositivo de exploración óptica (escáner) que puede reconocer la letra impresa. El OCR tiene que analizar la imagen punto a punto para identificar todos los posibles caracteres. Algunos programas OCR son capaces de identificar cualquier tipo de letra y tamaño, e incluso distinguen los caracteres en negrita y cursiva, y otros pueden llegar a reconocer letra manuscrita siempre que esté suficientemente clara (como en administraciones postales para procesos de clasificación). Para llevar a cabo esta tarea, el OCR compara los caracteres que encuentra en la página con definiciones internas de los caracteres que él tiene definidos. Es decir, el OCR ve un carácter y trata de igualarlo a lo que asume que debería parecerse dicho carácter. Para realizar esta función, utiliza un catálogo de patrones.

El software de OCR es muy complejo, porque es muy difícil hacer que una computadora reconozca un número ilimitado de caracteres tipográficos y fuentes. Los requisitos básicos que debe cumplir un programa OCR son la fiabilidad y la velocidad de interpretación de caracteres. Algunos programas que nos encontramos en el mercado son OmniPage, WordScan, Perceive, TextPert, Recognita, etc.

Reconocimiento de voz

Es uno de los campos de investigación más relevantes en la actualidad, pero aún no está muy desarrollado. Se pretende una comunicación directa del hombre con el ordenador. Básicamente, los dispositivos de reconocimiento de voz pretenden convertir el lenguaje humano al lenguaje máquina. Tratan de reconocer fonemas o palabras dentro de un repertorio o vocabulario muy reducido. Para ello, lo que hace al detectar un sonido es extraer características o parámetros de dicho sonido y compararlo con los parámetros de las palabras que es capaz de reconocer. Si se consigue identificar el sonido como una palabra del vocabulario memorizado en el ordenador, se transmite a la memoria intermedia del dispositivo el código binario identificador de la misma. Si el sonido no se identifica, se le indica al usuario mediante algún mecanismo (indicador luminoso). Existen sistemas que sólo reconocen la voz de un locutor determinado (necesitan un aprendizaje previo), y otros que son independientes de la persona que hable (reconocen menos palabras).

Ratón

El ratón es un dispositivo de entrada que sirve para introducir información gráfica o seleccionar coordenadas (x,y) de una pantalla. Dispone de uno o más pulsadores con los que el usuario envía órdenes a la computadora, relacionadas con el punto seleccionado en la pantalla. Internamente está constituido por una bola que puede girar libremente y unos rodillos perpendiculares entre sí. Cuando el ratón se desplaza sobre una superficie, la bola se mueve y hace girar los rodillos en un sentido u otro. Esta información es transmitida a través de un cable a la computadora y el programa gestor del ratón puede determinar la distancia, dirección y sentido del desplazamiento desde que se inició el último movimiento. Los ratones detectan

movimientos relativos. En la pantalla aparece un cursor que se mueve en el mismo sentido en el que se desplaza el ratón a través de una superficie, indicando el punto sobre el que se actuará. El dispositivo que acabamos de describir se conoce con el nombre de ratón mecánico, sin embargo existen también los denominados ratones ópticos. A diferencia del ratón mecánico, que puede deslizarse por cualquier superficie que permita el movimiento de su bola, en el ratón óptico el movimiento se tiene que realizar sobre una tablilla especial de material reflectante. El ratón contiene dos focos luminosos que proyectan dos haces sobre la tablilla, la cual los refleja y pasan a través de dos orificios para ser detectados por un par de fotosensores. Este tipo de ratón es menos propenso a fallos y averías, pero presenta el inconveniente de necesitar la tablilla para el desplazamiento y de ser más caro.

Con ordenadores portátiles o en situaciones en las que hay poco espacio para desplazar el ratón suelen utilizarse ratones estacionarios (trackball), que se usan con la bola hacia arriba, de forma que ésta se desplaza con el dedo pulgar y no haciéndola rodar por una superficie.

Palanca manual de control (Joystick)

Está constituida por una caja de la que sale una palanca o mando móvil. El usuario puede actuar sobre el extremo de la palanca y a cada posición de ella le corresponde sobre la pantalla un punto de coordenadas (x,y). La caja o la varilla dispone de un pulsador que debe ser presionado para que exista una interacción entre el programa y la posición de la varilla. Este dispositivo es muy utilizado en videojuegos y aplicaciones gráficas.

Digitalizadores

También denominados tablas digitalizadoras o tabletas gráficas, permiten transferir directamente gráficas, figuras, planos, mapas, etc. al ordenador. Esto se hace pasando una pieza móvil (lápiz o cursor) por encima de la línea a digitalizar (como si se estuviese calcando), automáticamente se pasan las coordenadas de los puntos que forman la imagen. Partiendo de un dibujo, se obtiene una representación digital de él. Los digitalizadores constan de un tablero donde se ubica el dibujo a digitalizar.

DISPOSITIVOS DE SALIDA

Transmiten información desde el procesador y la memoria del ordenador al exterior mediante la transformación de señales eléctricas binarias en un lenguaje inteligible para los humanos (normalmente caracteres escritos o visualizados).

Perforadora de tarjetas

Su misión consiste en recibir datos de la computadora o de las personas y realizar las correspondientes perforaciones. Suele llevar incorporada otra unidad verificadora de tarjetas, que repite las perforaciones para comprobar la información que debe contener la tarjeta.

Monitor

La forma más cómoda de recibir información es a través de la vista. Los monitores constituyen el sistema más cómodo y usual de captar las salidas de una computadora.

La imagen de pantalla de ordenador se forma con multitud de puntos denominados puntos de imagen o píxeles. La imagen se forma físicamente con la activación selectiva de unos elementos denominados puntos de pantalla. Un punto de pantalla se iluminará más cuanto mayor sea la activación del elemento correspondiente.

Cuando la pantalla se utiliza para visualizar texto, se considera dividida en celdas con un determinado número de píxeles de ancho y largo para representar un carácter.

Un monitor está constituido por dos elementos básicos:

- Controlador de vídeo o controlador gráfico. La mayor parte de los monitores no activan los puntos de pantalla de una forma continua, sino que lo hacen de forma periódica y durante un corto intervalo de tiempo. Esta actualización periódica se llama refresco de pantalla, e implica un recorrido o barrido de la pantalla. Los códigos de los caracteres que van llegando son analizados por los circuitos que constituyen el controlador de vídeo. La mayoría de estos caracteres son para visualizarlos en pantalla, y el controlador los almacena directamente en una memoria denominada memoria o buffer de vídeo.

