Fundamentos de Programación – Introducción a la Programación:
1.- LENGUAJES DE PROGRAMACIÓN

Maider Huarte Arrayago

Fundamentos de Programación – Introducción a la Programación:

1.- LENGUAJES DE PROGRAMACIÓN

1.- LENGUAJES DE PROGRAMACIÓN

1.1.- CONCEPTOS BÁSICOS DE UN COMPUTADOR

Informática: Ciencia o técnica que estudia el tratamiento de la información a través de recursos automáticos.

Computador: Es una herramienta electrónica que puede almacenar y procesar información siguiendo las instrucciones de un programa. Desde el punto de vista de la utilización, es una máquina automática de propósito general, es decir, se pueden ejecutar aplicaciones que hagan cualquier tipo de cálculo o procesen cualquier tipo de información.

Según lo especificado en la definición anterior, un computador realiza su trabajo siguiendo las instrucciones de los programas correspondientes. Por eso, se dice que un computador es *programable*.

Los conceptos relativos a los computadores se agrupan en dos conjuntos:

- **Hardware**: Conjunto de conceptos relativos a la tecnología. Circuitos, buses, motores, memorias, dispositivos periféricos,...
- **Software**: Conjunto de conceptos relativos a la información y a los programas.

1.2.- ESTRUCTURA INTERNA BÁSICA DE COMPUTADORES

1.2.1.- Introducción

Se dice que el inventor y matemático inglés Charles Babbage (s. XIX) fue el padre de los computadores. Inventó la *Máquina de Restar*, con intención de mejorar las tablas logarítmicas de su tiempo. Más tarde, ideó y definió lo que llamó *Máquina Analítica*, para hacer operaciones matemáticas.

Aquella Máquina Analítica de Babbage ya contenía los elementos básicos de las máquinas de propósito general actuales. Sin embargo, las teorías de Babbage eran más avanzadas que la tecnología de su tiempo, con lo cual, la máquina nunca llegó a desarrollarse en la práctica.

Después de más de 100 años desde la Máquina Analítica de Babbage, John Von Neumann diseñó otra máquina, la EDVAC. La aportación más importante de Von Neumann fue la de guardar programas en la memoria de la máquina y controlar el funcionamiento de la máquina mediante esos programas.

Así, la conocida como Máquina de Von Neumann conforma la estructura básica interna de un computador de hoy.

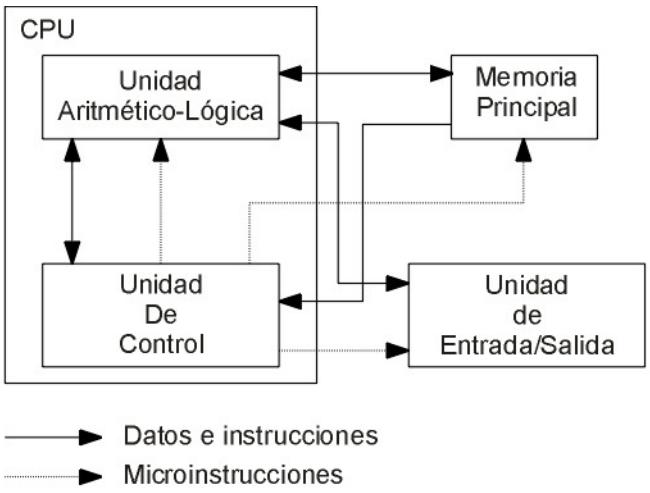

Figura 1: Máquina de Von Neumann

Según se ve en la Figura 1, la Máquina de Von Neumann está formada por los siguientes elementos:

- **Memoria**
- **Unidad de Control**
- **Unidad Aritmético-lógica**
- **Unidad de Entrada/Salida**

1.2.1.1.- Memoria

Es la unidad del computador en la que se almacena la información (se considera información tanto los datos como los programas).

La información se guarda en la memoria de forma digital, es decir, se usa una representación binaria (0s y 1s). La unidad básica de información es el **bit** (**Bi**nary Digi**t**). Mediante la agrupación de bits, se consigue la representación de cualquier número o carácter.

