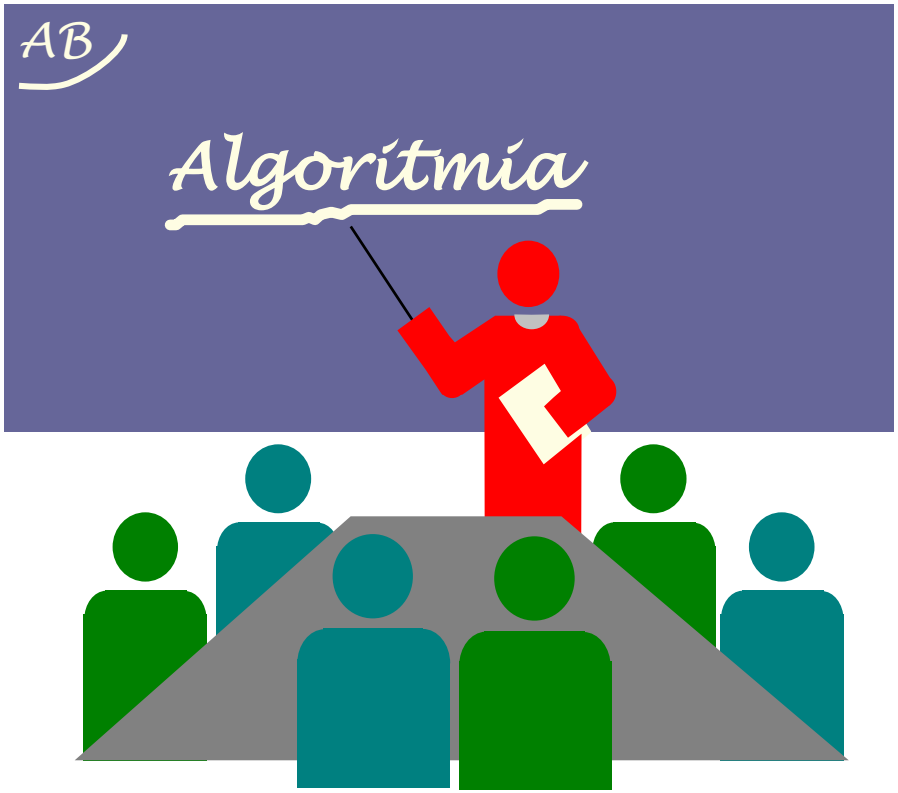
Introducción a la algoritmia

Introducción a la algoritmia



Algoritmia =

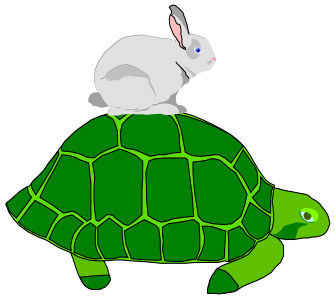
= tratamiento sistemático de técnicas fundamentales para el diseño y análisis de algoritmos eficientes

Introducción a la algoritmia

- Computadores cada vez más rápidos y a más bajo precio:
  - Se resuelven problemas de cálculo antes impensables.
  - Inconscientemente se tiende a restar importancia al concepto de **eficiencia**.

- Existen problemas que seguirán siendo intratables si se aplican ciertos algoritmos por mucho que se aceleren los computadores

  ⇒ importancia de nuevas soluciones eficientes



Introducción a la algoritmia

- Ejemplo:

  "En Agosto de 1977, *Scientific American* proponía a sus lectores el reto consistente en descifrar un mensaje secreto, para así ganar cien dólares. Parecía algo seguro: se estimaba en aquel momento que el ordenador más rápido existente, empleando el algoritmo más eficiente de los conocidos, no podría ganar la apuesta salvo funcionando sin interrupción durante un tiempo equivalente a millones de veces la edad del Universo. Sin embargo, ocho meses de cálculo que comenzaron dieciséis años después bastaron para la tarea. ¿Qué había pasado?…"

  G. Brassard y P. Bratley
  Fundamentos de Algoritmia (Prólogo)

Introducción a la algoritmia

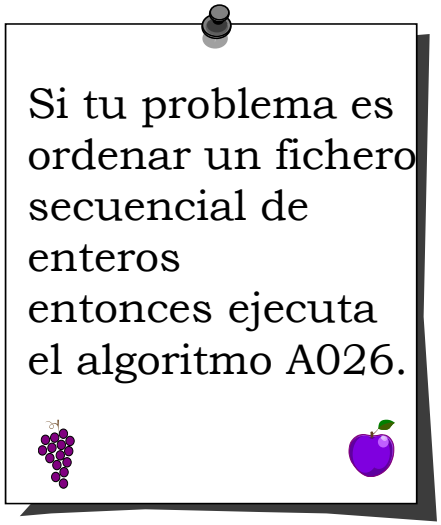
- Un curso de algoritmia

  NO ES

  – ni un curso de programación
  (ya deberíais saber programar)
  – ni un curso de estructuras de datos
  (ya deberíais conocer las fundamentales)

