INFORMATORIO- MÓDULO I: FUNDAMENTOS DE PROGRAMACIÓN

Teoría

Elaborado por Noelia Pinto – Blas Cabas Geat

Tabla de contenido

Unidad I: Introducción a las Computadoras 3
Contenido Temático 3
Qué es una Computadora 4
Concepto 4
Historia 4
El hardware. Organización física de una computadora 5
Introducción 6
El Hardware de la Computadora: Componentes Básicos 8
Representación de la información en las computadoras 12
Introducción 12
Tipos de Representación de la Información 12
El software. Los programas y computadoras 16
Introducción 16
Software del Sistema 17
Software de Aplicación 17
El sistema operativo 19
Concepto 19
Funciones de los Sistemas Operativos 20
Clasificación de los Sistemas Operativos 21
Conceptos de proceso, algoritmo y programa 23
Concepto de Proceso 23
Concepto de Algoritmo 23
Concepto de Programa 24
Primeros pasos en programación 26
Programación estructurada VS Programación orientada a objetos 27
Propiedades de la Programación Orientada a Objetos 28
Lenguajes de Programación: Python 32

Unidad I: Introducción a las Computadoras

Contenido Temático

a. ¿Qué es una Computadora?
b. El Hardware. Organización física de una computadora.
c. Representación de la información en las computadoras.
d. El software. Los programas y las computadoras.
e. Sistema operativo.
f. Concepto de Proceso, Programa y Algoritmo.
g. Primeros pasos en Programación. Programación Estructurada y Orientada a Objetos. Lenguajes de Programación. Ejemplo: Python.

Qué es una Computadora

Concepto

Una **computadora** es un dispositivo electrónico utilizado para procesar información y obtener resultados. Los datos y la información se pueden introducir en la computadora por la **entrada** (*input*) y a continuación se procesan para producir una **salida** (*output*, resultados).

La computadora se puede considerar como una unidad en la que se ingresan ciertos datos (entrada de datos), seprocesa estos datos y se obtienen datos de salida. Los datos de entrada y los datos de salida pueden ser realmente cualquier cosa, texto, dibujos o sonido. El sistema más sencillo de comunicarse una persona con la computadora es esencialmente mediante un ratón (mouse), un teclado y una pantalla (monitor). Pero hoy en día existen diversas formas de interacción con las computadoras.

Historia

La computación es el resultado de una serie de hitos que evolucionaron en el tiempo, desde los orígenes del hombre hasta la actualidad.

La primera máquina de calcular mecánica, llamada **Pascalina**, origen de la computadora digital, fue inventada en 1642 por el matemático francés ***Blaise Pascal***. Aquel dispositivo utilizaba una serie de ruedas de diez dientes en las que cada uno de los dientes representaba un dígito del 0 al 9. Las ruedas estaban conectadas de tal manera que podían sumarse números haciéndolas avanzar el número de dientes correcto. Pero esta máquina tenía el inconveniente que sólo podía sumar.

Fue así que en 1670 el filósofo y matemático alemán ***Gottfried Wilhelm Leibniz*** perfeccionó esta máquina e inventó una que también podía multiplicar mediante sumas sucesivas, y con un sistema escalonado de engranajes, se llamó la **Multiplicadora de Leibniz**.

El inventor francés ***Joseph Marie Jacquard***, al diseñar un telar automático, utilizó delgadas placas de madera perforadas para controlar el tejido utilizado en los diseños complejos. El artilugio utilizaba tarjetas perforadas para conseguir tejer patrones en la tela, permitiendo que hasta los usuarios más inexpertos pudieran elaborar complejos diseños.

Durante la década de 1880 el estadístico estadounidense ***Herman Hollerith*** concibió la idea de utilizar tarjetas perforadas, similares a las placas de Jacquard, para procesar datos. Hollerith consiguió compilar la información estadística destinada al censo de población de 1890 de Estados Unidos mediante la utilización de un sistema que hacía pasar tarjetas perforadas sobre contactos eléctricos.

Luego de todos los inventos antes mencionados, finalmente la historia reconoció al ***Dr. John Atanasoff*** como el "Invento de la computadora digital electrónica", ayudado por un estudiante universitario, Clifford Berry. El Dr. Atanasoff, catedrático de la Universidad Estatal de Iowa, desarrolló la primera computadora entre los años 1937-1942. Llamó a su invento la computadora **Atanasoff-Berry** (ABC – Atanasoff Berry Computer).

En 1944 se construyó en la Universidad de Harvard, la **Mark I**, diseñada por un equipo encabezado por ***Howard H. Aiken***. Esta computadora tomaba seis segundos para efectuar una multiplicación y doce para una división. Su estrcutura se basaba en rieles (tenía aprox. 3000), con 800 kilómetros de cable, con dimensiones de 17 metros de largo, 3 metros de alto y 1 de profundidad. Al Mark I se le hicieron mejoras sucesivas, obteniendo así el Mark II, Mark III y Mark IV.

Figura 1 – Computadora y Dispositivos de Entrada-Salida

El hardware. Organización física de una computadora

Introducción

Los componentes físicos que constituyen la computadora, junto con los dispositivos que realizan las tareas de entrada y salida, se conocen con el término **hardware**. El conjunto de instrucciones que hacen funcionar a la computadora se denomina **programa**, que se encuentra almacenado en su memoria; a la persona que escribe programas se llama **programador** y al conjunto de programas escritos para una computadora se llama **software**.

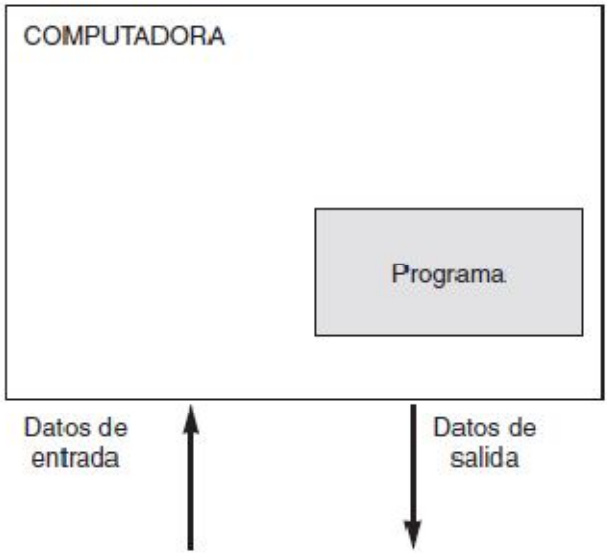

Figura 2 - Flujo de procesamiento en la computadora

La mayoría de las computadoras, grandes o pequeñas, están organizadas como se muestra en la Figura 3. Constan fundamentalmente de tres componentes principales**: Unidad Central de Proceso** (UCP) o procesador (compuesta de la UAL, Unidad Aritmética y Lógica, y la UC, Unidad de Control); la **memoria principal** o central y el **programa**.

