Introduction

COMP215: Design & Analysis of Algorithms

Today

- Karatsuba Multiplication
- Work on HW A
- Start on DQ #1

Karatsuba Multiplication

- We will start by **Recursive Algorithm** for integer multiplication then we will move to **Karatsuba Multiplication**.

The algorithm design space is surprisingly rich

Integer Multiplication

- Lets go back to our example:
- x= **2698,** y= **4263**

x * y= 26 98 * 42 63

a = 26, b = 98, c = 42, d = 63

a **b** **c** **d**

Integer Multiplication

x * y= 26 98 * 42 63

a b c d

- To calculate x*y:
  1. Compute a * c = 26 * 42= 1,092
  2. Compute b * d = 98 * 63= 6,174
  3. Compute (a + b) * (c + d) = (26+98) * (42+63)

     = 13,020

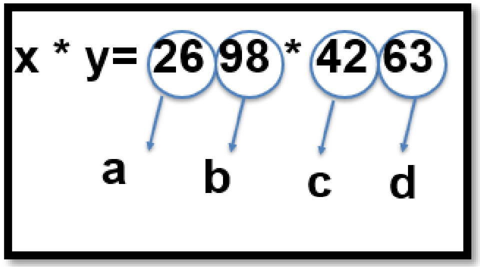
  4. Subtract the results of the first two steps from the result of the third step: 13,020 - 1,092 - 6,174

     = 5,754

  5. Compute 10^4 * 1,092 + 10^2 * 5,754 + 6,174=

     11,501,574

Integer Multiplication- A Recursive Algorithm



- In general, a number x with an **even number** *n* of digits can be expressed in terms of two **n/2-digit** numbers, its first half a and second half b:

$$x = 10^{n/2} * a + b.$$

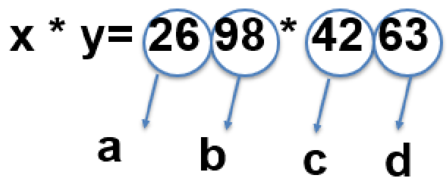
- Similarly, we can write

$$y = 10^{n/2} *c + d.$$

- To compute the product of x and y, let's use the two expressions above and multiply out:

$$x * y = (10^{n/2} *a + b) * (10^{n/2} *c + d) = 10^{n} *(a * c) + 10^{n/2} *(a * d + b * c) + b * d$$

Integer Multiplication



- $x * y = 10^n * (a * c) + 10^{n/2} * (a * d + b * c) + b * d$

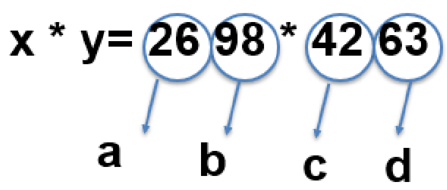
RecIntMult

Input: two n-digit positive integers x and y.
Output: the product $x \cdot y$.
Assumption: n is a power of 2.

---

if $n = 1$ **then** `// base case`
  compute $x \cdot y$ in one step and return the result
else `// recursive case`
  $a, b :=$ first and second halves of x
  $c, d :=$ first and second halves of y
  recursively compute $ac := a \cdot c$, $ad := a \cdot d$,
   $bc := b \cdot c$, and $bd := b \cdot d$
  compute $10^n \cdot ac + 10^{n/2} \cdot (ad + bc) + bd$ using
   grade-school addition and return the result

Karatsuba Multiplication



- Karatsuba multiplication is an optimized version of the RecIntMult algorithm.
- We again start from the expansion of x · y in terms of a, b, c, and d:

$$x * y = 10^{n} * (a * c) + 10^{n/2} * (a * d + b * c) + b * d$$

Karatsuba Multiplication

x * y= 26 98 * 42 63

a b c d

$$x * y = 10^{n} * (a * c) + 10^{n/2} * (a * d + b * c) + b * d$$

Compute x*y using Karatsuba Multiplication

Step 1: Recursively compute **a * c**.

Step 2: Recursively compute **b * d**.

