

**INSTITUTO POLITÉCNICO NACIONAL**

**ESCUELA SUPERIOR DE CÓMPUTO**

**ANÁLISIS DE ALGORITMOS**

**PROFESORA: LUZ MARÍA SÁNCHEZ GARCÍA**

**INTEGRANTES:**

**VÁZQUEZ MORENO MARCOS OSWALDO 2016601777**

**DE LOS SANTOS DÍAZ LUIS ALEJANDRO 2017630451**

**PRÁCTICA 6 ANÁLISIS DE ALGORITMOS DIVIDE Y VENCERÁS MULTIPLICACIÓN ENTERA**

**3CM2**

**9 DE ABRIL DE 2019**

**Introducción**

Multiplicación de enteros grandes

• Coste de realizar las operaciones elementales de suma y multiplicación:

– Es razonable considerarlo constante si los operandos son directamente manipulables por el hardware, es decir, no son muy grandes.

– Si se necesitan enteros muy grandes, hay que implementar por software las operaciones.

• Enteros muy grandes:

– Representación de un entero de n cifras: puede hacerse en un vector con un espacio en O(n) bits.

– Operaciones de suma, resta: pueden hacerse en tiempo lineal.

– Operaciones de multiplicación y división entera por potencias positivas de 10: pueden hacerse en tiempo lineal (desplazamientos de las cifras).

– Operación de multiplicación con el algoritmo clásico (o con el de multiplicación rusa): tiempo cuadrático. (Campos, 2019)