- Pantalla de vídeo. Las pantallas tradicionales contienen un tubo de rayos catódicos (similar a los de TV) y son de tipo curvo. Cerca de la parte trasera de la cubierta del monitor se encuentra un cañón de electrones, el cual dispara un rayo de electrones a través de una bobina magnética, que apunta el rayo a la parte frontal del monitor. La cara interna del tubo está recubierta con miles de puntos de fósforo. Los electrones al estrellarse sobre el fósforo hacen que éste se ilumine. Un CRT es básicamente un tubo de vacío con un cátodo (el emisor del haz de electrones) y un ánodo (la pantalla recubierta de fósforo) que permite a los electrones viajar desde el terminal negativo (cátodo) hasta el positivo (ánodo). Los monitores monocromos utilizan un único haz de electrones y un único tipo de fósforo, mientras que los monitores en color emplean tres haces y fósforo de tres colores distintos, uno por cada color básico (rojo, verde y azul). Los colores usuales en las pantallas monocromáticas son blanco, verde y ámbar sobre fondo negro. Las pantallas planas son menos voluminosas y menos pesadas (aunque más caras), por lo que se utilizan preferentemente para ordenadores portátiles. Normalmente están compuestas por dos cristales planos unidos a presión y los elementos activos se sitúan entre ambos (pantallas de cristal líquido).

Impresoras

Son periféricos que escriben la información de salida sobre papel. Junto con el monitor son los dispositivos de salida más utilizados. Existen multitud de tipos y modelos. Se clasifican según dos criterios:

- Por el modo de impresión de los caracteres:
 - Impresoras con impacto. Son aquellas que para imprimir los caracteres precisan golpear sobre el papel el carácter preformado en relieve o configurado en una cabeza de escritura. La ventaja de este tipo de impresoras es que se pueden realizar varias copias simultáneas del documento intercalando papel carbón. Como inconveniente, puede considerarse el excesivo ruido producido con el golpeo.
 - Impresoras sin impacto. Se eliminan los movimientos mecánicos y el

impacto, con lo que se consiguen mayores velocidades y desaparece el ruido. No se pueden obtener copias simultáneas. Utilizan técnicas basadas en fenómenos térmicos, electrostáticos, químicos, así como el rayo láser.

- Por el número de caracteres que pueden escribir simultáneamente:
 - Impresoras de caracteres. Realizan la impresión carácter a carácter de forma secuencial. Son dispositivos lentos que consiguen velocidades de hasta 600 cps.
 - Impresora de líneas. Realizan la impresión línea a línea, de forma que seleccionando previamente los caracteres que se han de imprimir en una línea, con un único golpe se imprimen simultáneamente todos los caracteres que la componen. Se consideran rápidas, alcanzando velocidades de hasta 2.400 lpm o 5.113 cps.
 - Impresoras de páginas. Imprimen una página de una vez. Son las más rápidas. Se consiguen velocidades de 88.000 cps que son aproximadamente 570 ppm (10 pps).

A continuación, se describen algunos de los tipos de impresoras más importantes entre las existentes en el mercado:

- Impresora de margarita, de cilindro, de bola. Son impresoras con impacto y de tipo carácter. La velocidad de impresión no supera los 50 cps. La cabeza de impresión es una margarita con hojas, un cilindro o una bola que contiene los caracteres en relieve. El mecanismo donde se encuentran preformados los caracteres gira hasta que el carácter que se quiere imprimir se encuentra delante de un martillo que lo golpea, produciendo la impresión.
- Impresora de matriz de puntos o de agujas. Constan de una cabeza de impresión en la que por medio de unos electroimanes que llevan en su interior unos punzones se configura el carácter a imprimir. Pertenecen al tipo de impresoras de carácter y de impacto. Existen impresoras con más de una cabeza de matriz de puntos. En este caso se considera como impresora de línea. La velocidad oscila entre 180 y 500 cps.
- Impresoras de banda de acero. Son de tipo impacto y de línea. Los caracteres se encuentran modelados en ruedas, en tambores, en barras o en cadenas y la forma del molde pasa al papel al impactar sobre él un martillo. La velocidad de impresión está entre 600 y 2.400 lpm.
- Impresoras térmicas. Son similares a las impresoras de agujas. Se imprime sobre un papel especial termosensible que se ennegrece al aplicar calor. El calor se transfiere desde el cabezal por una matriz de pequeñas resistencias. Al pasar una corriente eléctrica por las resistencias, se calientan, formándose los puntos en el papel. Pueden ser de caracteres o de líneas, y son impresoras sin impacto. La velocidad oscila entre 100 y 2.000 cps.
- Impresoras de inyección de tinta. Utiliza tinta líquida que sale por una boquilla en forma de gotas. La tinta se carga eléctricamente y está guiada hacia el papel por medio de placas de desviación para formar el carácter deseado. La calidad de impresión es buena, debido a que los caracteres están formados por docenas de pequeños puntos de tinta, pudiéndose utilizar varios colores de tinta y tipos de letra que se controlan desde el programa.

Su velocidad oscila entre 60 y 660 cps.

- Impresoras láser. Tienen una gran importancia debido a su gran velocidad, calidad de impresión, bajo precio y uso de papel normal. Son impresoras de páginas y sin impacto. La página a imprimir se transfiere al papel por contacto, desde un tambor que contiene la imagen impregnada en tóner (polvo de carbón). El tambor está recubierto de un material fotoconductor. La imagen se forma haciendo incidir sobre el tambor un rayo láser. La velocidad de las impresoras láser va desde 4 a 350 ppm.

Síntesis de voz

Son unidades que dan los resultados de los programas emitiendo sonidos similares al habla humana. Funcionan a base de almacenar muestras codificadas digitalmente de sonidos clave. Las palabras se construyen mediante la unión de estas muestras a través de algún algoritmo.

Registradores gráficos (Plotters)

Estos dispositivos producen salidas en forma de planos, dibujos, mapas, esquemas e imágenes en general. El plotter dispone de una o varias plumas que se mueven sobre la superficie del papel bajo el control del procesador (requieren un software especial para su control). Hoy en día, la importancia de los registradores gráficos ha decrecido, debido a que en muchas ocasiones pueden ser sustituidos por impresoras gráficas. Su uso se suele reservar para dibujos de gran tamaño (AO) o que requieran diversidad de colores. Actualmente, se comercializan plotters con tecnología de inyección de tinta.

DISPOSITIVOS MIXTOS

Los dispositivos mixtos incluyen simultáneamente unidades de entrada y unidades de salida.