=a*c + a*d + b*c + b*d

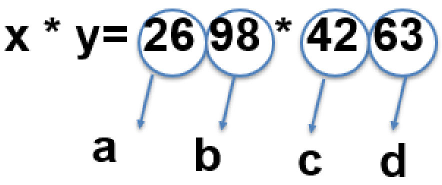
Step 3: Instead of recursively computing **a*d or b*c**, we recursively compute the product of a + b and c + d. Compute **a + b** and **c + d** using grade-school addition and recursively compute **(a + b) * (c + d)**

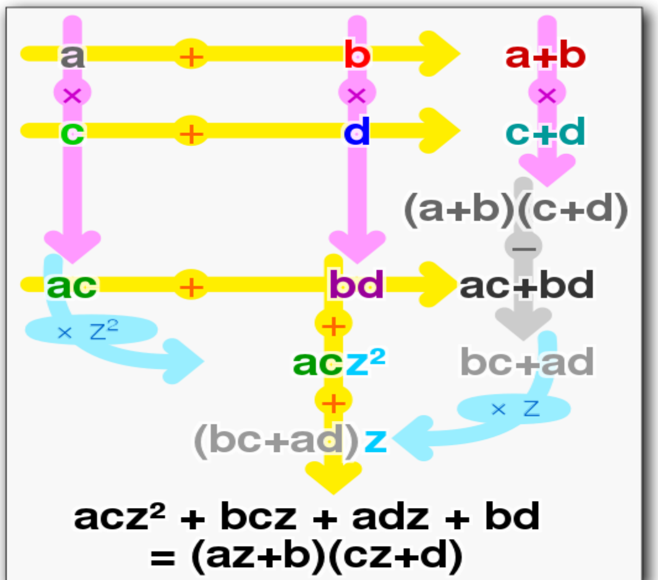
(a + b) * (c + d) – a*c - b*d = a*d + b*c

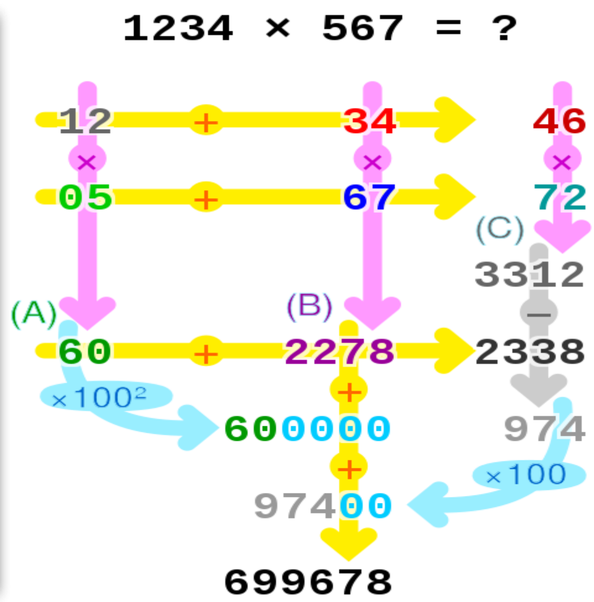
Step 4: Subtract the results of the first two steps from the result of the third step to obtain **a * d + b * c**.

Step 5: Compute (x*y) by adding up the results of steps 1, 2, and 4, after adding n trailing zeroes to the answer in step 1 and n/2 trailing zeroes to the answer in step 4.

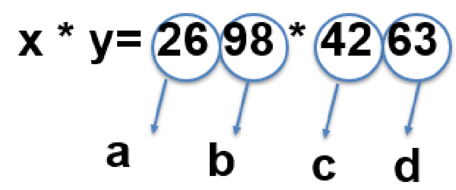
Karatsuba Multiplication







Karatsuba Multiplication



Karatsuba

Input: two n-digit positive integers x and y.
Output: the product $x \cdot y$.
Assumption: n is a power of 2.

---

if $n = 1$ **then** `// base case`
  compute $x \cdot y$ in one step and return the result
else `// recursive case`
  $a, b :=$ first and second halves of x
  $c, d :=$ first and second halves of y
  compute $p := a + b$ and $q := c + d$ using grade-school addition
  recursively compute $ac := a \cdot c$, $bd := b \cdot d$, and $pq := p \cdot q$
  compute $adbc := pq - ac - bd$ using grade-school addition
  compute $10^n \cdot ac + 10^{n/2} \cdot adbc + bd$ using grade-school addition and return the result