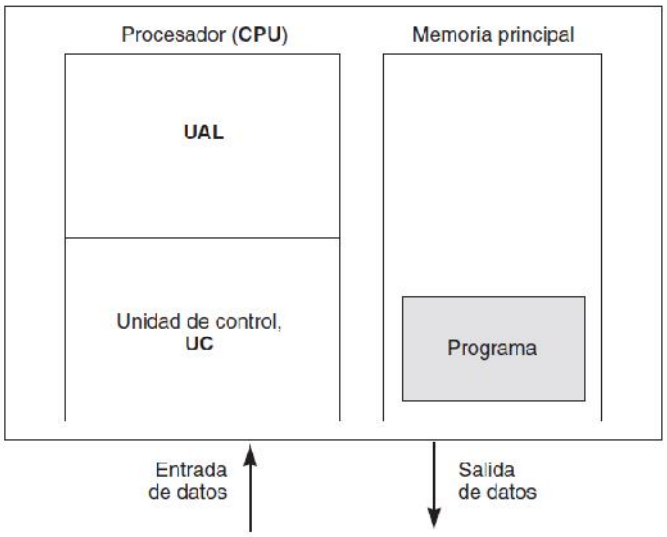

Figura 3 - Organización física de una computadora

Una computadora solo es capaz de ejecutar órdenes y de realizar operaciones tan básicas como sumar, restar, multiplicar y dividir valores numéricos, comparar valores numéricos o alfanuméricos, almacenar o recuperar la información.

Por tanto, su potencia y capacidad de cálculo dependerá básicamente de su eficacia, fiabilidad, rapidez y precisión, así como de la memoria disponible (tanto principal como auxiliar (la principal es la RAM y la auxiliar los soportes de almacenamiento permanente)). Con la combinación de operaciones básicas, una computadora puede llegar a realizar operaciones o cálculos verdaderamente complejos, pero siempre existirá una estrecha dependencia de la máquina con el hombre, y sin la cual la computadora es una herramienta carente de utilidad, pues el hombre es quien marca las pautas para su correcto funcionamiento a través de la lógica y el razonamiento.

Por lo tanto, si a la organización física de la Figura 3 se le añaden los dispositivos para comunicación con la computadora, aparece la estructura típica de un sistema de computadora: **dispositivos de entrada, dispositivos de salida**, **memoria externa** y el **procesador/memoria central** con su programa.

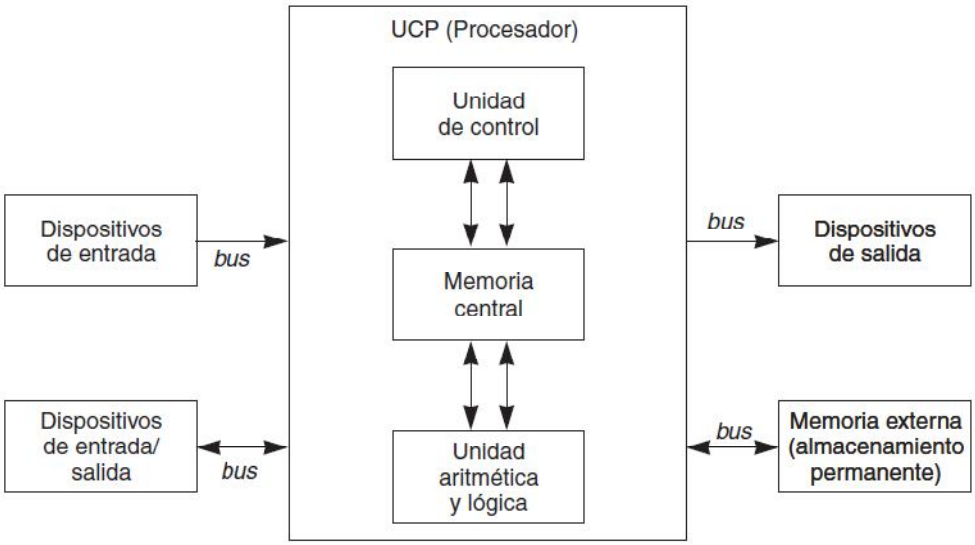

Figura 4 – Estructura típica de un sistema de computadora

El Hardware de la Computadora: Componentes Básicos

1) *Dispositivos de Entrada, de Salida y de Entrada-Salida*

   a. **Dispositivos de Entrada**

      Son aquéllos que sirven para introducir datos y órdenes en la computadora para su procesamiento. Entre los más usuales encontramos al teclado, mouse, lector de códigos de barra, entre otros.

      Los datos se leen de los dispositivos de entrada y se almacenan en lamemoria central o interna.

   b. **Dispositivos de Salida**

      Son los encargados de proporcionar los resultados del proceso interno. Por ejemplo, el monitor, la impresora, los parlantes, etc.

   c. **Dispositivos de Entrada-Salida**

      Son aquéllos capaces de realizar las operaciones anteriores indistintamente. El ejemplo más clásico es el de los CD's o DVD's. Podemos agregar también Unidades de

almacenamiento, USB, cardreader, o dispositivos actuales como las pantallas touch, etc.

2) *Memoria*

Se pueden diferenciar principalmente dos tipos de memoria: Memoria Central o Principal (RAM) y Memoria Auxiliar, constituida por dispositivos de almacenamiento masivo de información.

a) Memoria Central

La **memoria central** o simplemente **memoria** (*interna o principal*) puede ser de dos tipos: RAM (Random Access Memory) que se utiliza para almacenar información y ROM (ReadOnlyMemory) que viene grabada de fábrica y consta de la rutina de arranque o POST y de la rutina BIOS (Basic Input/Output System). Nosotros vamos a enfocarnos en el estudio de la memoria RAM.