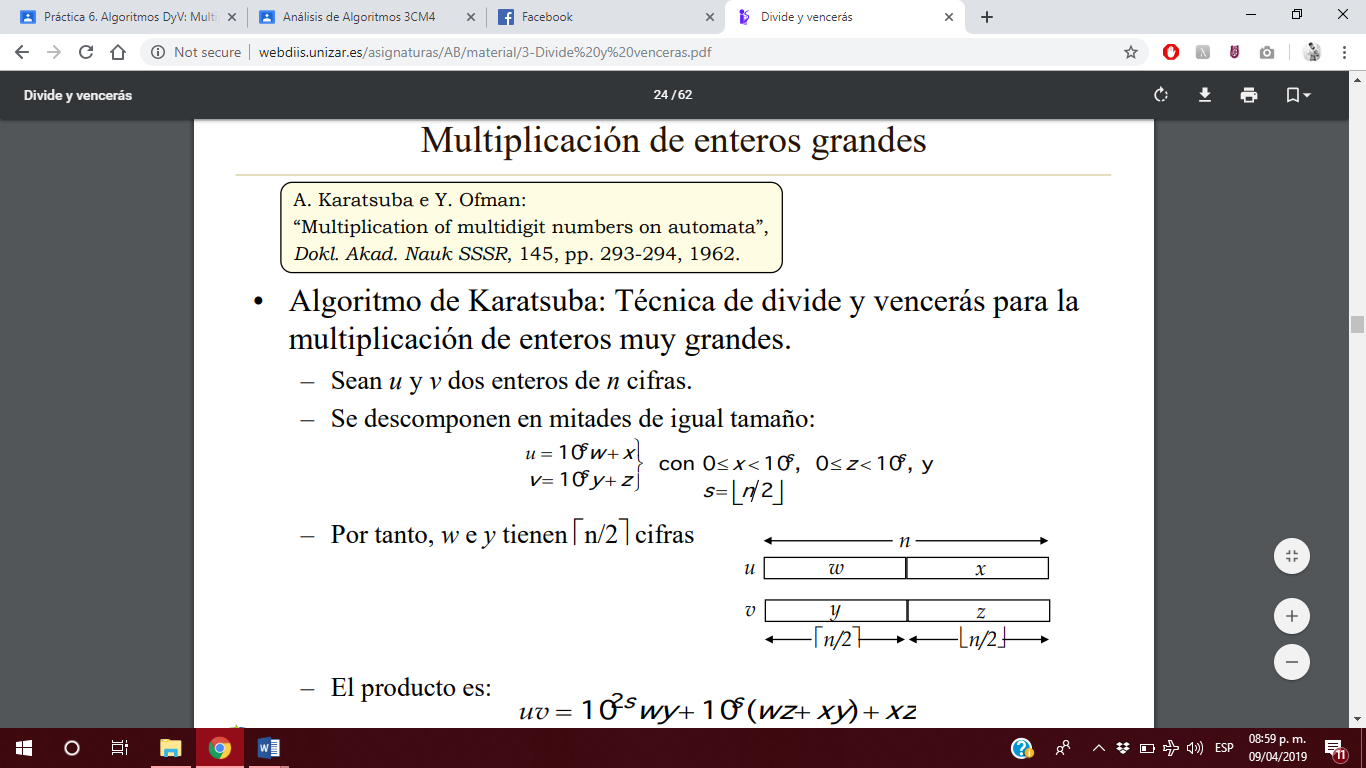
**Planteamiento del problema**

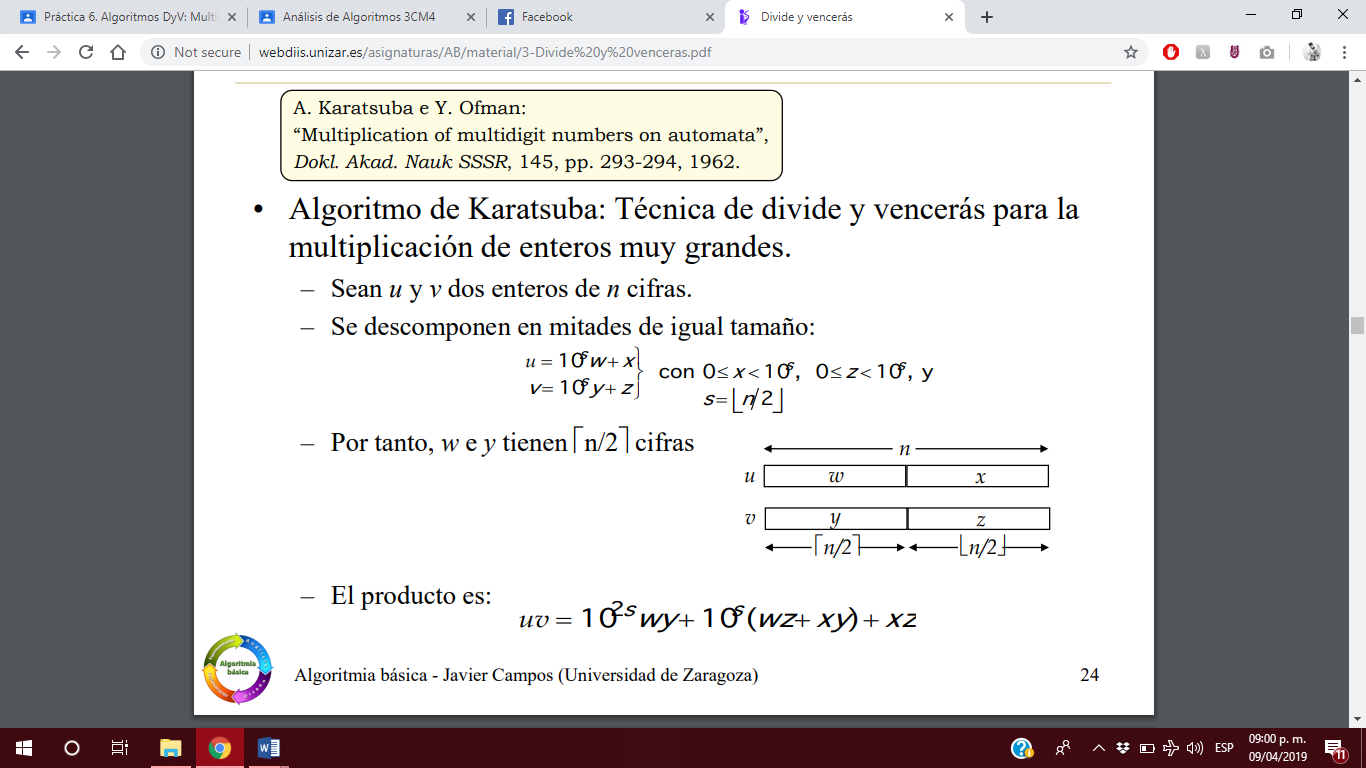
Sean *u* y *v dos números naturales n* bits donde, por simplicidad, *n*  es una potencia de 2. El algoritmo tradicional para multiplicarlos es de complejidad **O(n2).**

