UD1 Introducción a la programación

En este tema comenzaremos por dejar claro un conjunto de conceptos básicos, necesarios para empezar a comprender conceptos mas avanzados. Estableceremos las bases de los temas que están por venir, de forma que en cualquier momento podamos volver a este para refrescar ideas.

Contenido

[1. Algoritmo, programación y programa 1](#_Toc145861845)

[2. Lenguajes de programación 7](#_Toc145861846)

[3. Ciclo de vida del desarrollo de software 9](#_Toc145861847)

[4. Metodologías ágiles 9](#_Toc145861848)

[5. Entorno integrado de desarrollo 9](#_Toc145861849)

[6. Repositorios de código y sistemas de control de versiones 9](#_Toc145861850)

# Algoritmo, programación y programa

Definamos primero los conceptos:

* Algoritmo: Conjunto de instrucciones o reglas definidas, no ambiguas, ordenadas y finitas que permiten, solucionar un problema, realizar un cómputo, procesar datos, …
* Programación: Proceso de crear un conjunto de instrucciones que le indican a una computadora como realitzar algún tipo de tarea. O, dicho de otra manera, resolución de problemas a través de algoritmos.
* Programa: Algoritmo escrito en un lenguaje concreto de programación.

Coloquialmente podemos referirnos a programa y programación sin estar relacionados con el contexto de la informática, por ejemplo, como acepción según la RAE (Real Academia Española) programación se refiere al “Conjunto de los programas de un espectáculo, como el teatro, o de la radio o la televisión.”

Pero en cualquier ámbito, su uso nos aporta una idea de su significado: Conjunto de hechos ordenados secuencialmente en el tiempo.

En programación, un algoritmo supone el paso previo a ponerse a escribir el código. Primero debemos encontrar la forma de obtener la solución al problema (definir el algoritmo informático), para luego, a través del código, poder indicarle a la máquina qué acciones queremos que lleve a cabo. No obstante, los algoritmos no son algo exclusivo de los ámbitos de las matemáticas, la lógica y la computación. Utilizamos numerosos algoritmos para resolver problemas en nuestra vida cotidiana. Algunos de los ejemplos más habituales son los manuales de instrucciones o las recetas de cocina.

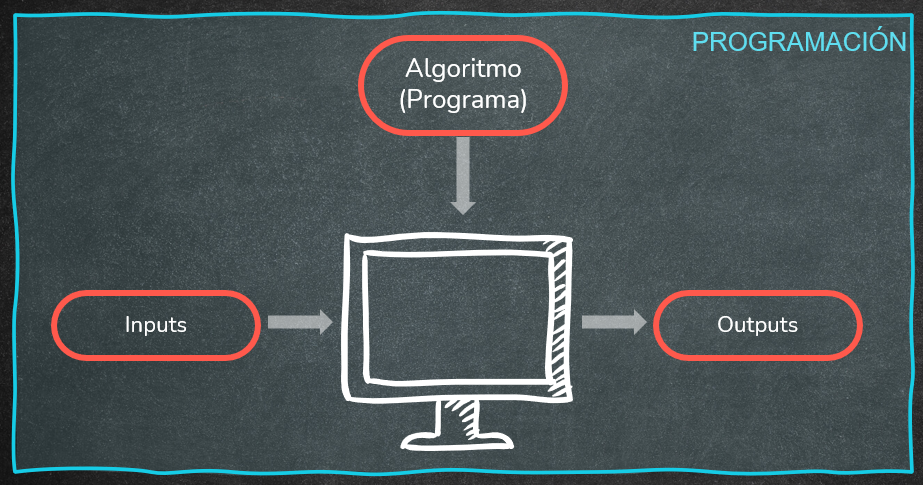
Las tres partes de un algoritmo son:

Input (entrada). Información que damos al algoritmo con la que va a trabajar para ofrecer la solución esperada.

Proceso. Conjunto de pasos para que, a partir de los datos de entrada, llegue a la solución de la situación.