Pantallas sensibles al tacto

Son unidades de E/S similares a una pantalla convencional en la que se incluye un dispositivo capaz de reconocer la zona donde se aplica una presión (por ejemplo, el contacto con el dedo). En general, se utiliza para representar información realizando operaciones mediante un grupo de opciones localizadas a lo largo de la pantalla, de forma que una de ellas puede ser reconocida por el contacto. Puede ser útil para usuarios principiantes, tales como niños de corta edad. Es frecuente también encontrar este tipo de dispositivos en algunos comercios y lugares públicos para suministrar información de cualquier índole (precios de artículos, horarios de transportes, etc.).

Robots

Permiten la entrada de datos a través de dispositivos muy variados, como sensores, teclados, analizadores de voz, etc. La salida la realizan por medio de movimientos, síntesis de voz, displays. En general, son dispositivos que mezcla distintas unidades de entrada y salida en una única máquina.

Terminales punto de venta

Son unidades de E/S especiales para aplicaciones muy concretas de tipo comercial. Constan por lo general de un teclado, una impresora y una caja de monedas y billetes controlada por el propio teclado. Funcionan por un software hecho a medida y las funciones que realizan son búsqueda

y actualización automática de precios, gestión de compras, impresión de factura o tique de venta, reconocimiento de código de barras, etc. Actualmente ha sustituido a las clásicas máquinas registradoras.

Terminales de operaciones financieras

También denominados cajeros automáticos, son unidades conectadas a una computadora central de una entidad financiera para la realización de operaciones de los clientes con la mencionada entidad.

DISPOSITIVOS DE MEMORIA MASIVA AUXILIAR

Los sistemas de memoria masiva son periféricos que sirven para almacenar información permanente de manera que se pueda recuperar de forma automática y eficiente. Estos dispositivos tratan de solventar los problemas de la memoria principal: volatilidad y capacidad. La información contenida en un dispositivo de memoria masiva se transfiere desde o hacia la CPU y la memoria principal a través de bloques o registros físicos de información. Cada bloque contiene una cantidad fija de información. Se denomina tiempo de acceso al tiempo medio que se tarda en acceder a cualquier registro físico. Si para acceder a un bloque concreto es necesario que la cabeza vaya leyendo uno a uno los bloques que hay desde el principio hasta el registro deseado, se dice que el dispositivo es de acceso secuencial. Si, por el contrario, la cabeza lectora puede situarse directamente en un registro dado, se dice que el dispositivo es de acceso directo. Estos últimos son más rápidos que los secuenciales.

UNIDAD 2.

PROBLEMAS DE COMPUTACION

1.1 Objetivo

La resolución de problemas utilizando como herramienta una computadora requiere contar con la capacidad de expresión suficiente como para indicar a la máquina lo que debe llevar a cabo.

Se comenzara resolviendo situaciones del mundo real tratando de utilizar determinados elementos que caracterizan a una secuencia de ordenes que una computadora puede comprender.

El tema central de este curso es la definición del concepto de algoritmo y los elementos que lo componen.

1.2 Introducción

La **Informática** es la ciencia que estudia el análisis y resolución de problemas utilizando computadoras.

La palabra ciencia se relaciona con una metodología fundamentada y racional para el estudio y resolución de los problemas.

Si se busca en el diccionario una definición de la palabra problema podrá hallarse alguna de las siguientes:

- Cuestión o proposición dudosa, que se trata de aclarar o resolver.
- Enunciado encaminado a averiguar el modo de obtener un resultado cuando se conocen ciertos datos.

La resolución de problemas mediante una computadora consiste en dar una adecuada formulación de pasos precisos a seguir.

1.3. Resolución de Problemas

Si se piensa en la forma en que una persona indica a otra como resolver un problema, se vería que habitualmente se utiliza un lenguaje común y corriente para realizar la explicación, quizá entremezclado con algunas palabras técnicas. Esto es un riesgo muy grande. Los que tienen cierta experiencia al respecto saben que es difícil transmitir el mensaje y por desgracia, con mucha frecuencia se malinterpretan las instrucciones y por lo tanto se "ejecuta incorrectamente" la solución, obteniéndose errores.

Cuando de una computadora se trata, no pueden utilizarse indicaciones ambiguas. Ante cada orden resulta fundamental tener una única interpretación de lo que hay que realizar. Una máquina no posee la capacidad de decisión del ser humano para resolver situaciones no previstas.

Si al dar una orden a la computadora se produce una situación no contemplada, será necesario abortar esa tarea y recomenzar todo el procedimiento nuevamente.

Además, para poder indicar a la computadora las ordenes que debe realizar es necesario previamente entender exactamente lo que se quiere hacer. Es fundamental conocer con qué información se cuenta y qué tipo de transformación se quiere hacer sobre ella.

Una vez que se comprende un problema, se debe decidir que tipo de problema es. Dos tipos de problemas comunes son:

- Los **problemas que buscan respuestas**: si un ejercicio implica el cálculo del número de palabras que hay en un libro o la bisección de una línea utilizando regla y compás, se trata de un trabajo en que se debe encontrar algo que se desconoce. La forma en que esto se haga no es de particular importancia siempre y cuando se obtenga la respuesta correcta (aunque es deseable que sea por un medio fácil).
- Los **problemas que buscan pruebas**: cuando se pide que se pruebe que hay 720 formas de colocar seis libros en una fila, se está dando la respuesta. La tarea es distinta a la de encontrar una respuesta porque ya se sabe lo que se desea. Todo lo que se tiene que hacer es determinar la relación entre los datos y la respuesta.

Es importante observar que en un problema en que se buscan respuestas se tiene que elaborar una solución, mientras que cuando se pide que se pruebe sólo es necesario demostrar que existe (o no existe) una solución, sin crearla. Los problemas de computación no pueden ser problemas en que se busquen pruebas porque el propósito del trabajo de las computadoras es encontrar respuestas que no se conocen de antemano. Pero tampoco se les puede considerar como problemas en que se busquen respuestas porque es la computadora misma la que determina la respuesta, no la persona que trabaja con ella.

Los problemas de computación pertenecen a una tercera clase: los **problemas que buscan métodos**, aquí se busca un método mediante el cual se pueda derivar una respuesta.

El proceso de resolver problemas mediante computadoras se describe en la figura 1. Se trata de encontrar un método por medio del cual se pueda resolver un problema. Una vez que se haga esto, la computadora se hace cargo del mismo y suministra las respuestas a la pregunta.



Figura 1.

Esta es una simplificación porque una vez que se tiene un método es necesario expresar este método en una forma en que la computadora pueda operarlo (un programa).