La agrupación de 8 bits es conocida como **byte**, y se usa para la representación de caracteres. Todos los caracteres imprimibles (tanto letras como números, signos de puntuación,...) están tabulados de forma que cada uno tiene asignado un número que en binario, se representa mediante un byte. La tabla más usada es la tabla ASCII.

Otro concepto importante en la Memoria es el concepto de **Dirección**. Una memoria contiene una serie de bits, agrupados en bytes. Cada byte ocupa una

posición en la memoria (1er byte, 2º byte,...). El concepto de dirección permite identificar un byte concreto de la memoria, mediante un número positivo. Así, cuando la CPU quiera leer o escribir un byte en la memoria, tendrá que indicar la dirección en la que se encuentra ese byte.

Ejemplo:

> Si tenemos una memoria de 1Kbyte, eso quiere decir que tenemos 1024 bytes.
> Las direcciones de esa memoria tienen el rango [0,1023].

Dentro de un computador actual, en general, se distinguen dos tipos de memorias, la Memoria Principal y la Memoria Externa (discos duros, disquetes,...).

- **Memoria Principal**: Engloba lo que en principio era el concepto de Unidad de Memoria en la Máquina de Von Neumann. Tiene una parte ROM y otra parte RAM. La parte ROM tiene contenido fijo grabado, que no desaparece aunque se le quite la alimentación. En esta parte están grabadas las primeras instrucciones que han de ejecutarse nada más encender el computador (p. e., reconocimiento de los diferentes elementos que conforman el computador).
  La parte RAM es la que usa la CPU para ejecutar un programa. Es decir, **cuando un computador ejecuta un programa, lo carga primero en la parte RAM de la Memoria Principal, y es desde ahí donde va leyendo las instrucciones** indicadas en el mismo. En caso de que se deje de alimentar (p. e., cuando se apaga el computador), toda la información contenida en él se pierde.

- **Memoria Externa**: Son dispositivos externos a lo que en principio era el concepto de Unidad de Memoria en la Máquina de Von Neumann. Son dispositivos de almacenaje en los que el usuario almacena los diferentes programas y datos que quiere ejecutar en el computador, pero que para hacerlo, tendrá antes que cargarlos en la Memoria Principal. Ejemplos: Discos Duros, disquetes, CDs,...

1.2.1.2.- Unidad de Control

Es la unidad que controla todo el sistema. Lee las instrucciones de un programa desde la Memoria Principal y las interpreta, creando y enviando a la unidad adecuada, las *microinstrucciones* necesarias para su ejecución.

Ejemplo:

> Una instrucción puede ser la de escribir el dato de salida de la Unidad Aritmético-Lógica en una dirección de memoria; la Unidad de Control, genera las instrucciones necesarias para indicar a la Unidad de Memoria que escriba en una dirección concreta el dato que se le indica desde la Unidad Aritmético-Lógica.

Para ello, cuenta entre sus elementos internos con un *Contador de Programa*. El Contador de programa indica la dirección de memoria de la siguiente instrucción que ejecutará la Unidad de Control. Así, cuando se comienza con la ejecución de un programa, la dirección que indica el Contador de Programa será la dirección a partir de la cuál se ha cargado el programa en la Memoria Principal.

En general, la ejecución de los programas es secuencial, es decir, que una vez leída la instrucción, se incrementa el valor del Contador de Programa para que apunte a la siguiente instrucción a ejecutar. Puede ocurrir, que en un momento dado, la ejecución no sea secuencial, sino que hay que saltar a otra dirección cualquiera. Para ello, hay instrucciones especiales que lo que hacen es poner un valor determinado en el registro Contador de Programa.

1.2.1.3.- Unidad de Aritmético-Lógica

Es la unidad, que por orden de la Unidad de Control, realiza operaciones tanto aritméticas como lógicas con datos de la Memoria.