Output (salida). Resultados, a partir de la transformación de los valores de entrada durante el proceso.

De este modo, un algoritmo informático parte de un estado inicial y de unos valores de entrada, sigue una serie de pasos sucesivos y llega a un estado final en el que ha obtenido una solución.



Los algoritmos presentan una serie de características comunes. Son:

* Precisos. Objetivos, sin ambigüedad.
* Ordenados. Presentan una secuencia clara y precisa para poder llegar a la solución.
* Finitos. Contienen un número determinado de pasos.
* Concretos. Ofrecen una solución determinada para la situación o problema planteados.
* Definidos. El mismo algoritmo debe dar el mismo resultado al recibir la misma entrada.

**EJEMPLO**

PROBLEMA A RESOLVER: Hacer una tortilla de patatas.

A continuación, se define un posible algoritmo como solución:

DATOS DE ENTRADA: 4 Huevos, 250grms de patatas, aceite, sartén, fuego o elemento calentador

DATOS DE SALIDA: Tortilla de patatas.

ALGORITMO:

1. Poner el aceite en la sartén
2. Poner la sartén al fuego o elemento calentador
3. **Mientras**, pelar y cortar patatas
4. Cuando el aceite está caliente, freír las patatas
5. Batir los huevos
6. **Cuando** las patatas estén al gusto, sacarlas del aceite y juntarlas con el huevo
7. Vaciar la sartén de aceite
8. Añadir esta mezcla de huevos y patatas a la sartén.
9. Esperar que se cocine por un lado y dar la vuelta
10. Esperar que se cocine por el otro
11. Apagar fuego y extraer tortilla

Algunos ejemplos de problemas que podemos resolver con ayuda de los algoritmos:

* + Determinar si un numero dado es primo
  + Buscar en una lista de palabras, palabras repetidas
  + Dibujar por pantalla un círculo de radio R.
  + Pensar para casa un algoritmo para calcular los metros cuadrados de una habitación.

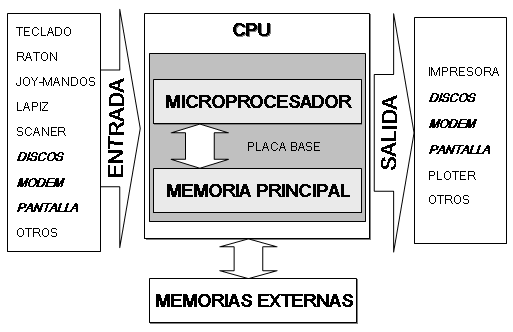
Dependiendo de la envergadura del problema a resolver los algoritmos pueden ser más complejos o menos. Pongamos como ejemplo la diferencia entre los ejercicios que vamos a llegar a resolver en clase y el desarrollo de un Sistema operativo.

Un ordenador es incapaz de hacer nada por sí mismo, para realizar cualquier tarea necesita seguir órdenes. Aunque parezca que las personas no interactúan con él, en realidad desde el minuto uno, como mínimo le hemos tenido que dar una orden y ha sido al presionar el botón de encendido. A partir de ese momento, le hemos introducido el INPUT necesario para empezar a seguir las órdenes del programa (algoritmo) de arranque. Como OUTPUT en este caso, tendremos un sistema de computación a ala escucha. El sistema operativo es un programa que siempre está en ejecución, a la espera de INPUTS por parte del usuario, que activen programas.

Para llevar a cabo estas tareas, un computador puede aceptar diferentes tipos de órdenes. Estas ordenes están limitadas por la capacidad de los componentes del mismo. Para comprenderlo mejor, según el ejemplo visto en clase, un robot de cocina no tiene la capacidad de gratinar, por lo que nunca podrá ejecutar una orden de este tipo.

Es importante tener en cuenta esto a la hora de programar un computador.