ETAPAS PARA LA RESOLUCIÓN DE PROBLEMAS CON COMPUTADORAS

CAPITULO 2 LIBRO DE JOYANES AGUILAR (El libro digital está en el Aula Virtual de la asignatura) solo el punto 2.1

ALGORITMOS

2.1. Introducción

La etapa vital de la solución de un problema con una computadora es el diseño del algoritmo y de la estructura fundamental de datos. Un algoritmo es un procedimiento expresado precisamente para obtener la solución del problema, la que se presenta de manera subsecuente a una computadora en el lenguaje de programación seleccionado. Los algoritmos se presentan de una manera conveniente para un lector humano, mientras que los programas sirven a las necesidades de las computadoras.

Es importante recordar mientras diseñamos un algoritmo que una computadora sólo sigue las instrucciones y no puede actuar si no se le ha ordenado de manera explícita. Por lo tanto, el solucionador de problemas debe prever cualquier aspecto del problema en el propio algoritmo.

La palabra **algoritmo** deriva del nombre de un matemático árabe del siglo IX, llamado Alkhuwarizmi, quien estaba interesado en resolver ciertos problemas de aritmética y describió varios métodos para resolverlos. Estos métodos fueron presentados como una lista de instrucciones específicas (como una receta de cocina) y su nombre se utiliza para referirse a dichos métodos.

2.2. Definición

Un algoritmo es, en forma intuitiva, una receta, un conjunto de instrucciones o de especificaciones sobre un proceso para hacer algo. Ese "algo" generalmente es la solución de un problema de algún tipo. Formalmente un algoritmo se puede definir de la siguiente forma:

Un algoritmo puede definirse como una secuencia ordenada de pasos elementales, exenta de ambigüedades, que lleva a la solución de un problema dado en un tiempo finito.

Para comprender la definición anterior se clarifica algunos conceptos.

Ejemplo: escriba un algoritmo que permita preparar una tortilla de papas de tres huevos.

Si la persona que resuelva el problema es un cocinero lo resuelve sin mayor nivel de detalle, pero si no lo es, se deben describir los pasos necesarios para realizarlo:

Paso 1: Mezclar papas fritas, huevos y una pizca de sal en un recipiente.

Paso 2: Freír.

Esto podría resolver el problema, pero si la persona que lo ejecute no sabe cocinar, se

debe detallar, cada uno de los pasos mencionados, pues estos no son lo bastante simples para un principiante. De esta manera el primer paso se puede descomponer en:

Paso 1: Pelar las papas

Paso 2: Cortar las papas en cuadraditos

Paso 3: Freír las papas

Paso 4: Batir los huevos en un recipiente

Paso 5: Verter las papas en un recipiente y echar una pizca de sal

El tercer paso puede descomponerse en:

- Calentar el aceite en la sartén
- Verter las papas en la sartén
- Sacar las papas ya doradas en un recipiente

Nótese que si la tortilla la va a ejecutar un niño, alguna tareas, por ejemplo batir huevos, pueden necesitar una mejor especificación.

Con este ejemplo se pretende mostrar que la lista de **pasos elementales** que compongan nuestro algoritmo depende de quien sea el encargado de ejecutarlo. Si en particular, el problema va a ser resuelto utilizando una computadora, el conjunto de pasos elementales conocidos es muy reducido, lo que implica un alto grado de detalle para los algoritmos.

Se considera entonces como un paso elemental aquel que no puede volver a ser dividido en otros más simples.

Otro aspecto importante de un algoritmo es su nivel de detalle, que no debe confundirse con el concepto de paso elemental. En ocasiones, no se trata de descomponer una orden en acciones más simples sino que se busca analizar cuáles son las órdenes relevantes para el problema. Para comprender lo expuesto se lo analiza con un ejemplo:

Ejemplo: escriba un algoritmo que describa la manera en que Ud. se levanta todas las mañanas para ir al trabajo.

Paso 1: Salir de la cama

Paso 2: Quitarse el pijama

Paso 3: Ducharse

Paso 4: Vestirse

Paso 5: Desayunar

Paso 6: Arrancar el auto para ir al trabajo

La solución del problema se expresó en seis pasos, descartando aspectos que se suponen o sobreentienden, como “colocarse los zapatos al vestirse” o “abrir la puerta del auto” pues a nadie se le ocurriría ir a trabajar descalzo. En cambio existen aspectos que no pueden obviarse o suponerse porque el algoritmo perdería lógica, por ejemplo el paso “vestirse”, no puede ser omitido. Puede discutirse si se requiere un mayor nivel de detalle o no, pero no puede ser eliminado del algoritmo.

Un buen desarrollador de algoritmos deberá reconocer esos aspectos importantes y tratar de simplificar al mínimo su especificación de manera de seguir resolviendo el problema con la menor cantidad de órdenes posibles.

Además, en la definición de algoritmo se hace referencia a la ambigüedad y tiempo de respuesta, debido a que todo algoritmo debe cumplir con ciertas propiedades para que se lo considere como tal y proporcione el resultado deseado cuando un programa basado en él se presenta a una computadora.

Un algoritmo debe cumplir las siguientes propiedades:

Ausencia de Ambigüedad: si se trabaja dentro de cierto marco o contexto, la representación de cada paso de un algoritmo debe tener una única interpretación.

Ejemplo: indique la forma de condimentar una salsa.

Incorrecto: ponerle algunas especias a la salsa.

Correcto: ponerle sal, pimienta y orégano a la salsa.

Generalidad: un algoritmo se puede realizar para varios problemas que se relacionan entre si, es decir, se debe aplicar a un problema o clase de problemas específicos.

Ejemplo: indique la forma de marcar un número de teléfono.

Incorrecto: si la solución del algoritmo sirve para marcar solamente el número 4220234, solo tendrá valor para ese número.

Correcto: si la solución es un método para marcar cualquier número, entonces es útil. Por supuesto, debe haber alguna restricción a la generalidad de un algoritmo.

Tiempo de respuesta: la ejecución de un algoritmo debe finalizar después de que se haya llevado a cabo una cantidad finita de pasos. De otra manera, no se puede exigir que la ejecución produzca una solución.

Ejemplo: Llene la zanja con ese montón de arena

Algoritmo: tome una pala y empiece a echar arena en la zanja. Cuando se llene la zanja deténgase.

(se está seguro que en algún momento parará, aunque no se sabe cuanto tardará).

2.3. Dominio de un Algoritmo

La clase o el conjunto de datos y las condiciones para las cuales un algoritmo trabaja correctamente se llama dominio. Cuando se trata de resolver cualquier problema es necesario definir el dominio del algoritmo y después verificar que trabaja para todos los casos que se encuentran dentro de ese dominio.