  TAMPOCO ES

  – una colección de “recetas”
  o algoritmos listos para
  ser introducidos en el
  computador para resolver
  problemas específicos



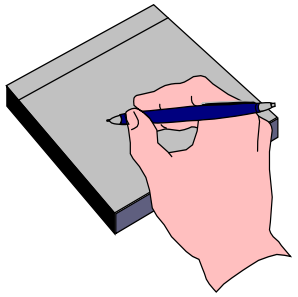
Introducción a la algoritmia

- Un curso de algoritmia tiene como objetivo principal:

  – dar más herramientas fundamentales para facilitar el desarrollo de programas

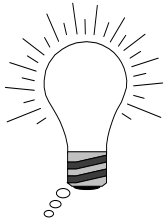
¿qué herramientas?:

técnicas o “esquemas” de diseño de algoritmos eficientes

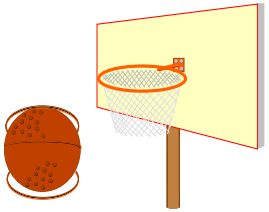


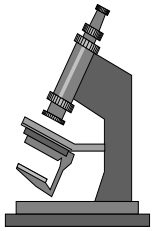
Introducción a la algoritmia

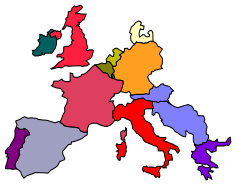
- Un medio para alcanzar ese objetivo es:
  - presentar cada esquema de forma genérica, incidiendo en los principios que conducen a él, e



  - ilustrar el esquema mediante ejemplos concretos de algoritmos tomados de varios dominios de aplicación







Introducción a la algoritmia

- Un problema muy sencillo:

  Multiplicación de dos enteros positivos con lápiz y papel.

  – En España:

```
  981
 1234
 3924
2943
1962
981
1210554
```

  – En Inglaterra:

```
  981
 1234
 981
1962
2943
3924
1210554
```

  – Ambos métodos son muy similares, los llamaremos algoritmo “clásico” de multiplicación.

Introducción a la algoritmia

– Algoritmo de multiplicación “a la rusa”:

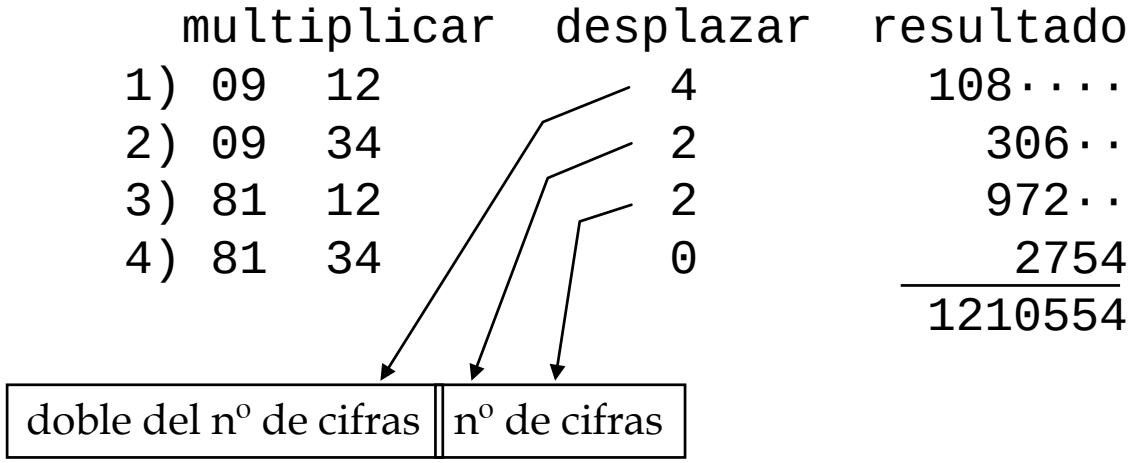
✓	981	1234	1234
	490	2468	
✓	245	4936	4936
	122	9872	
✓	61	19744	19744
	30	39488	
✓	15	78976	78976
✓	7	157952	157952
✓	3	315904	315904
✓	1	631808	631808
			1210554

- Ventaja: no hay que almacenar los productos parciales.
- Sólo hay que saber sumar y dividir por 2.

Introducción a la algoritmia

- Todavía otro algoritmo:
  - De momento, exigimos que ambos números tengan igual nº de cifras y que éste sea potencia de 2.

    Por ejemplo: `0981` y `1234`.
  - En primer lugar, partimos ambos números por la mitad y hacemos cuatro productos:



Es decir, hemos reducido un producto de n<u>os</u> de 4 cifras en cuatro productos de n<u>os</u> de 2 cifras, varios desplazamientos y una suma.

Introducción a la algoritmia

- Los productos de números de 2 cifras pueden hacerse con la misma técnica. Por ejemplo, `09` y `12`:

```
   multiplicar  desplazar  resultado
1)  0   1          2          0··
2)  0   2          1           0·
3)  9   1          1           9·
4)  9   2          0           18
                                108
```

– Es un ejemplo de algoritmo que utiliza la técnica de “divide y vencerás”.