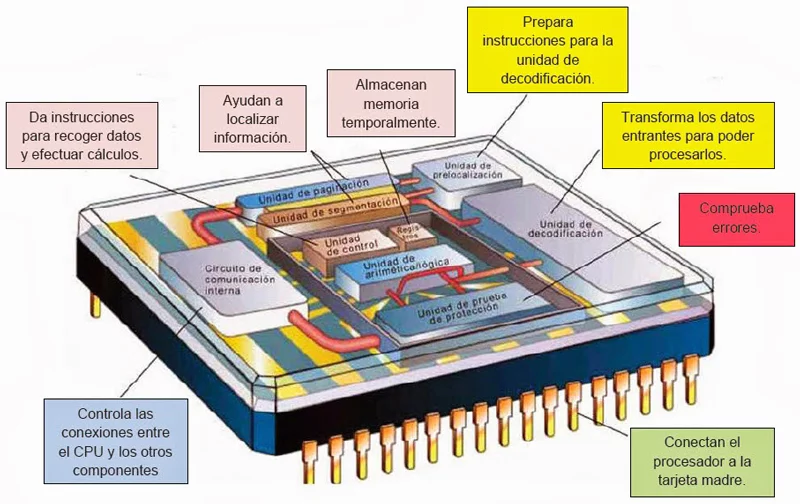
El ordenador está formado por un conjunto de componentes, los cuales se comunican entre ellos. Mostramos a continuación un gráfico resumen:



No es el objetivo de este módulo entrar en el funcionamiento exhaustivo de estos componentes, pero se considera necesario establecer una relación directa entre ellos y los conceptos estudiados de forma que ayude a comprender la relación entre ambos.

* Unidad de entrada/salida: realiza la transferencia de información con las unidades exteriores, lo que permite cargar datos y programas en la memoria principal y extraer los resultados bien sean impresos, por pantalla...
* Memoria principal: es una unidad dividida en celdas que se identifican mediante una dirección. Todas las celdas son del mismo tamaño y su función es guardar los datos y el programa que se está ejecutando en el computador.
* Microprocesador: se llama al conjunto formado por la unidad de control, los registros (elementos que contienen la información del programa que se está ejecutando) y la unidad aritmético-lógica de un computador.

A su vez, el microprocesador, también contiene componentes que se relacionan entre ellos:



La función del microprocesador es decodificar y ejecutar de forma sincronizada las instrucciones almacenadas en memoria. Los datos sobre los que opera esta unidad provienen de la memoria principal y pueden estar almacenados de forma temporal en alguno de los registros de la propia unidad.

Unidad aritmético-lógica (ALU): es la que permite realizar una serie de operaciones elementales tales como suma, resta, and, or, etc. llamada también unidad de ejecución, donde se ejecutan el conjunto de operaciones lógicas y aritméticas del ordenador.

Unidad de control (UC): gobierna el funcionamiento del ordenador. Recibe e interpreta las instrucciones, generando las señales de control necesarias y para ejecutar la operación especificada por cada instrucción. Se encarga de leer las instrucciones de máquina almacenadas en la memoria principal, y de generar las señales de control necesarias para que el computador las ejecute.

Como hemos indicado un poco más arriba, un ordenador es incapaz de hacer nada por sí mismo, necesita programas que le indiquen los pasos a realizar. Un programa es un conjunto de instrucciones que son almacenadas secuencialmente en posiciones o direcciones sucesivas de memoria y que serán ejecutadas una tras otra.

Por un lado, en la memoria tenemos una serie de instrucciones ubicadas secuencialmente y un conjunto de datos. Por otra tenemos una unidad que es capaz de ejecutar instrucciones y por último, otra unidad que es capaz de descodificar las instrucciones y controla la sincronización entre todas las unidades.

Es por eso que el funcionamiento de las computadoras consiste en ir extrayendo sucesivamente instrucciones de la memoria, interpretarlas, extraer de memoria los operandos implicados en la operación, enviarlos a la unidad que realiza las operaciones y hallar el resultado.