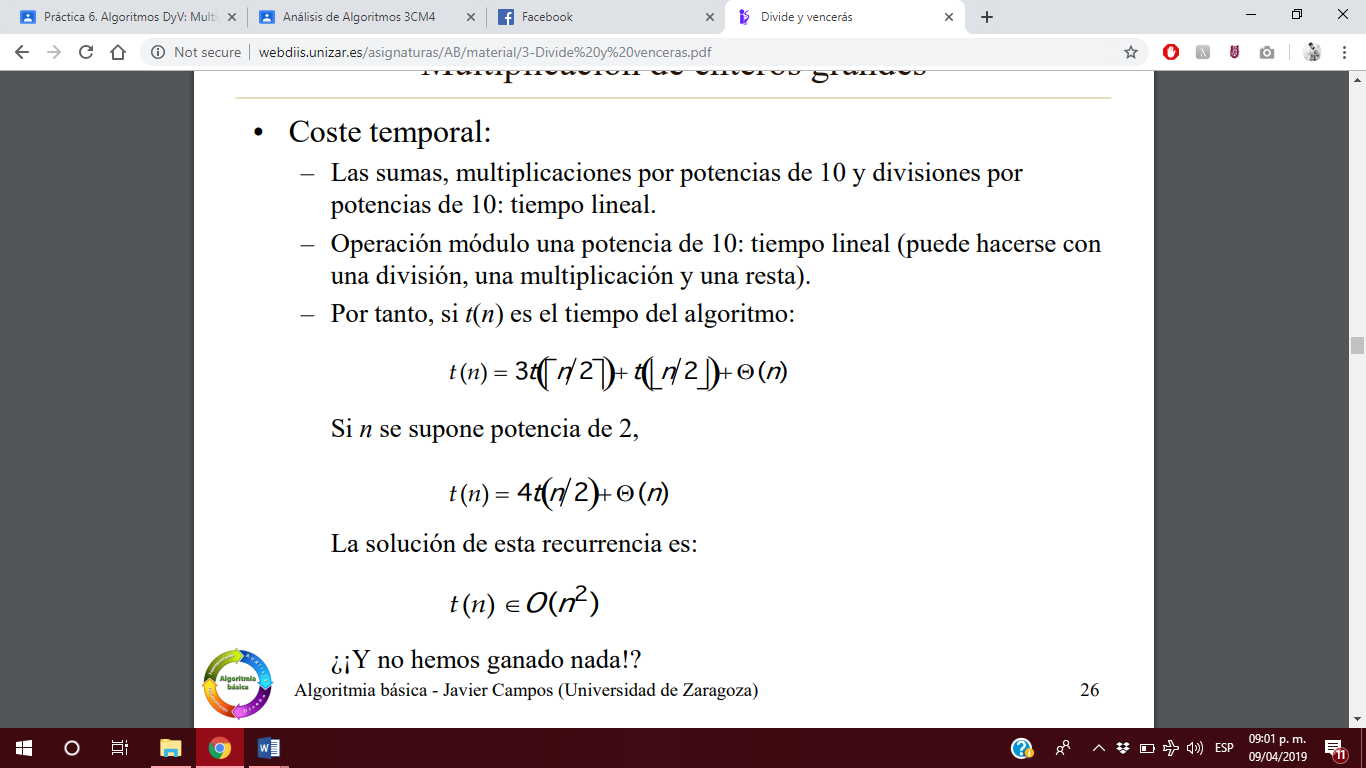
Implementar el algoritmo para lograr reducir una multiplicación de 4 cifras a 4 multiplicaciones de 2 cifras, más tres sumas y varios desplazamientos.