En general, la información almacenada en memoria puede ser de dos tipos: *instrucciones*, de un programa y *datos* con los que operan las instrucciones. Por ejemplo, para que un programa se pueda *ejecutar*, debe ser situado en la memoria central, en una operación denominada *carga* (*load*) del programa. Después, cuando se ejecuta el programa, cualquier dato a procesar por el programa se debe llevar a la memoria mediante las instrucciones del programa. En la memoria central, hay también datos diversos y espacio de almacenamiento temporal que necesita el programa cuando se ejecuta a fin de poder funcionar.

Con el objetivo de que el procesador pueda obtener los datos de la memoria central más rápidamente, normalmente todos los procesadores actuales (muy rápidos) utilizan una *memoria* denominada *caché* que sirve para almacenamiento intermedio de datos entre el procesador y la memoria principal.

La memoria central de una computadora es una zona de almacenamiento organizada en centenares o millares de unidades de almacenamiento individual o celdas. La memoria central consta de un conjunto de celdas de memoria (o palabras).

Cada celda de memoria consta de un cierto número de bits (normalmente 8, un byte). La unidad elemental de memoria se llama byte (octeto). Un byte tiene la capacidad de almacenar

un carácter de información, y está formado por un conjunto de unidades más pequeñas de almacenamiento denominadas bits, que son dígitos binarios (0 o 1).

Existen dos conceptos importantes asociados a cada celda o posición de memoria: su dirección y sucontenido. Cada celda o byte tiene asociada una única dirección que indica su posiciónrelativa en memoriay mediante la cual se puede acceder a la posición para almacenar orecuperar información. Lainformación almacenada en una posición de memoria es su contenido.

La memoria central de una computadora puede tener desde unos centenares de millares de bytes hasta millones de bytes. Como el byte es una unidad elemental de almacenamiento, se utilizan múltiplos de potencia de 2 para definir el tamaño de la memoria central:

Kilo-byte (KB o Kb) igual a 1.024 bytes (210), prácticamente se consideran 1.000

Megabyte (MB o Mb) igual a 1.024 × 1.024 bytes = 1.048.576 (220), prácticamente se consideran 1.000.000

Gigabyte (GB o Gb) igual a 1.024 MB (230) = 1.073.741.824, prácticamente se consideran 1.000 millones de MB.

b) Memoria Auxiliar

También conocida como *memoria externa*, son dispositivos de almacenamiento masivo; los datos y programas pueden quedar almacenados de forma permanente, dando opción al usuario a recuperarlos en próximas sesiones de trabajo.

Comparación de la memoria central y la memoria externa

La memoria central o principal es mucho más rápida y cara que la memoria externa. Se deben transferir los datos desde la memoria externa hasta la memoria central, antes de que puedan ser procesados. Los datos en memoria central son: *volátiles* y desaparecen cuando se *apaga* la computadora. Los datos en memoria externa son *permanentes* y no desaparecen cuando se *apaga* la computadora.

3) *Unidad Central de Procesos (CPU)*

La Unidad Central de Proceso, UCP (Central ProcessingUnit, CPU, en inglés), dirige y controla el proceso de información realizado por la computadora. La UCP procesa o manipula la información almacenada en memoria; puede recuperar información desde memoria (esta información son datos o instrucciones: programas). También puede almacenar los resultados de estos procesos en memoria para su uso posterior.

La UCP consta de dos componentes: unidad de control (UC) y unidad aritmética-lógica (UAL). La unidad de control (Control Unit, CU) coordina las actividades de la computadora y determina qué operaciones se deben realizar y en qué orden; asimismo controla y sincroniza todo el proceso de la computadora. La unidad aritmético-lógica (Arithmetic-LogicUnit, ALU) realiza operaciones aritméticas y lógicas, tales como suma, resta, multiplicación, división y comparaciones. Los datos en la memoria central se pueden leer (recuperar) o escribir (cambiar) por la UCP.

4) *El Microprocesador*

El microprocesador es un chip (un circuito integrado) que controla y realiza las funciones y operaciones con los datos. Se suele conocer como procesador y es el cerebro y corazón de la computadora. En realidad el microprocesador representa a la Unidad Central de Proceso de una computadora.

El primer microprocesador comercial, el Intel 4004 fue presentado el 15 de noviembre de 1971.

Representación de la información en las computadoras

Introducción

Es importante estudiar la forma de representación de la información en dicha computadora. Es necesario considerar cómo se puede codificar la información en patrones de bits que sean fácilmente almacenables y procesables por los elementos internos de la computadora.

Las formas de información más significativas son: textos, sonidos, imágenes y valores numéricos y, cada una de ellas presentan peculiaridades distintas.

Tipos de Representación de la Información

1) Representación de Textos

La información en formato de texto se representa mediante un código en el que cada uno de los distintos símbolos del texto (tales como letras del alfabeto o signos de puntuación) se asignan a un único patrón de bits. El texto se representa como una cadena larga de bits en la cual los sucesivos patrones representan los sucesivos símbolos del texto original.

Se puede representar cualquier información escrita (texto) mediante caracteres. Loscaracteres que se utilizan en computación suelen agruparse en cinco categorías:

a. *Caracteres alfabéticos* (letras mayúsculas y minúsculas)

A, B, C, D, E,....., g, h, i,...

b. *Caracteres numéricos* (dígitos del sistema de numeración)

0, 1, 2, 3, 4, 5, 6, 7, 8, 9 sistema decimal

c. *Caracteres especiales* (símbolos ortográficos y matemáticos no incluidos en los grupos anteriores)

{ } ! ? <> # ...

d. *Caracteres geométricos y gráficos*

| ¬ ...

e. *Caracteres de control (representan órdenes de control)*

Por ejemplo, salto de línea NL

Al introducir un texto en una computadora, a través de un periférico, los caracteres se codifican según un **código de entrada/salida** de modo que a cada carácter se le asocia una determinada combinación de *n* bits.

Los códigos más utilizados en la actualidad son: **EBCDIC**, **ASCII** y **Unicode**.

- **Código EBCDIC** (*Extended Binary Coded Decimal Inter Change Code*).

Este código utiliza *n = 8* bits de forma que se puede codificar hasta *m* = 28 = 256 símbolos diferentes. Éste fue el primer código utilizado para computadoras, aceptado en principio por IBM.

- **Código ASCII** (*American Standard Code for Information Interchange*)

El código ASCII básico utiliza 7 bits y permite representar 128 caracteres (letras mayúsculas y minúsculas del alfabeto inglés, símbolos de puntuación, dígitos de 0 a 9 y ciertos controles). Este código es el más utilizado en computadoras, aunque el ASCII ampliado con 8 bits permite llegar a 2^8 (256) caracteres distintos, entre ellos ya símbolos y caracteres especiales de otros idiomas como el español.

