Práctica 5.

Algoritmos y Programas

Un algoritmo establece, de manera genérica e informal, la secuencia de pasos o acciones que resuelve un determinado problema. Los algoritmos constituyen la documentación principal que se necesita para poder iniciar la fase de codificación y, para representarlos, se utiliza, fundamentalmente, dos tipos de notación: pseudocódigo y diagramas de flujo. El diseño de un algoritmo es independiente del lenguaje que después se vaya a utilizar para codificarlo.

¿Qué cualidades tiene que tener un algoritmo?

Para cualquier problema dado no existe una única solución algorítmica; es tarea de la persona que diseña un algoritmo encontrar la *solución más óptima*, ésta no es otra que aquella que cumple más fielmente las cualidades deseables de todo algoritmo bien diseñado:

- **Finitud**. Un algoritmo siempre tiene que finalizar tras un número finito de acciones.
- **Precisión**. Todas las acciones de un algoritmo deben estar bien definidas, esto es, ninguna acción puede ser ambigua, sino que cada una de ellas sólo se debe poder interpretar de una única manera. Dicho de otra forma, si el programa que resulta de un algoritmo se ejecuta varias veces con los mismos datos de entrada, en todos los casos se obtendrán los mismos datos de salida.
- **Claridad**. Lo normal es que un problema se pueda resolver de distintas formas. Por tanto, una de las tareas más importantes del diseñador de un algoritmo es encontrar la solución más legible, es decir, aquella más comprensible para el ser humano.
- **Generalidad**. Un algoritmo debe resolver problemas generales. Por ejemplo, un programa que realice sumas de números enteros deberá servir para realizar sumas de dos números enteros cualesquiera, y no, solamente, para sumar dos números determinados, como pueden ser el 3 y el 5.
- **Eficiencia**. La ejecución del programa resultante de codificar un algoritmo deberá consumir lo menos posible los recursos disponibles del ordenador (memoria, tiempo de CPU, etc.).
- **Sencillez**. A veces, encontrar la solución algorítmica más eficiente a un problema puede llevar a escribir un algoritmo muy complejo, afectando a la claridad del mismo. Por tanto, hay que intentar que la solución sea sencilla, aun a costa de perder un poco de eficiencia, es decir, se tiene que buscar un equilibrio entre la claridad y la eficiencia. Escribir algoritmos sencillos, claros y eficientes se consigue a base de práctica.
- **Modularidad**. Nunca hay que olvidarse del hecho de que un algoritmo puede formar parte de la solución a un problema mayor. Pero, a su vez, dicho algoritmo debe descomponerse en otros, siempre y cuando, esto favorezca a la claridad del mismo.

La persona que diseña un algoritmo debe ser consciente de que todas las propiedades de un algoritmo se transmitirán al programa resultante.

1. Diseñar algoritmos e implementar una función R para calcular:
    a. El factorial de un número dado.
    b. El número de Fibonacci de un número dado.
    c. El algoritmo de Euclides (mcd)
2. Encontrar aplicaciones de los algoritmos anteriores, ¿qué funciones R resuelven dichos algoritmos?