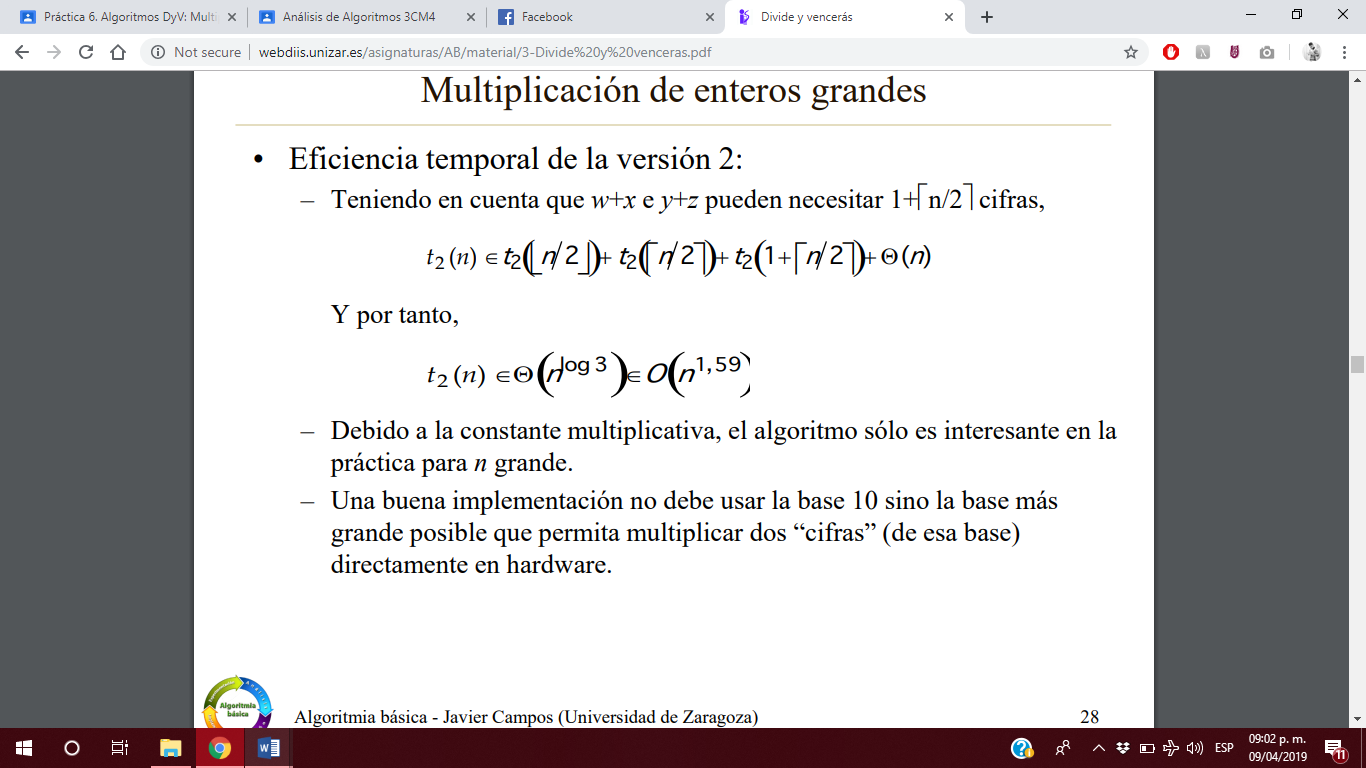
**Diseño de la solución**

Según el algoritmo de Karatsuba con la técnica de DyV para este problema dice que se descomponen en mitades de igual tamaño:









A continuación, se muestran los diagramas de flujo de nuestra propuesta de solución para los algoritmos de ordenamiento.