Operaciones aritméticas: suma, resta, multiplicación,...
Operaciones lógicas: AND, OR, NOT, XOR,...

1.2.1.4.- Unidad de Entrada/Salida

Es la Unidad que sirve al computador de interfaz con el exterior. A través de esta unidad, el computador recibe los datos y los programas a ejecutar, y también muestra los resultados de esas ejecuciones.

Ejemplo:

> El esquema de Von Neumann, se puede comparar con la situación de un alumno haciendo los ejercicios de, por ejemplo, Matemáticas, en su escritorio.
>
> El profesor, comunica al alumno los ejercicios que tiene que hacer, mediante la Unidad de Entrada/Salida.
>
> El alumno, cuando decide ponerse a hacer los ejercicios, se sienta cómodamente en su asiento, ordena un poco el escritorio, saca los bolígrafos de colores,... es decir, hace una

serie de preparativos antes de ponerse a hacer nada. Esos preparativos son, al fin y al cabo, como instrucciones que se encuentran en la parte ROM de la Memoria Principal.

El alumno saca los ejercicios de su carpeta, que es el Disco Duro del sistema (Memoria Externa), y los pone sobre la mesa, que es la parte RAM de la Memoria Principal.

Una vez que tiene todo lo necesario en la mesa, comienza a hacer los ejercicios, valiéndose de la calculadora. Eso se puede asemejar a una CPU que va leyendo y ejecutando las instrucciones de un programa en la Memoria Principal, mediante la Unidad Aritmético-Lógica (la calculadora del alumno).

Al terminar los ejercicios, el alumno prepara las respuestas y se las da al profesor, mediante la Unidad de Entrada/Salida.

El ejemplo anterior puede ayudar a asimilar los conceptos de la máquina Von Neumann de una forma más familiar.

Siguiendo con el ejemplo, el alumno podría usar los apuntes de clase o libros de texto para hacer los ejercicios; tendría que buscarlos y ponerlos en la mesa. Esto se asemeja a los programas grandes que necesitan cargar ficheros especiales en la Memoria Principal.

Así, queda clara la importancia de tener una Memoria Principal suficiente en la máquina, ya que si el alumno no tiene sitio en su mesa para poner todo lo que necesita, no podrá hacer bien los ejercicios, o en cualquier caso, tardará más tiempo. Los computadores también, cuanto más ocupada tengan la Memoria Principal, más lento funcionan. Hoy en día, la ocupación de la Memoria Principal no es un gran problema, ya que al ser las memorias RAM muy baratas, la mayoría de los computadores no tienen problemas a la hora de ejecutar programas convencionales. Incluso se pueden tener abiertos varios programas a la vez, sin que haya ningún problema; antes, eso ralentizaba el funcionamiento del computador.

Sin embargo, el ejemplo tiene un fallo: cuando el alumno saca los apuntes de la carpeta y los pone en la mesa, los apuntes ya no están en la carpeta. Si lo quisiéramos asemejar a lo que pasa en un computador, se podría pensar que cuando un programa se pasa del Disco Duro a la Memoria Principal, el fichero del programa desaparece del Disco Duro, pero eso **no** es así. Lo que ocurre cuando decimos la palabra “pasa”, es que se hace una **copia** de lo que hay en el fichero del Disco Duro en la Memoria Principal, y es sobre esa copia sobre la que se trabaja. Sería como decir que el alumno, en vez de poner en su mesa los ejercicios que le ha dado el profesor, sacase una copia y trabajase con ella, manteniendo el original en la carpeta... pero es una forma poco útil de estudiar.

1.3.- INTRODUCCIÓN A LOS LENGUAJES DE PROGRAMACIÓN

Los computadores están formados por elementos electrónicos digitales, es decir, sólo entienden *1s y 0s*. Los lenguajes de programación son necesarios para poder superar la diferencia entre el lenguaje que entiende un computador y el que entiende la persona que lo programa.