Al decidir el dominio de un algoritmo es necesario incluir todas las situaciones similares, pero los casos remotos o poco probables se pueden omitir.

2.4. Errores en la Construcción de un Algoritmo

En la construcción de algoritmos se consideran dos tipos de errores:

- **Errores de Dominio:** se presentan cuando no se han especificado todas las situaciones que se pueden presentar en la práctica o se ha descuidado la apreciación de su importancia. Las pruebas más difíciles son aquellas que verifican que se ha seleccionado un dominio correcto para el algoritmo. Cuando una situación no prevista se presenta, hay tres opciones:
 - Ignorarla porque es improbable y quizás nunca ocurra.
 - Restringir el dominio del algoritmo para excluirla.
 - Corregir el algoritmo.
- **Errores de Lógica:** son aquellos errores que se detectan después de que se ha definido en forma adecuada el dominio de un algoritmo, en la etapa de prueba o verificación. se deben principalmente a las siguientes causas:
 - Etapas incorrectas
 - Secuencia incorrecta de etapas.

3. FORMAS DE EXPRESAR UN ALGORITMO

Un mismo algoritmo puede ser expresado de distintas formas.

- **Lenguaje común.** En el lenguaje normal que hablamos y escribimos; útil para comunicar un algoritmo a otra persona o en una fase de análisis previo de un sistema computacional.
- **Diagramas de flujo.** Es un lenguaje gráfico; útil para visualizar en forma rápida la secuencia lógica de pasos a seguir para un algoritmo y de gran ayuda para la traducción del mismo a un programa de computación
- **Pseudocódigo:** Es una técnica para expresar en lenguaje natural la lógica de un programa, es decir, su flujo de control. El pseudocódigo no es un lenguaje de programación sino un modo de plantear un proceso de forma que su traducción a un lenguaje de alto nivel sea sencillo para el programador.
- **Lenguajes de Programación:** Es la forma obligada de expresión de un algoritmo para que pueda ser leído, ejecutado y almacenado por el computador.

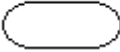

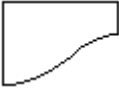

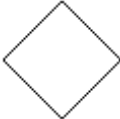


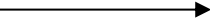
3.1. Diagramas de Flujo

Un diagrama de flujo es la representación gráfica o visual de un algoritmo. Se usan en el planeamiento, desarrollo y estructuración de un algoritmo. Mediante los diagramas de flujo el algoritmo se puede comunicar y documentar (porque enseña y describe el proceso).

Formalmente, un diagrama de flujo es un diagrama formado por símbolos (cajas, bloques, figuras) y flechas o líneas de flujo que conectan los símbolos entre si. Los símbolos denotan los pasos esenciales del algoritmo y las flechas indican la secuencia. Se dibujan de tal manera que la dirección del flujo sea hacia abajo o de izquierda a derecha.

3.2. Simbología Básica de los Diagramas de Flujo

Los símbolos que se describen a continuación, para la representación gráfica de los diagramas de flujo, son de uso universal.

SÍMBOLO	SIGNIFICADO
	Indica principio o fin de un algoritmo.
	Indica entrada de datos.
	Indica la salida de la información.
	Se usa generalmente para sentencias o enunciados de asignación (acción de asignar) y para la realización de un proceso matemático.
	Permite evaluar una expresión relacional ó lógica que puede tomar un valor verdadero o falso. En función de este resultado el flujo del algoritmo seguirá una determinada dirección.
	Se usa cuando el diagrama es largo y se requiere más de una hoja de papel o para evitar líneas que se crucen.
	Simbolizan módulos o segmentos lógicos
	Indica la dirección del algoritmo en cada momento mediante una flecha.

3.3. Utilidad de los Diagramas de Flujo

El diagrama de flujo refleja los pasos sucesivos que el computador debe dar para llegar a la solución de un problema. Las principales razones por las cuales es generalmente aconsejable el trazado de un diagrama de flujo son las siguientes:

- Oportunidad de verificar la lógica de la solución.
- Sirve de guía al programador para la codificación del programa.
- Permite fácilmente modificar un programa.
- Es útil para la discusión grupal.
- Sirve para documentar el programa.

ESTRUCTURAS DE CONTROL BÁSICAS

Al ser un algoritmo una secuencia de pasos ordenados, estos deben seguir una trayectoria para su ejecución desde el primer paso hasta el último. Esta trayectoria se denomina **flujo de control** que indica el orden en el cual deben ejecutarse los pasos elementales.

Para organizar el flujo de control de un algoritmo se utilizan **estructuras de control**, estas son construcciones algorítmicas lineales, de selección e iteración. Las dos últimas alteran el flujo de control lineal del algoritmo.

Las estructuras de control básicas para organizar el flujo de control en un algoritmo, son las siguientes:

- Estructura secuencial
- Estructura de selección
- Estructura de iteración

5.1. Estructura Secuencial

La estructura de control más simple está representada por una sucesión de acciones que se ejecutan de arriba hacia abajo sin bifurcaciones, es decir, una acción a continuación de otra.

Ejemplo: escriba un algoritmo que describa la forma en que una persona se levanta todas las mañanas para ir al trabajo.

Paso 1: Salir de la cama

Paso 2: Quitarse el pijama

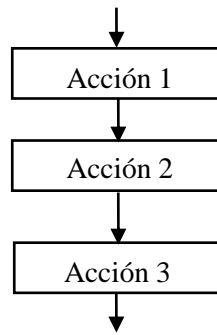
Paso 3: Ducharse

Paso 4: Vestirse

Paso 5: Desayunar

Paso 6: Arrancar el auto para ir al trabajo

A continuación se presenta gráficamente esta estructura.



5.2. Estructura de Selección

En un algoritmo representativo de un problema real es prácticamente imposible que las instrucciones sean secuenciales puras. Es necesario tomar decisiones en función de los datos del problema.

A través de la selección se incorpora, a la especificación del algoritmo, la capacidad de decisión. De esta forma será posible seleccionar una de **dos alternativas** de acción posibles durante la ejecución del algoritmo.