Primeramente, se muestra en el diagrama 1.1 el main.c de nuestra propuesta de solución.

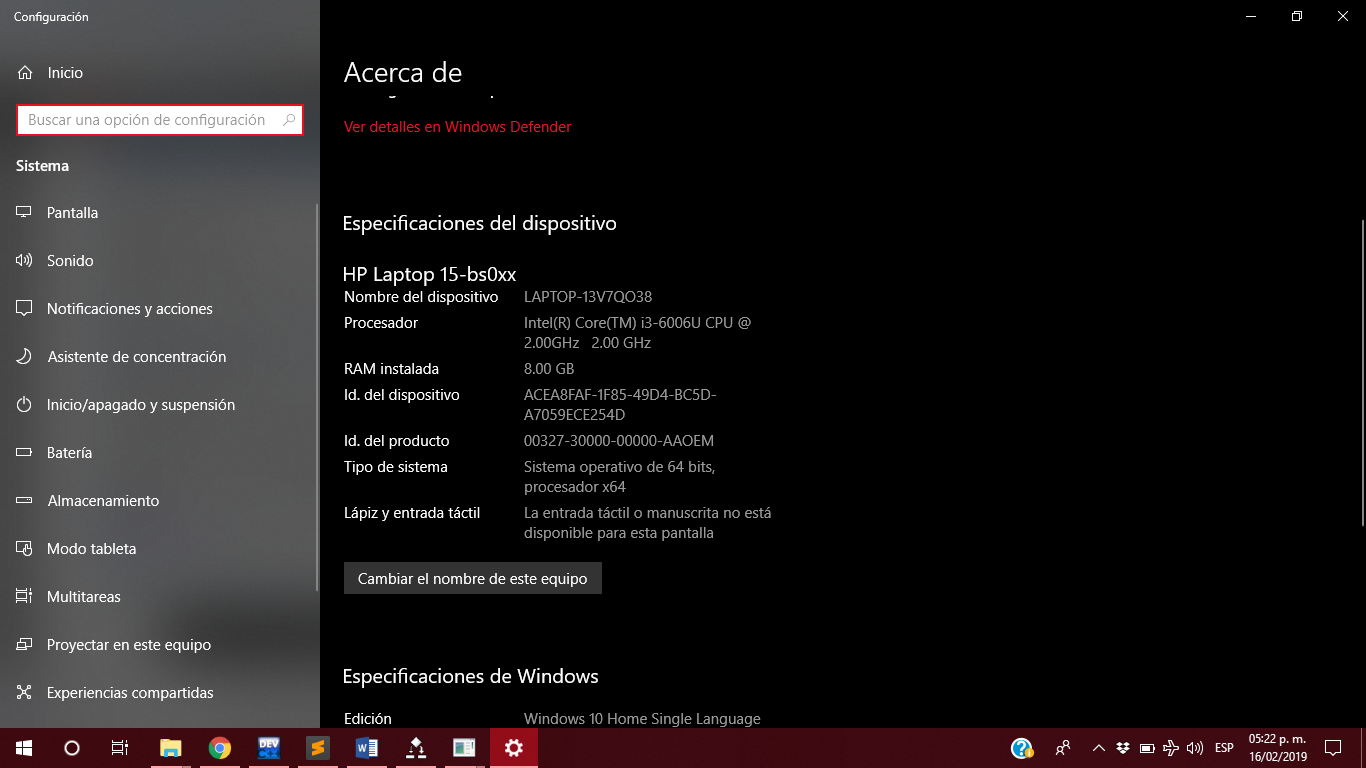
Diagrama 1.1 Main.c

**Implementación de la solución**

**Funcionamiento**

**Plataforma experimental**

La ejecución de los algoritmos anteriores se llevó a cabo en una computadora personal que se describe en la siguiente imagen.

