ALGORITMOS/2

Algoritmo con n.º enteros ordenados: búsqueda binaria

Supón que tenemos 100 cajas que contienen 100 números ordenados de menor a mayor. Queremos saber si uno de esos números es uno determinado (el 37?, el 65?). ¿Como hacerlo en el mínimo n.º de pasos?. Con la búsqueda binaria (algoritmo que sirve para cualquier n.º de cajas) que se resume en:

1) abre la caja del medio (la n.º 50 en nuestro ejemplo, si el n.º de cajas es impar, redondea). Si es el n.º buscado, stop.
2) Si no es y el n.º buscado es mayor que el de la caja, quedate con las cajas de la derecha (en nuestro ejemplo, de la 51 al 100). Vuelve al paso 1.
3) Si es menor, quédate con las cajas de la izquierda (de la 1 a la 49) . Vuelve al paso 1.

Con la búsqueda binaria en vez de tener que abrir, en el peor de los casos, 100 cajas sólo tenemos que abrir 7 como mucho. Pero lo sorprendente es que si tenemos 1000 cajas sólo tendremos que abrir 10. En general con N cajas, el algoritmos de búsqueda binaria garantiza encontrar la solución en $\log_2 N$ número de pasos.
La búsqueda binaria la usa el ordenador para buscar cosas ordenadas, por ejemplo el corrector ortográfico cuando recomienda palabras está recorriendo el diccionario, que son varios miles de palabras. Y también las bases de datos para encontrar datos ordenados.
Para programarlo en un lenguaje de programación usamos las siguientes variables:

- n aleatorio: es el n.º buscado, en nuestro ejemplo vale entre 0 y 99
- x es el n.º que usamos para ir buscando, que siempre vale el medio de el límite inferior y el superior. Cuando x = n aleatorio, se acabó el algoritmo.
- L inferior y L superior son los intervalos de las cajas que abrimos (siempre abrimos la del medio). Al principio l inferior = 0 y L superior = 99 y luego irán cambiando.
- El algoritmo se basa en saber si x es mayor o menor que n aleatorio. Si es menor (por ejemplo, x = 50) ahora el L superior es 50 y x pasa a valer el medio de 0 y 50, 25. Si es mayor l inferior vale 50 y x 75.