La selección se expresa con el siguiente pseudocódigo:

Si (condición)

entonces

acción o acciones a realizar si la condición es verdadera (1)

sino

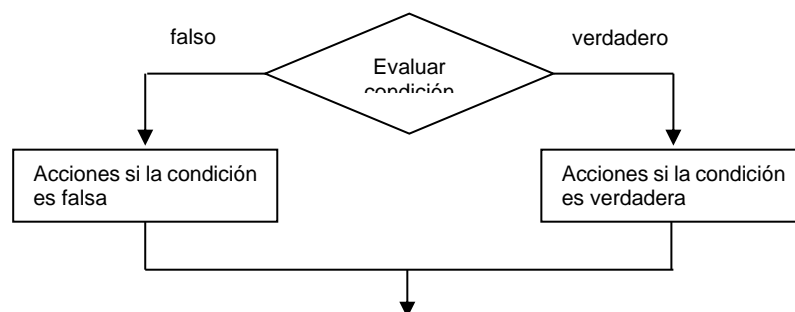
acción o acciones a realizar si la condición es verdadera (2)

FinSi

donde **condición** es una expresión que al ser evaluada puede tomar solamente uno de los dos valores posibles: verdadero o falso.

En el caso que la condición a evaluar resulte verdadera se ejecutarán las acciones (1) y **no** se ejecutarán las acciones (2). Si la condición a evaluar resulta falsa se ejecutarán las acciones (2) y **no** las acciones (1).

A continuación se presenta gráficamente esta estructura.



Puede ocurrir que no se tengan que representar acciones cuando la condición es falsa. En este caso se utilizará la siguiente notación:

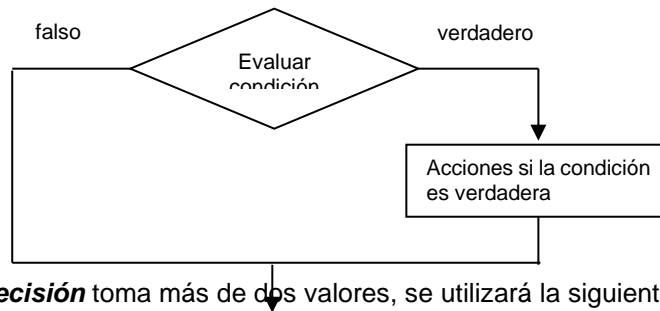
Si (condición)

entonces

acción o acciones a realizar si la condición es verdadera

FinSi

A continuación se presenta gráficamente esta estructura.



Si al evaluar un **decisión** toma más de dos valores, se utilizará la siguiente notación:

Si (variable de decisión)

Valor 1: Acción 1

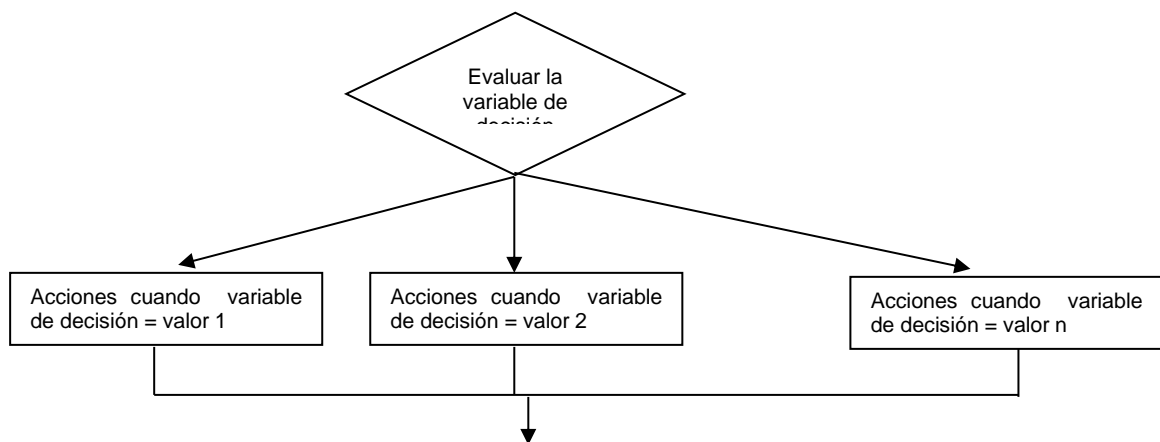
Valor 2: Acción 2

.....

Valor N: Acción N

[Otro: Acción N+1]

A continuación se presenta gráficamente esta estructura.



5.3. Estructura de Iteración

Una extensión natural de una estructura secuencial consiste en repetir N veces un bloque de acciones. El fin de la repetición dependerá de un valor predefinido o del cumplimiento de una determinada condición.

Existen dos formas de expresar esta estructura la *Repetición* y la *Iteración* en este curso solo utilizaremos la iteración.

Iteración

Puede ocurrir que se desee ejecutar un conjunto de acciones de un algoritmo desconociendo el número exacto de veces que se ejecutan. Para estos casos existen estructuras

de control iterativas condicionales, es decir, las acciones se ejecutan dependiendo de la evaluación de una condición.

Por lo tanto, dentro de una estructura iterativa, además de una serie de pasos elementales que se repiten; es necesario contar con un mecanismo que lo detenga.

Podemos definir una estructura iterativa como la estructura de control que permite al algoritmo ejecutar en forma repetitiva un conjunto de acciones utilizando una condición para indicar su finalización.

Estas estructuras se clasifican en **pre-condicionales y pos-condicionales**.

Las estructuras pre-condicionales evalúan la condición y, si es verdadera, se ejecuta el conjunto de acciones; esto hace que dicho conjunto se puede ejecutar 0, 1 o más veces.

La notación para esta estructura es la siguiente:

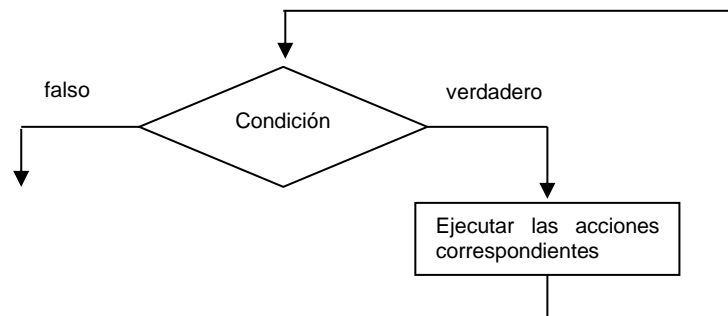
Mientras (condición)

Acción o acciones a realizar en caso de que la condición sea verdadera.

FinMientras

La condición es una expresión que sólo puede tener uno de dos valores posibles: verdadero o falso.

A continuación se presenta gráficamente esta estructura.