– Tal y como lo hemos presentado…

NO mejora en eficiencia al algoritmo clásico.

– Pero, puede mejorarse:

Es posible reducir un producto de dos números de muchas cifras a 3 (en vez de 4) productos de números de la mitad de cifras, y éste SÍ que mejora al algoritmo clásico.

– Y aún se conocen métodos más rápidos para multiplicar números muy grandes.

Introducción a la algoritmia

- Ideas clave:
  - Incluso para un problema tan básico pueden construirse **MUCHAS** soluciones.
  - El método clásico lo usamos con lápiz y papel porque nos resulta muy **familiar** (lo que se aprende en la infancia…).
  - El método "a la rusa" se implementa en hardware en los computadores por la naturaleza **elemental** de los cálculos intermedios.
  - El método de divide y vencerás es más **rápido** si se quiere multiplicar números grandes.

    La **algoritmia** estudia las propiedades de los algoritmos y nos ayuda a **elegir** la solución más adecuada en cada situación.

    En muchos casos, una buena elección **ahorra tiempo** y **dinero**.

    En algunos casos, una buena elección marca la diferencia entre **poder** resolver un problema y **no poder** hacerlo.

Introducción a la algoritmia

- ¿A quién puede interesar este curso?

  A todo aquél a quien:

  – le guste diseñar algoritmos para resolver nuevos problemas o algoritmos mejores a los triviales para resolver viejos problemas, y

  – tenga curiosidad por conocer, por ejemplo, cómo se resuelven los siguientes problemas:

Introducción a la algoritmia

- ¿Cómo multiplicar dos números enteros de forma más eficiente que con el algoritmo clásico?
- ¿Cómo se compactan ficheros mediante el algoritmo de Huffman?
- ¿Cómo se consiguen implementaciones muy eficientes de la multiplicación y potenciación de enteros muy grandes para el algoritmo RSA?
- ¿Por qué funcionan algoritmos vistos en Matemática Discreta como el de Dijkstra, Prim o Kruskal?

Introducción a la algoritmia

- ¿Existen árboles binarios de búsqueda óptimos suponiendo que se conocen las probabilidades de búsqueda de cada clave?

- ¿Alguna forma de resolver el problema del viajante de comercio?

- ¿Cómo se compara el ADN de un *Homo sapiens* y de un *Homo neanderthalensis*?

Algoritmia básica: Contenidos de la asignatura

- Introducción a la algoritmia
- Algoritmos voraces
- Divide y vencerás
- Programación dinámica
- Búsqueda con retroceso
- Ramificación y poda
- Programación lineal y reducciones

Algoritmia básica: Bibliografía básica

[CLRS09] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein. *Introduction to Algorithms (3rd edition)*, The MIT Press, 2009.

[DPV08] S. Dasgupta, C. Papadimitriou, U. Vazirani. *Algorithms*, McGraw-Hill, 2008.

[BB97] G. Brassard, P. Bratley. *Fundamentos de Algoritmia*, Prentice Hall, 1997.

Algoritmia básica: Bibliografía complementaria

[HSR08] E. Horowitz, S. Sahni, and S. Rajasekaran. *Computer algorithms*, 2nd ed., Silicon Press, 2008.

[KT05] J. Kleinberg, E. Tardos. *Algorithm Design*, Addison-Wesley, 2005.

[PG02] I. Parberry and W. Gasarch. *Problems on Algorithms (2nd edition)*, free book, 2002.

Además: transparencias y otro material adicional disponibles en la web de la asignatura, hojas de ejercicios, enlaces de Internet (ejemplo: http://jeffe.cs.illinois.edu/teaching/algorithms/), etc.

Chuleta

- Coste de algoritmos recursivos:

  El coste $T(n)$ de una función para datos de tamaño n se define en función del coste $T(m)$ de las llamadas recursivas para otros tamaños m (menores que n)

Teorema (*Master Theorem*):

$$T_1(n) = \begin{cases} f(n) & \text{si } 0 \le n < c \\ a \cdot T_1(n-c) + b \cdot n^k & \text{si } c \le n \end{cases} \Rightarrow T_1(n) \in \begin{cases} \Theta(n^k) & \text{si } a < 1 \\ \Theta(n^{k+1}) & \text{si } a = 1 \\ \Theta(a^{n/c}) & \text{si } a > 1 \end{cases}$$

$$T_2(n) = \begin{cases} g(n) & \text{si } 0 \le n < c \\ a \cdot T_2(n/c) + b \cdot n^k & \text{si } c \le n \end{cases} \Rightarrow T_2(n) \in \begin{cases} \Theta(n^k) & \text{si } a < c^k \\ \Theta(n^k \cdot \log n) & \text{si } a = c^k \\ \Theta(n^{\log_c a}) & \text{si } a > c^k \end{cases}$$