Sigue un ciclo de operaciones determinado, que funciona de la siguiente manera:

1. Cuando se alimenta/reinicia se prepara para hacer referencia a una posición inicial de memoria dada

2. Se lee el dato en la posición y se manda a la UC

3. La UC decodifica la instrucción y emite las señales a la ALU para que las ejecute.

4. Se incrementa el contador de programa a la siguiente posición indicada.

5. Se repite el ciclo desde el paso 2 hasta recibir una señal de parada o instrucción de fin de programa

Y hasta aquí hemos visto como interpreta cualquier sistema digital los programas.

Como vemos, la creación de un programa y, por lo tanto, el algoritmo para resolverlo está íntimamente relacionado con las instrucciones que va a ejecutar el computador. Por ello, es importante tener en mente ciertos parámetros de calidad a la hora de construirlos.

El algoritmo más eficiente es aquel que toma el mínimo tiempo de ejecución y uso de memoria posibles, y todavía produce una respuesta correcta.

Hay dos cosas que no se deben olvidar nunca a la hora de crear programas:

1. Todos los programas tienen errores y descubrirlos sólo es cuestión de tiempo y de que el programa tenga éxito y se utilice frecuentemente.

2. Todos los programas sufren modificaciones a lo largo de su vida, al menos todos aquellos que tienen éxito.

Ahora bien, hasta ahora, hemos visto como se expresa un algoritmo en lenguaje natural. El lenguaje natural es el lenguaje hablado o escrito por humanos para propósitos generales de comunicación. Son aquellas lenguas que han sido generadas espontáneamente en un grupo de hablantes con propósito de comunicarse. En contraposición tenemos otras lenguas, como pueden ser un lenguaje de programación o los lenguajes usados en el estudio de la lógica matemática.

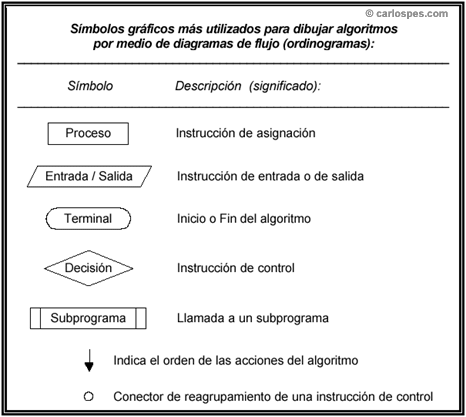
Podemos expresar un algoritmo de muchas maneras, incluyendo lenguaje natural, diagramas de flujo, pseudocódigo y, por supuesto, en lenguajes de programación reales. El lenguaje natural es popular, pues se nos da naturalmente y puede comunicar los pasos de un algoritmo a una audiencia general. Pero, es necesario poderlos expresar de una forma más formal lo cual a su vez, también hace que el algoritmo gane precisión, el lenguaje natural está lleno de ambigüedades y aspectos semánticos que pueden depender totalmente del contexto, por ello disponemos de:

Diagramas de Flujo: Se basan en la utilización de diversos símbolos para representar operaciones específicas. Se les llama diagramas de flujo porque los símbolos utilizados se conectan por medio de flechas para indicar la secuencia de operación.

El ordinograma es un estilo específico de diagrama de flujo, que muestra la secuencia lógica y detallada de las operaciones que necesitamos para la realización de un programa, paso a paso y con sus distintos caminos, tomas de decisiones, repeticiones, etc.

Pseudocódigo: Es una descripción de un algoritmo de programación informático de alto nivel compacto e informal que utiliza las convenciones estructurales de un lenguaje de programación verdadero, pero que está diseñado para la lectura humana en lugar de la lectura en máquina, y con independencia de cualquier otro lenguaje de programación.

Para trabajar con ordinogramas podemos utilizar cualquiera de los siguientes elementos:



**EJEMPLO**

PROBLEMA A RESOLVER: Sistema que indique si una nota numérica es aprobado o suspenso:

|  |  |
| --- | --- |
| Mediante ordinograma | Mediante pseudocódigo |