- **Código Unicode**

Aunque ASCII ha sido y es dominante en los caracteres se leen como referencia, hoy día se requiere de la necesidad de representación de la información en muchas otras lenguas, como el portugués, español, chino, el japonés, el árabe, etc. Este código utiliza un patrón único de 16 bits para representar cada símbolo, que permite 216 bits o sea hasta 65.536 patrones de bits (símbolos) diferentes.

2) Representación de Valores Numéricos

La solución que se adopta para la representación de datos numéricos es la siguiente: al introducir un número en la computadora se codifica y se almacena como un texto o cadena de caracteres, pero dentro del programa a cada dato se le envía un tipo de dato específico y es tarea del programador asociar cadadato al tipo adecuado correspondiente a las tareas y operaciones que se vayan a realizar con dicho dato.

El método práctico realizado por la computadora es que una vez definidos los datos numéricos de unprograma, una rutina (función interna) de la biblioteca del compilador (traductor) del lenguaje

de programaciónse encarga de transformar la cadena de caracteres que representa el número en su notaciónbinaria.

Se pueden representar números enteros y números decimales.

Los datos de tipo entero se representan en el interior de la computadora en notación binaria. La memoria ocupada por los tipos enteros depende del sistema, pero normalmente son dos, bytes (en las versiones de MS-DOS y versiones antiguas de Windows y cuatro bytes en los sistemas de 32 bits como Windows o Linux).

Los números reales son aquellos que contienen una parte decimal como 2,6 y 3,14152. Los reales se representan en notación científica o en coma flotante; por esta razón en los lenguajes de programación, como C++, se conocen como números en coma flotante.

3) Representación de Imágenes

Las imágenes se adquieren mediante periféricos especializados tales como escáners, cámaras digitales de vídeo, cámaras fotográficas, etc. Una imagen, al igual que otros tipos de información, se representa por patrones de bits, generados por el periférico correspondiente.

Mapa de Bits

En las técnicas de mapas de bits, una imagen se considera como una colección de puntos, cada uno de los cuales se llama pixel (abreviatura de «pictureelement»). Una imagen en blanco y negro se representa como una cadena larga de bits que representan las filas de píxeles en la imagen, donde cada bit es bien 1 o bien 0, dependiendo de que el pixel correspondiente sea blanco o negro. En el caso de imágenes en color, cada pixel se representa por una combinación de bits que indican el color de los pixel. Cuando se utilizan técnicas de mapas de bits, el patrón de bits resultante se llama mapa de bits, significando que el patrón de bits resultante que representa la imagen es poco más que un mapa de la imagen.

Mapa de Vectores

Otros métodos de representar una imagen se fundamentan en descomponer la imagen en una colección de objetos tales como líneas, polígonos y textos con sus respectivos atributos o detalles (grosor, color, etc.).

Formato	Origen y descripción
BMP	**Microsoft**. Formato sencillo con imágenes de gran calidad pero con el inconveniente de ocupar mucho (no útil para la web).
JPEG	Grupo **JPEG**. Calidad aceptable para imágenes naturales. Incluye compresión. Se utiliza en la web.
GIF	**CompuServe**. Muy adecuado para imágenes no naturales (logotipos, banderas, dibujos anidados...). Muy usuado en la web.

Figura 5 – Mapas de Bits

Formato	Descripción
IGES	ASME/ANSI. Estándar para intercambio de datos y modelos de (AutoCAD,...).
Pict	Apple Computer. Imágenes vectoriales.
EPS	Adobe Computer.
TrueType	Apple y Microsoft para EPS.

Figura 6 – Mapas de Vectores

4) Representación de Sonidos

El método más genérico de codificación de la información de audio para almacenamiento y manipulación en computadora es mostrar la amplitud de la onda de sonido en intervalos regulares y registrar las series de valores obtenidos. La señal de sonido se capta mediante micrófonos o dispositivos similares y produce una señal analógica que puede tomar cualquier valor dentro de un intervalo continuo determinado. En un intervalo de tiempo continuo se dispone de infinitos valores de la señal analógica, que es necesario almacenar y procesar, para lo cual se recurre a una técnica de muestreo.

El software. Los programas y computadoras

Introducción

El **software** de una computadora es un conjunto de instrucciones de programa detalladas que controlan y coordinan los componentes hardware de una computadora y controlan las operaciones de un sistema informático. El auge de las computadoras en el siglo pasado y en el actual siglo XXI, se debe esencialmente, al desarrollo de sucesivas generaciones de software potente y cada vez más fácil de usar.

Las operaciones que debe realizar el hardware son especificadas por una lista de instrucciones, llamadas **programas**, o **software**. **Un programa de software** es un conjunto de sentencias o instrucciones a la computadora. El proceso de escritura o codificación de un programa se denomina **programación** y las personas que se especializan en esta actividad se denominan **programadores**. Existen dos tipos importantes de software: software del sistema y software de aplicaciones. Cada tipo realiza una función diferente.

Software del sistema es un conjunto generalizado de programas que gestiona los recursos de la computadora,tal como el procesador central, enlaces de comunicaciones y dispositivos periféricos.

Software deaplicaciones son el conjunto de programas escritos por empresas o usuarios individuales o en equipo yque instruyen a la computadora para que ejecute una tarea específica.

Los dos tipos de software están relacionados entre sí, de modo que los usuarios y los programadores pueden hacer así un uso eficiente del computador. En la figura que se muestra abajo, se muestra una vista organizacional de un computador donde muestran los diferentes tipos de software a modo de capas de la computadora desde su interior (el hardware) hasta su exterior (usuario): Las diferentes capas funcionan gracias a las instrucciones específicas (instrucciones máquina) que forman parte del software del sistema y llegan al software de aplicación, que es utilizado por el usuario que no requiere ser un especialista.