  
Imagen 3.1 Plataforma Experimental

El compilador utilizado fue gcc integrado dentro del IDE DevC en un sistema operativo de 64 bits Windows 10.

**Gráfica del comportamiento temporal.**

|  |  |  |
| --- | --- | --- |
| Tamaño de M | Tamaño de N | Tiempo |
| 100 | 100 | 0.000077 |
| 80 | 80 | 0.000076 |
| 60 | 60 | 0.0000732 |
| 40 | 40 | 0.000025 |
| 20 | 20 | 0.000006 |
| 10 | 10 | 0.000003 |
| 8 | 8 | 0.000001 |
| 4 | 4 | 0.000001 |

Tabla 1.1

Gráfica 1.1

**Cuestionario**

**1.- ¿Cuál de los 2 algoritmos es más útil en la práctica?**

Se ha decidido en conjunto que el caso de la multiplicación de números grandes es más útil si tomamos en cuenta que no logramos programar la transformada ráida de Fourier por indicación de la profesora debido al corto tiempo y las vacaciones de semana santa.

**2.-¿En qué aplicaciones se pueden implementar?**

La transformada rápida de Fourier FFT es un algoritmo que reduce el tiempo de cálculo de n2 pasos a n·log2(n). El único requisito es que el número de puntos en la serie tiene que ser una potencia de 2 (2n puntos), por ejemplo 32, 1024, 4096, etc.

**3.- ¿Considera que el algoritmo de Fourier es eficiente? ¿Por qué?**

Si porque debido al tipo de complejidad en el algoritmo provoca un cambio brusco siendo la Serie Rápida de Fourier una función exponencial.

**4.- Describa brevemente otros 3 algoritmos que se implementen con la técnica DyV.**

1. *Máxima suma en cualquier subarreglo contiguo:*

Nos dan un arreglo A[0,1,2,3….n-1] de enteros, y queremos encontrar la máxima suma en cualquier subarreglo contiguo, mismo problema quew se puede solucionar con la estrategia greedy para hallar la máxima subsuma.

1. *Amigos y regalos*:

Tenemos dos números naturales C1 y C2 y dos primos P1 y P2 tal que p1<P2. Queremos hallar el mínimo valor de n perteneciente a los naturales tal que podamos particionar el conjunto 1{1, 2, 3… n} en dos conjuntos A y B que cumplan:

* |A| >= C1
* |B| >= C2
* Ningún elemento de A es divisible entre P1.
* Ningún elemento de B es divisible entre P2.

1. *Cúmulo:*

Dados n puntos en el plano cartesiano XY, hay que hallar la mínima distancia entre cualquier par de puntos.

**5.- ¿Qué tipo de problemas tuvo al realizar esta práctica?**

Se tuvieron complicaciones en cuestión de tiempos, ya que el algoritmo es demasiado rápido en números sumamente pequeños, sin embargo con números ya más grandes comienza a notarse la diferencia, al principio creímos que lo estábamos implementando mal ya que no notábamos diferencias de ningún tipo.

**Conclusiones**

**Bibliografía**

# Bibliografía

Academy, K. (15 de 6 de 2006). *Aloritmo Divide yVencerás .* Obtenido de https://es.khanacademy.org/computing/computer-science/algorithms/merge-sort/a/divide-and-conquer-algorithms

Campos, J. (07 de Abril de 2019). *Algoritmia básica.* Obtenido de Algoritmia básica (Universidad de Zaragoza): http://webdiis.unizar.es/asignaturas/AB/material/3-Divide%20y%20venceras.pdf

García, L. M. (15 de 03 de 2019). *Classrom.* Obtenido de Práctica 6 Ánalisis de Algoritmos: https://classroom.google.com/c/NzEzNTUxMDU1NVpa/a/NzgxNTE2NTA3MVpa/details