Las estructuras pos-condicionales, primero se ejecuta el conjunto de acciones, luego se evalúa la condición y, si es falsa, se ejecuta nuevamente el bloque de acciones. A diferencia de la estructura anterior iterativa anterior, el conjunto de acciones se debe ejecutar 1 o más veces. Nótese que, en este caso, el bloque de acción se ejecuta antes de evaluar la condición, por lo tanto se lleva a cabo al menos una vez.

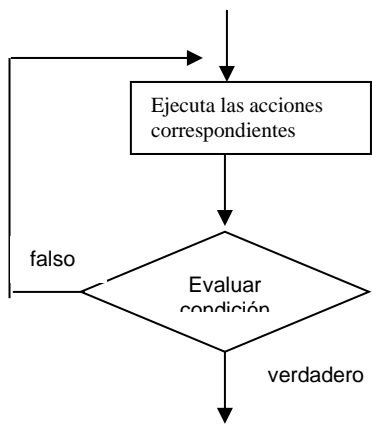
La notación a utilizar para esta estructura es la siguiente:

Repetir

Acción o acciones a realizar en caso de que la condición sea falsa

Hasta (condición)

A continuación se presenta gráficamente esta estructura.



6 TIPO DE DATOS

6.1. Datos Simples

Los programas que implementan los algoritmos necesitan alguna manera de representar los objetos del mundo real. Para ello los algoritmos operan sobre datos y estructuras de datos de distinta naturaleza, tales como números, letras, símbolos, etc.

Por lo tanto un **dato** es la expresión general que describe los objetos con los cuales opera una computadora. Los tipos de datos están ligados a un conjunto de operaciones que permiten su creación y manipulación.

Los tipos de datos se caracterizan por:

- Un rango de valores posibles
- Un conjunto de operaciones realizables sobre ese tipo
- Su representación

Los tipos de datos simples son:

- Numérico
- Lógico
- Carácter

6.1.1. Tipo de Dato Numérico

El **tipo de dato numérico** es el conjunto de los valores numéricos que pueden representarse de dos formas:

- Enteros
- Reales

Enteros

El tipo entero consiste de un subconjunto finito de los números enteros y su tamaño puede variar según el valor que tenga:

...-3, -2, -1, 0, 1, 2, 3...

Dado que una computadora tiene memoria finita, la cantidad de valores enteros que se pueden representar sobre ella son finitos, por esto se deduce que existe un número entero máximo y otro mínimo.

La cantidad de valores que se pueden representar depende de la cantidad de memoria (bits) que se utilicen para representar un entero.

Ejemplo:

Hay sistemas de representación numérica que utilizan 16 dígitos binarios (bits) para almacenar en memoria cada número entero, permitiendo un rango de valores enteros entre -215 y +215. Otros sistemas utilizan 32 bits, por lo que el rango es entre -231 y +231.

En general, para cada computadora existe el entero **maxint**, tal que todo número entero n puede ser representado si

$$-\text{maxint} \leq n \leq \text{maxint}$$

donde **maxint** identifica al número entero más grande que se puede representar con la cantidad de dígitos binarios disponibles.

Reales

El tipo de dato real es una clase de dato numérico que permite representar números decimales. Los valores fraccionarios forman una serie ordenada, desde un valor mínimo negativo, hasta un valor máximo determinado por la norma IEEE 754, de 1985, pero los valores no están distribuidos de manera uniforme en ese intervalo, como sucede con los enteros.

Se debe tener en cuenta que el tipo de dato real tiene una representación finita de los números reales; dicha representación tiene una precisión fija, es decir, un número fijo de dígitos significativos. Esta condición es la que establece una diferencia con la representación matemática de los números reales. En este caso se tienen infinitos números diferentes, en tanto que la cantidad de representaciones del tipo de dato real está limitada por el espacio en memoria disponible.

La representación para números reales se denomina **coma flotante**. Esta es una generalización de la conocida notación científica, que consiste en definir cada número como una mantisa (o fracción decimal), que da los dígitos contenidos en el número; y un exponente (o característica), que determina el lugar del punto decimal con respecto a estos dígitos.

Ejemplo:

Sea el número 8941295000000000. Su representación científica en cualquier calculadora es $8,941295 \times 10^{16}$.

La mantisa del número real representado en la computadora en este caso es : 0,8941295 y el exponente 17. Nótese que, a diferencia de la representación anterior, la coma se ubica inmediatamente a la izquierda del primer dígito significativo y, por lo tanto, la magnitud del exponente se incrementa en 1.

Si el número es 0,0000000000356798, su representación utilizando notación científica es $3,56798 \times 10^{-11}$.

La mantisa es 0,356798 y el exponente es -10 .

Se observa que un exponente positivo lleva a desplazar la coma decimal tantos lugares

6.1.2. Tipo de Dato “Carácter”

Un tipo de dato carácter proporciona objetos de la clase de datos que contiene solo un elemento como su valor. Este conjunto de elementos está establecido y normalizado por un

estándar llamado ASCII (American Standard Code for Information Interchange), el cual establece cuales son los elementos y el orden de precedencia entre los mismos. Los elementos son las letras, número y símbolos especiales disponibles en el teclado de la computadora y algunos otros elementos gráficos. Cabe acotar que el código ASCII no fue único, pero es el más utilizado internacionalmente.

Son elementos carácter:

- Letras minúsculas: 'a', 'b', 'c', ..., 'y', 'z'
- Letras mayúsculas: 'A', 'B', 'C', ..., 'Y', 'Z'.
- Dígitos: '0', '1', '2', '3', ..., '8', '9'.
- Caracteres especiales tales como: '!', '@', '#', '\$', '%', ...

Se debe tener en cuenta que no es lo mismo el valor entero 1 que el símbolo carácter '1'. Un valor del tipo de dato carácter es **solo uno** de los símbolos mencionados.

Operaciones sobre Datos Carácter

Los operadores relacionales descriptos en el tipo de dato numérico, pueden utilizarse también sobre los valores del tipo de dato carácter. Esto es, dos valores de tipo carácter se pueden comparar por =, <>, >, <, >=, <=; el resultado de cualquiera de ellos es un valor de tipo de dato lógico.

La comparación de dos caracteres es posible dado que el código ASCII define para cada símbolo un valor en su escala. De esta manera, al comparar dos símbolos, para determinar el resultado se utiliza el valor dado por el código.

Dentro del código ASCII los valores de los dígitos son menores que los valores de las letras mayúsculas, y estos a su vez menores que los de las letras minúsculas. Los valores dentro del código para los símbolos especiales, o bien son menores que los dígitos, o bien mayores que las letras minúsculas.