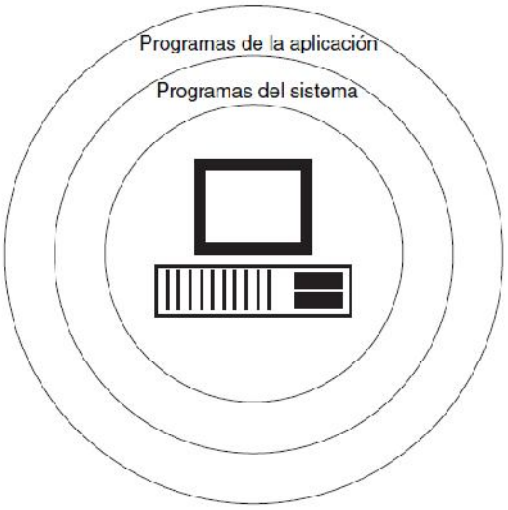

Figura 7 - Relación entre programas de aplicación y de sistema

Software del Sistema

El software del sistema coordina las diferentes partes de un sistema de computadora y conecta e interactúa entre el software de aplicación y el hardware de la computadora. Por ejemplo, el software del sistema que gestiona y controla las actividades de la computadora se denomina ***sistema operativo***. Otro software del sistema son los ***compiladores*** que convierten los lenguajes de programación, entendibles por los programadores, en lenguaje máquina que entienden las computadoras.

El Software del Sistema es el Conjunto de programas necesarios para que la máquina funcione, reciben también el nombre de Programas del Sistema. Ejemplos: Sistema Operativo, Editor de textos, compiladores, entre otros.

Software de Aplicación

El software de aplicación tiene como función principal asistir y ayudar a un usuario de una computadora para ejecutar tareas específicas. Los programas de aplicación se pueden desarrollar con diferentes lenguajes y herramientas de software.

Los lenguajes de programación sirven para escribir programas que permitan la comunicación usuario/máquina. Por ejemplo: Python, Java, C++, entre otros.

Los *programas de utilidad* facilitan el uso de la computadora. Un buen ejemplo es un editor de textos que permite la escritura y edición de documentos, por ejemplo Microsoft Word.

Los programas que realizan tareas concretas, nóminas, contabilidad, análisis estadístico, etc., se denominan *programas de aplicación*. Por ejemplo, el sistema Tango.

El sistema operativo

Concepto

Un *Sistema Operativo* es el software encargado de ejercer el control y coordinar el uso del hardware entre diferentes programas de aplicación y los diferentes usuarios. Es un *administrador* de los recursos de hardware del sistema.

En una definición informal es un sistema que consiste en ofrecer una distribución ordenada y controlada de los procesadores, memorias y dispositivos de E/S entre los diversos programas que requieren su uso.

El sistema operativo es el *administrador principal* de la computadora, y por ello a veces, se lo compara con el director de una orquesta ya que es el responsable de dirigir todas las operaciones de la computadora y gestionar todos sus recursos.

Los sistemas operativos realizan tareas básicas, tales como reconocimiento de la conexión del teclado, enviar la información a la pantalla, y controlar los dispositivos periféricos tales como impresoras, escáner, etc.

En sistemas grandes, el sistema operativo tiene incluso mayor responsabilidad y poder, es como un policía de tráfico, se asegura de que los programas y usuarios que están funcionando al mismo tiempo no interfieran entre ellos. El sistema operativo también es responsable de la seguridad, asegurándose de que los usuarios no autorizados no tengan acceso al sistema.

El sistema operativo *asigna recursos, planifica el uso de recursos* y *tareas* de la computadora, y *monitoriza* a las actividades del sistema informático. Estos recursos incluyen memoria, dispositivos de E/S (Entrada/Salida), y la UCP (Unidad Central de Proceso). El sistema operativo proporciona servicios tales como asignar memoria a un programa y manipulación del control de los dispositivos de E/S tales como el monitor el teclado o las unidades de disco.Cuando un usuario interactúa con un computador, la interacción está controlada por el sistema operativo. Un usuario se comunica con un sistema operativo a través de una interfaz de usuario de ese sistema operativo. Los sistemas operativos modernos utilizan una *interfaz gráfica de usuario*, **IGU** (**G**raphical**U**ser**I**nterface, **GUI**) que hace uso masivo de iconos, botones, barras y cuadros de

diálogo para realizar tareas que se controlan por el teclado o el ratón (mouse) entre otros dispositivos.

Funciones de los Sistemas Operativos

- **Administración del procesador**: el sistema operativo administra la distribución del procesador entre los distintos programas por medio de un algoritmo de programación. El tipo de programador o planificador depende del sistema operativo, según el objetivo deseado.
- **Gestión de la memoria de acceso aleatorio**: el sistema operativo se encarga de gestionar el espacio de memoria asignado para cada aplicación y para cada usuario, si resulta pertinente. Cuando la memoria física es insuficiente, el sistema operativo puede crear una zona de memoria en el disco duro, denominada "memoria virtual". La memoria virtual permite ejecutar aplicaciones que requieren una memoria superior a la memoria RAM disponible en el sistema. Sin embargo, esta memoria es mucho más lenta.
- **Gestión de entradas/salidas**: el sistema operativo permite unificar y controlar el acceso de los programas a los recursos materiales a través de los drivers (también conocidos como administradores periféricos o de entrada/salida).
- **Gestión de ejecución de aplicaciones**: el sistema operativo se encarga de que las aplicaciones se ejecuten sin problemas asignándoles los recursos que éstas necesitan para funcionar.
- **Administración de autorizaciones**: el sistema operativo se encarga de la seguridad en relación con la ejecución de programas garantizando que los recursos sean utilizados sólo por programas y usuarios que posean las autorizaciones correspondientes.
- **Gestión de archivos**: el sistema operativo gestiona la lectura y escritura en el sistema de archivos, y las autorizaciones de acceso a archivos de aplicaciones y usuarios.
- **Gestión de la información**: el sistema operativo proporciona cierta cantidad de indicadores que pueden utilizarse para diagnosticar el funcionamiento correcto del equipo.

Clasificación de los Sistemas Operativos

Los sistemas operativos pueden clasificarse de la siguiente forma:

1. Multiusuario: Permite que dos o más usuarios utilicen sus programas al mismo tiempo.
2. Multiprocesador: La técnica de multiprocesamiento consiste en hacer funcionar varios procesadores en forma paralela para obtener un poder de cálculo mayor que el obtenido al usar un procesador de alta tecnología o al aumentar la disponibilidad del sistema (en el caso de fallas del procesador).
3. Multitareas: Un sistema operativo se clasifica como multitareas o multiprocesos cuando muchas "*tareas*" (también conocidas como procesos) se pueden ejecutar al mismo tiempo.

   Estos procesos permanecen activos, en espera, suspendidos, o se eliminan en forma alternativa, según la prioridad que se les haya concedido, o se pueden ejecutar en forma simultánea.