Los lenguajes de programación están formados por un conjunto limitado de palabras, símbolos y estructuras sintácticas, que el programador entiende. Para que el computador entienda las instrucciones que el programador le quiere dar a través del programa, se necesitará un proceso intermedio de traducción.

1.3.1.- Lenguaje Máquina

Tal y como hemos mencionado antes, los computadores sólo entienden 1s y 0s. Son capaces de interpretar instrucciones codificadas mediante un código binario. Este tipo de Lenguajes de Programación se conocen como *Código Máquina*, y no son nada fáciles de programar.

Ejemplo:

```
00001111
11101010
11001100
11000001
11100001
```

La única ventaja que podría tener el escribir un programa directamente en Código Máquina, es que no sería necesario traducirlo para que lo entendiera la CPU, ya que está en su propio lenguaje. Hay que tener en cuenta, sin embargo, que las unidades CPU en el mercado son diferentes, y que además, cada una entiende su propio Código Máquina.

Cuando surgieron los primeros computadores, la única forma de programar era hacerlo directamente en Código Máquina.

1.3.2.- Lenguajes de Bajo Nivel

Al ser casi imposible la programación directa en Código Máquina, desde el principio se hicieron esfuerzos por resolver ese problema. Así, surgieron los Lenguajes de Bajo Nivel o Lenguajes Ensamblador, en los que los programadores usaban mnemónicos de instrucciones simbólicas. En este tipo

de lenguajes, cada instrucción se corresponde con un código binario que es, a su vez, una instrucción en Código Máquina.

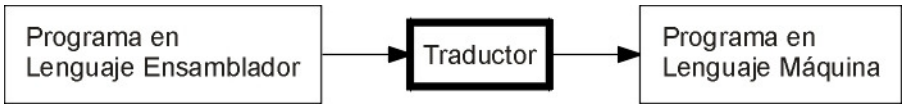

Figura 2: Lenguaje Ensamblador

El fichero escrito en Lenguaje Ensamblador, se traducía, mediante un traductor, al Código Máquina correspondiente.

En el siguiente ejemplo se ve una sentencia escrita en Lenguaje Ensamblador de un procesador concreto:

Etiqueta	**Mnemónico**	**Operando**	**Comentario**
Sentencia_1	LDA	#4A	; Introducir el dato 4A en el Registro Acumulador

En este Lenguaje, el Mnemónico LDA es la abreviatura de ***LoaD A**ccumulator*, que en inglés, significa "cargar en el Acumulador".

Así, con los Lenguajes Ensamblador, los programadores no se tenían que preocupar por los códigos binarios de las instrucciones, y la ejecución de los programas seguía siendo rápida, porque la traducción era inmediata.

Sin embargo, aunque la forma de programar se había mejorado con los Lenguajes Ensamblador, aún seguía siendo bastante difícil, por la gran diferencia con las estructuras de lenguajes humanos. Además, cada CPU tiene su propio Lenguaje Ensamblador.

1.3.3.- Lenguajes de Alto Nivel

Aunque los Lenguajes de Bajo Nivel eran más fáciles de usar que el Código Máquina, distaban mucho de ser cómodos. Un lenguaje de programación útil, tenía que dar facilidades para escribir los programas y para encontrar los fallos en los mismos, posibilitar el entendimiento de un programa escrito por otro programador, proporcionar estructuras de control adecuadas, independencia de la CPU usada,...

Así, surgieron los Lenguajes de Alto Nivel. Esos lenguajes están diseñados para que sean fácilmente entendibles por las personas.

Una instrucción de un Lenguaje de Alto Nivel corresponde a más de una instrucción en Código Máquina, por lo que es necesario, como con los Lenguajes de Bajo Nivel, hacer una traducción de los programas. Esas traducciones se hacen mediante unos programas especiales llamados, en

general, *traductores*. Hay dos tipos de traductores, los *compiladores* y los *intérpretes*.

Sin embargo, el uso de Lenguajes de Alto Nivel tiene una serie de desventajas:

- No se usan de forma óptima los recursos internos de la CPU.
- Se suele necesitar más Memoria.
- El tiempo de ejecución de los programas suele ser mayor.