Ejemplo:

('á' = 'Á')

da como resultado *falso*

('c' < 'Z')

da como resultado *falso*

('c' < 'z')

da como resultado *verdadero*

('X' > '5')

da como resultado *verdadero*

(' ' < 'H')

da como resultado *verdadero*

observación: ' ' ,

representa un espacio en blanco

('4' < '4')

no puede evaluarse pues los operandos son de tipos

6.1.3. Tipo de Dato “Alfanumérico”, “String” o “Cadena de Caracteres”

En la mayoría de los lenguajes de programación existe un tipo estándar que es la cadena de caracteres(string).

De acuerdo a definiciones previas, un carácter es una representación determinada de una letra, número o símbolo que se guarda internamente de acuerdo a su representación determinada por el código ASCII. Cuando se trabaja con el tipo de dato string, se tienen n caracteres tratados como una única variable, donde n proviene de la definición del string.

Un tipo de dato String es una sucesión de caracteres que se almacenan en un área contigua de memoria y que puede ser leído o escrito.

El tipo de dato string debe indicar el tamaño máximo que se desea manejar, y se define como:

Type nombre-string = string [longitud]

Donde *nombre-string* es el identificador del tipo, y *longitud* es el número máximo de caracteres que puede contener.

Una vez definido el tipo se pueden declarar variables de ese tipo.

Ejemplos (en lenguaje Pascal)

```
Type cadena1 = string[10]
      cadena2 = string[25]
Var c1, c2: cadena1
    denominación: cadena2
```

Ejemplos de asignación en variables string

```
c1 = 'pepe'
c2 = '678@#$abc'
denominación = 'remera deportiva'
```

6.1.4. Tipo de Dato “Lógico”

El tipo de dato lógico, también llamado booleano, en honor al matemático británico George Boole (quien desarrolló el Álgebra lógica o de Boole), es un dato que puede tomar un valor entre un conjunto formado por dos posibles:

- Verdadero (true)
- Falso (false)

Se utiliza en casos donde se representan dos alternativas a una condición. Por ejemplo, si se debe determinar si un valor es primo; la respuesta será verdadera (true) si el número es divisible solamente por si mismo y la unidad; en caso contrario, si tiene algún otro divisor, la respuesta será falsa (false).

6.2. Constantes y Variables

Los algoritmos contienen ciertos valores que no deben cambiar durante la ejecución del mismo. Tales valores se llaman Constantes. De igual forma existen otros valores que cambian durante la ejecución del mismo, estas son las Variables.

Una **constante** es una partida de datos(objetos) que permanece sin cambios durante todo el desarrollo del algoritmo o durante la ejecución del programa.

Una **variable** es un objeto o partida de datos cuyo valor puede cambiar durante el desarrollo del algoritmo o ejecución del programa.

6.3. Expresión: Definición

“Una expresión es un conjunto de variables y/o constantes unidas por operadores”.

Dependiendo del tipo de operadores con que se este trabajando y el resultado obtenido podremos hablar de *Expresiones Aritméticas* o *Expresiones Lógicas*.

Operadores Aritméticos

+	Suma
-	Resta
*	Multipliación
/	División
^	Potencia

Una Expresión Aritmética es aquella que cuando se la evalúa siempre se obtiene un resultado numérico.

Ejemplos de expresiones aritméticas

$2 * 5 + 7 = 17$
 $3 * A / 4 + Cont \wedge 2$
 $Suma = Suma + N$

Operadores Relacionales

<	Menor
>	Mayor
<=	Menor igual
> =	Mayor igual
=	Igual a
<>	Distinto a

Operadores Lógicos

Los operadores lógicos o boléanos básicos son:

- Negación (Not)
- Conjunción (And)
- Disyunción (Or)

El resultado de estas operaciones es el correspondiente a las conocidas tablas de verdad.

Los operadores relacionales y lógicos mencionados anteriormente, que permiten comparar dos valores, dan como resultado un valor lógico.

Ejemplo:

(8 < 4)	da como resultado <i>falso</i>
(2.5 * 4 = 10)	da como resultado <i>verdadero</i>
(8 > 3)	da como resultado <i>verdadero</i>
not (8 > 3)	da como resultado <i>falso</i>
(7 > 2.4), (9 = 3)	son expresiones <i>verdaderas</i> y <i>falsa</i>
respectivamente	
(7 > 2.4) and (9 = 3)	una expresión que da un resultado <i>falso</i>

Tabla

2.1.

Operación	Operador	Simbolización
Conjunción	Y / AND	\wedge
Disyunción	O / OR	\vee
Negación	NO / NOT	\sim

Tabla 2.1. Conectores Lógicos

Tablas de verdad

Para poder analizar cualquier proposición compuesta y decir qué valor de verdad tiene, es usual hacerlo a través de lo que se conoce como tabla de verdad, la cual se define de la siguiente manera:

La tabla de verdad de una proposición es, como su nombre lo indica, una tabla donde se muestran todas las combinaciones posibles de los valores de verdad de dicha proposición.

Conjunción

Dadas dos proposiciones cualquiera p y q, la proposición molecular $p \wedge q$ representa la conjunción de p y q.

La conjunción de dos proposiciones es cierta únicamente en el caso en que ambas proposiciones lo sean.

p	q	$p \wedge q$
V	V	V

V	F	F
F	V	F
F	F	F

Disyunción

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

Negación

Dada una proposición P su negación $\sim P$ permitirá obtener el valor de verdad opuesto.

El valor de verdad de la negación de una proposición verdadera es falso y el valor de verdad de la negación de una proposición falsa es verdadero.

p	$\sim p$
V	F
F	V

Ejemplos de Expresiones Lógicas

- $(8 < 4)$ da como resultado falso
- $(2.5 * 4 = 10)$ da como resultado verdadero
- $(8 > 3)$ da como resultado verdadero
- NOT $(8 > 3)$ da como resultado falso
- $(7 > 2.4)$ AND $(9=3)$ es una expresión que da un resultado falso
- $(7 > 2.4)$ OR $(9=3)$ es una expresión que da un resultado verdadero

Orden de Evaluación en la expresiones

Operador	Prioridad
NOT	Más alta (se evalúa primero)
$*$, $/$, AND	↓
$+$, $-$, OR	↓
$<$, $>$, $<=$, $>=$, $=$, $<>$,	Más baja (se evalúa al último)
Si existen paréntesis, las expresiones de su interior se evalúan primero	

TIPOS DE DATOS ESTRUTURADOS

**CAPITULO 7 del Libro de Joyanes Aguilar, puntos 7.1, 7.2, 7.3 y 7.4
(El libro digital está en el Aula Virtual)**