   Un sistema se considera *preventivo* cuando cuenta con un *programador* (también llamado planificador) el cual, según los criterios de prioridad, asigna el tiempo de los equipos entre varios procesos que lo solicitan.
4. Multitramo: Permite que diversas partes de un solo programa funcionen al mismo tiempo.
5. De Tiempo Real: Los sistemas de tiempo real se utilizan principalmente en la industria y son sistemas diseñados para funcionar en entornos con limitaciones de tiempo. Un sistema de tiempo real debe tener capacidad para operar en forma fiable según limitaciones de tiempo específicas; en otras palabras, debe tener capacidad para procesar adecuadamente la información recibida a intervalos definidos claramente (regulares o de otro tipo).

   Para que su funcionamiento sea correcto no basta con que las acciones sean correctas, sino que tienen que ejecutarse dentro del intervalo de tiempo especificado.

   Algunos ejemplos de este tipo de SO son: QNX, VxWorks, RTLinux.

SISTEMA	PROGRAMACIÓN	USUARIO ÚNICO	MULTIUSUARIO	MULTITAREA
DOS	16 bits	X		
Windows 3.1	16/32 bits	X		X
Windows 95/98/Me	32 bits	X		X
Windows NT/2000	32 bits		X	X
Windows XP	32/64 bits		X	X
Windows Server 2003/2008	32/64 bits		X	X
Windows Vista	32/64 bits		X	X
Windows 7	64 bits		X	X
Windows 8	64 bits		X	X
Unix/Linux	32/64 bits		X	X

Tabla 1 – Sistemas Operativos: Clasificación

Conceptos de proceso, algoritmo y programa

Concepto de Proceso

Es la unidad más pequeña de trabajo, individualmente planificable por un SO. Formalmente un proceso es "Una unidad de actividad que se caracteriza por la ejecución de una secuencia de instrucciones, un estado actual, y un conjunto de recursos del sistemas asociados".

Concepto de Algoritmo

Un algoritmo es una secuencia finita de instrucciones, reglas o pasos que describen de modo preciso las operaciones que una computadora debe realizar para ejecutar una tarea determinada en un tiempo finito. En la práctica, un algoritmo es un método para resolver problemas mediante los pasos o etapas siguientes:

1. *Diseño del algoritmo* que describe la secuencia ordenada de pasos —sin ambigüedades— conducentes a la solución de un problema dado (Análisis del problema y desarrollo del algoritmo).
2. *Expresar el algoritmo* como un programa en un lenguaje de programación adecuado. (Fase de codificación).
3. *Ejecución* y *validación* del programa por la computadora.

Para llegar a la realización de un programa es necesario el diseño previo de un algoritmo indicando cómo hace el algoritmo la tarea solicitada, y eso se traduce en la construcción de un algoritmo. El resultado final del diseño es una solución que debe ser fácil de traducir a estructuras de datos y estructuras de control de un lenguaje de programación específico.

Las dos herramientas más comúnmente utilizadas para diseñar algoritmos son: *diagramas de flujo* y *pseudocódigos*.

Las características fundamentales que debe cumplir todo algoritmo son:

- Un algoritmo debe ser *preciso* e indicar el orden de realización de cada paso.
- Un algoritmo debe estar bien *definido*. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.

- Un algoritmo debe ser *finito*. Si se sigue un algoritmo, se debe terminar en algún momento; o sea, debe tener un número finito de pasos.

La definición de un algoritmo debe describir tres partes: *Entrada*, *Proceso* y *Salida*.

Ejemplo 1
Desarrolle un algoritmo que determine si un número es par o impar.

Solución
Primero vamos a indicar cuáles serían las partes necesarias:

- *Entrada*: El número X ingresado por el usuario
- *Proceso*: Dividir por dos el número X.
- *Salida*: Si el resto de la división es 0, entonces el número X es par. Sino es impar.

Ahora lo trasladamos a pseudocódigo (sólo vamos a indicar en este momento del curso cómo describir el proceso, más adelante estudiaremos detalladamente las secciones de un algoritmo)

```
Escribir("Ingrese el Número")
Leer(X)
Si X mod 2 = 0 entonces
       Escribir("El número", X, "es par")
Sino
       Escribir("El número", X, "es impar")
Fin Si
```

Concepto de Programa

Un programa es un conjunto de instrucciones u órdenes basadas en un lenguaje de programación que una computadora interpreta para resolver un problema o una función específica. Es la redacción de un algoritmo en un lenguaje de programación.

Un programa se escribe en *un lenguaje de programación* y las operaciones que conducen a expresar un algoritmo en forma de programa se llaman *programación*. Así pues, los lenguajes utilizados para escribir programas de computadoras son los lenguajes de programación y *programadores* son los escritores y diseñadores de programas. El proceso de traducir un algoritmo en pseudocódigo a un lenguaje de programación se denomina *codificación*, y el algoritmo escrito en un lenguaje de programación se denomina *código fuente*.

En la realidad la computadora no entiende directamente los lenguajes de programación sino que se requiere un programa que traduzca el código fuente a otro lenguaje que sí entiende la máquina directamente, pero muy complejo para las personas; este lenguaje se conoce como *lenguaje*

máquina y el código correspondiente *código máquina*. Los programas que traducen el código fuente escrito en un lenguaje de programación a código máquina se denominan *traductores*.