1.3.3.1.- Compiladores

Los compiladores, cogen un programa escrito en un Lenguaje de Alto Nivel, al que se llama *Programa* o *Fichero Fuente*, y lo traducen a Lenguaje Máquina, generando otro *Programa* o *Fichero Objeto*.

Los Ficheros Objeto generados por algunos compiladores son también ejecutables directamente por el computador. Se dice que en estos casos es suficiente con la *Fase de Traducción*. Otros compiladores sin embargo, no generan Ficheros Objeto ejecutables, por lo que para hacer la ejecución de esos Ficheros Objeto, hace falta otra fase, la *Fase de Enlazado*, que se lleva a cabo con otro programa especial llamado *Enlazador*. En esta segunda fase, se pueden enlazar varios Ficheros Objeto (e incluso, librerías) para generar un único Programa Ejecutable.

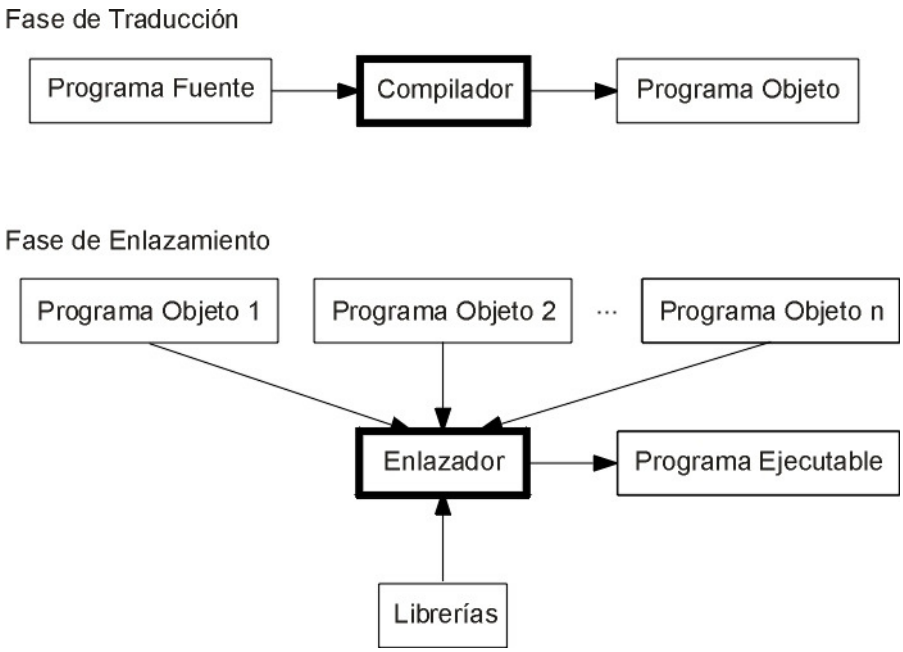

Figura 3: Compiladores

1.3.3.2.- Intérpretes

Los Intérpretes, en vez de traducir a Código Máquina un Fichero Fuente entero, lo que hacen es ir leyendo cada una de las sentencias del Fichero Fuente y según las leen, las traducen y las ejecutan. Así, los Intérpretes no generan Ficheros Objeto.

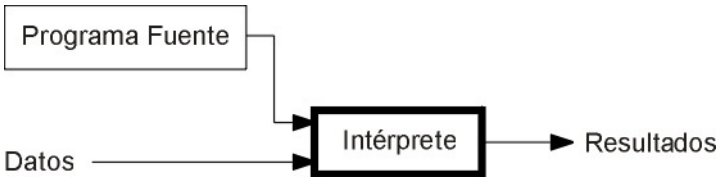

Figura 4: Intérpretes

La forma de ejecutar programas que proporcionan los Intérpretes es muy lenta. No sólo se ejecuta, sino que se hace también la lectura y traducción de cada instrucción.

Hoy en día, se usan más los Compiladores.