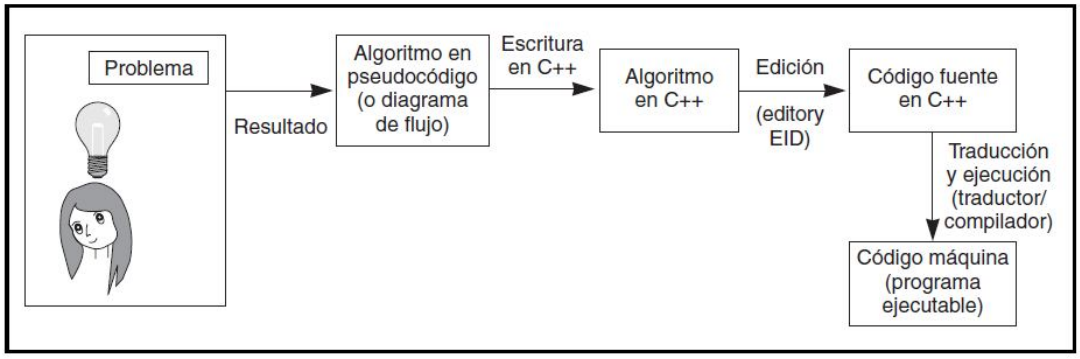

Figura 8 – Proceso de transformación desde pseudocódigo a un programa ejecutable

Qué son los Programas Traductores

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

1. *Intérpretes*

   Un intérprete es un traductor que toma un programa fuente, lo traduce y, a continuación, lo ejecuta. Los programas intérpretes clásicos como BASIC, prácticamente ya no se utilizan, más que en circunstancias especiales. Sin embargo, está muy extendida la versión interpretada del lenguaje Smalltalk, un lenguaje orientado a objetos puro. El sistema de traducción consiste en: traducir la primera sentencia del programa a lenguaje máquina, se detiene la traducción, se ejecuta la sentencia; a continuación, se traduce la siguiente sentencia, se detiene la traducción, se ejecuta la sentencia y así sucesivamente hasta terminar el programa.

2. *Compiladores*

   Un compilador es un programa que traduce los programas fuente escritos en lenguaje de alto nivel a lenguaje máquina. La traducción del programa completo se realiza en una sola operación

denominada compilación del programa; es decir, se traducen todas las instrucciones del programa en un solo bloque. El programa compilado y depurado (eliminados los errores del código fuente) se denomina programa ejecutable porque ya se puede ejecutar directamente y cuantas veces se desee; sólo deberá volver a compilarse de nuevo en el caso de que se modifique alguna instrucción del programa. De este modo el programa ejecutable no necesita del compilador para su ejecución. Los lenguajes compiladores típicos más utilizados son: Java, C, C++, Visual Basic, PHP, entre otros.

Primeros pasos en programación

Programación estructurada VS Programación orientada a objetos

La *programación estructurada* consiste en escribir un programa de acuerdo con unas reglas y un conjunto de técnicas. Las reglas son: el programa tiene un diseño modular, los módulos son diseñados descendentemente, cada módulo de programa se codifica usando tres estructuras de control (secuencia, selección e iteración).

Descomponer un programa en términos de recursos abstractos consiste en descomponer acciones complejas en términos de acciones más simples capaces de ser ejecutadas en una computadora.

El diseño descendente se encarga de resolver un problema realizando una descomposición en otros más sencillos mediante módulos jerárquicos. El resultado de esta jerarquía de módulos es que cada módulo se refina por los de nivel más bajo que resuelven problemas más pequeños y contienen más detalles sobre los mismos.

Las estructuras básicas de control sirven para especificar el orden en que se ejecutarán las distintas instrucciones de un algoritmo. Este orden de ejecución determina el *flujo de control del programa*.

Ventajas

- Los programas son más fáciles de entender, ya que sus líneas pueden leerse de manera secuencial.
- La estructura del programa es más clara, puesto que el código está correlacionado.
- Se reduce el esfuerzo en las pruebas. El seguimiento de los errores en los programas (conocido como “debugging”) se facilita, pues la estructura es más visible, y los errores pueden detectarse y corregirse más sencillamente.

Desventajas

- El principal inconveniente de este método, es que se obtiene un único bloque de programa, que cuando se hace demasiado grande puede resultar problemático su manejo.

La *programación orientada a objetos* aporta un nuevo enfoque a los retos que se plantean en la programación estructurada cuando los problemas a resolver son complejos. Al contrario que la

programación procedimental que enfatiza en los algoritmos, la POO enfatiza en los datos. En lugar de intentar ajustar un problema al enfoque procedimental de un lenguaje, POO intenta ajustar el lenguaje al problema. La idea es diseñar formatos de datos que se correspondan con las características esenciales de un problema. Los lenguajes orientados combinan en una única unidad o módulo, tanto los datos como las funciones que operan sobre esos datos. Tal unidad se llama ***objeto***. Si se desea modificar los datos de un objeto, hay que realizarlo mediante las funciones miembros del objeto. Ninguna otra función puede acceder a los datos. Esto simplifica la escritura, depuración y mantenimiento del programa.

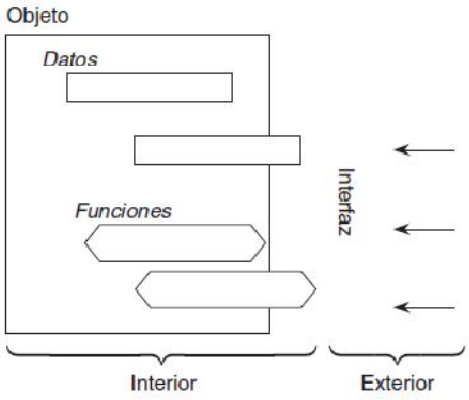

Figura 9 – Diagrama de un objeto

Propiedades de la Programación Orientada a Objetos

1) Abstracción

Es la propiedad de los objetos que consiste en tener en cuenta sólo los aspectos más importantes desde un punto de vista determinado y no tener en cuenta los restantes aspectos. Durante el proceso de abstracción es cuando se decide qué características y comportamiento debe tener el modelo. Un medio de reducir la complejidad es la abstracción. Las características y los procesos se reducen a las propiedades esenciales, son resumidas o combinadas entre sí.

Ejemplo 1.2
Diferentes modelos de abstracción del objeto AUTO:

- Un AUTO es la composición de diferentes partes, tales como, motor, cuatro ruedas, cinco puertas, carrocería, etc
- Un AUTO es un concepto común para diferentes tipos de autos. Pueden clasificarse por su fabricante (AUDI, BMW, Ford, etc), por su categoría (deportivo, 4x4, etc), por el combustible que usan (nafta, gasoil, gas, etc)

La abstracción AUTO se usará siempre que la marca, categoría o combustible no sean significativos. Por ejemplo se podrá utilizar para el proceso "Transportar personas".

2) Encapsulamiento y Ocultamiento de Datos

El encapsulado o encapsulación de datos es el proceso de agrupar datos y operaciones relacionadas bajo la misma unidad de programación. En el caso de que los objetos posean las mismas características y comportamiento se agrupan en clases (unidades o módulos de programación que encapsulan datos y operaciones).

La ocultación de datos permite separar el aspecto de un componente, definido por su interfaz con el exterior, de sus detalles internos de implementación. Los términos ocultación de la información (informationhiding) y encapsulación de datos (data encapsulation) se suelen utilizar como sinónimos, pero no siempre es así y muy al contrario son términos similares pero distintos.

Ejemplo 1.3

Por ejemplo pensemos en un electrodoméstico cualquiera, este electrodoméstico aplica el principio del encapsulamiento, en la medida que solo permite que los usuarios interactúen con él, mediante los elementos de acceso, es decir: los botones de comando y los controles remotos. Sin embargo, internamente el electrodoméstico está lleno de otras funcionalidades que nose necesitan mostrar para lograr utilizar satisfactoriamente el electrodoméstico.

3) Generalización y Especialización

La *generalización* es la propiedad que permite compartir información entre dos entidades evitando la redundancia. En el comportamiento de objetos existen con frecuencia propiedades que son comunes en diferentes objetos y esta propiedad se denomina generalización.

Por ejemplo, lavarropas, heladeras, hornos de microondas, tostadoras, lavavajillas, etc., son todos electrodomésticos. En el mundo de la orientación a objetos, cada uno de estos aparatos es una ***subclase*** de la claseElectrodoméstico y a su vez Electrodoméstico es una ***superclase*** de todas las otras clases (, lavarropas, heladeras, hornos de microondas, tostadoras, lavavajillas, etc.). El proceso inverso de la generalización porel cual se definen nuevas clases a partir de otras ya existentes se denomina *especialización*.

En orientación a objetos, el mecanismo que implementa la propiedad de generalización se denomina **herencia**. La herencia permite definir nuevas clases a partir de otras clases ya existentes, de modo que presentan las mismas características y comportamiento de éstas, así como otras adicionales.

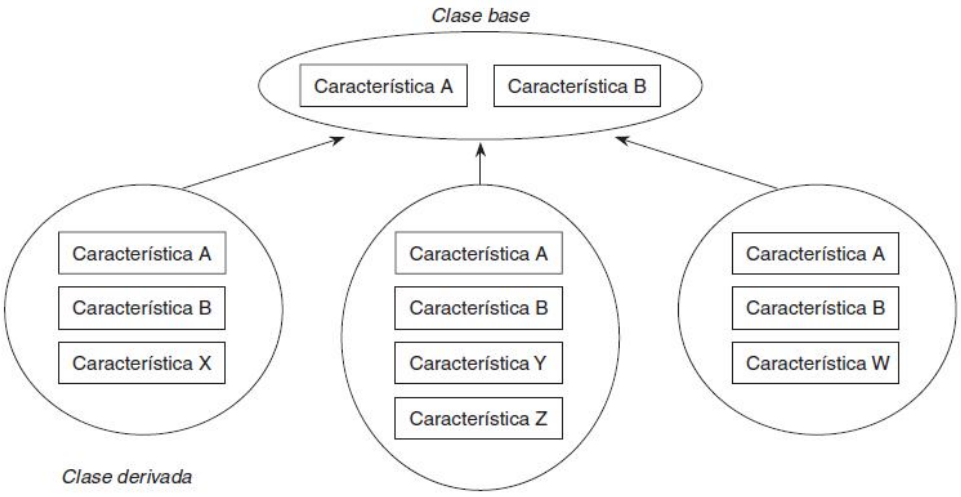

Figura 10 – Jerarquía de clases

Una clase hereda sus características (datos y funciones) de su superclase

4) Polimorfismo

La propiedad de polimorfismo es aquella en que una operación tiene el mismo nombre en diferentes clases, pero se ejecuta de diferentes formas en cada clase. Así, por ejemplo, la operación *abrir* se puede dar en diferentes clases: abrir una puerta, abrir una ventana, abrir un libro, abrir un archivo, abrir una cuenta corriente en un banco, etc. En cada caso se ejecuta una operación diferente aunque tiene el mismo nombre en todos ellos "*abrir*". El polimorfismo es la propiedad de una operación de ser interpretada sólo por el objeto al que pertenece.

Lenguajes de Programación: Python

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses "Monty Python". Es un lenguajecon una sintaxis muy limpia y que favorece un código legible.Se trata de un *lenguaje interpretado* o de *script*, con *tipado dinámico*, *fuertemente tipado*, *multiplataforma* y *orientado a objetos*.

¿Por qué elegimos Python? Python es un lenguaje que todo el mundo debería conocer. Su sintaxis simple, clara y sencilla; el tipado dinámico, el gestor de memoria, la gran cantidad de librerías disponibles y la potencia del lenguaje, entre otros, hacen que desarrollar una aplicación en Python sea sencillo, muy rápido y, lo que es más importante, divertido. La sintaxis de Python es tan sencilla y cercana al lenguaje natural que los programas elaborados en Python parecen pseudocódigo. Por este motivo se trata además de uno de los mejores lenguajes para comenzara programar. Python no es adecuado para la programación de bajo nivel o para aplicaciones en las que el rendimiento sea crítico.

Algunos casos de éxito en el uso de Python son Google, Yahoo, la NASA y todas las distribuciones Linux, en las que Python cada vez representa un tanto por ciento mayor de los programas disponibles.

Lenguaje Interpretado

Un lenguaje interpretado o de script es aquel que se ejecuta utilizando un programa intermedio llamado intérprete, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados). La ventaja de los lenguajes compilados es que su ejecución es más rápida. Sin embargo los lenguajes interpretados son más flexibles y más portables.

Python tiene, no obstante, muchas de las características de los lenguajes compilados, por lo que se podría decir que es *semi interpretado*.

Tipado Dinámico

La característica de tipado dinámico se refiere a que no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo.

Fuertemente Tipado

No se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario convertir de forma explícita dicha variable al nuevo tipo previamente. Por ejemplo, si tenemos una variable que contiene un texto (variable de tipo cadena o string) no podremos tratarla como un número (sumar la cadena "9" y el número 8). En otros lenguajes el tipo de la variable cambiaría para adaptarse al comportamiento esperado, aunque esto es más propenso a errores.

Multiplataforma

El intérprete de Python está disponible en multitud de plataformas (UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc.) por lo que si no utilizamos librerías específicas de cada plataforma nuestro programa podrá correr en todos estos sistemas sin grandes cambios.

Orientado a Objetos

La orientación a objetos es un paradigma de programación en el que los conceptos del mundo real relevantes para nuestro problema se trasladan a clases y objetos en nuestro programa. La ejecución del programa consiste en una serie de interacciones entre los objetos.

REFERENCIAS

- Información disponible en http://robotica.uv.es/pub/Libro/PDFs.zip