

CS11740 - Fall 2026

Algorithms

Integer Multiplication

Ibrahim Albluwi

Overview

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Overview

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Rules of the game.

- Adding or subtracting two n -digit numbers requires $\Theta(n)$ operations.

Overview

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Rules of the game.

- Adding or subtracting two n -digit numbers requires $\Theta(n)$ operations.
- Single-digit multiplication requires constant time.
(can be precomputed)

Overview

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Rules of the game.

- Adding or subtracting two n -digit numbers requires $\Theta(n)$ operations.
- Single-digit multiplication requires constant time.
(can be precomputed)
- Multiplying a decimal number by 10 requires constant time.
(shift-left)

Overview

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Rules of the game.

- Adding or subtracting two n -digit numbers requires $\Theta(n)$ operations.
- Single-digit multiplication requires constant time.
(can be precomputed)
- Multiplying a decimal number by 10 requires constant time.
(shift-left)

Goal. See an example of a divide and conquer algorithm that is not related to searching or sorting.

Attempt # 1: Repeated Addition

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)

Quiz 1

Problem. Given two positive integers x and y of length n decimal digits each, how many single digit operations are performed if $x \times y$ is computed using repeated addition?

- A.** $\Theta(n)$
- B.** $\Theta(n \log n)$
- C.** $\Theta(n^2)$
- D.** $\Theta(2^n)$
- E.** None of the above

Quiz 1

Problem. Given two positive integers x and y of length n decimal digits each, how many single digit operations are performed if $x \times y$ is computed using repeated addition?

A. $\Theta(n)$

B. $\Theta(n \log n)$

C. $\Theta(n^2)$

D. $\Theta(2^n)$

E. None of the above

Quiz 1

Problem. Given two positive integers x and y of length n decimal digits each, how many single digit operations are performed if $x \times y$ is computed using repeated addition?

A. $\Theta(n)$

B. $\Theta(n \log n)$

C. $\Theta(n^2)$

D. $\Theta(2^n)$

E. None of the above

- The largest decimal number representable using n digits is $10^n - 1$
- Therefore, y is added $\sim 10^n$ times in the worst case.
- Each addition involves $\Omega(n)$ single digit operations.
- **Total** = $\Omega(n10^n)$ single digit operations.

Attempt # 1: Repeated Addition

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)



$\sim 10^n$ additions each involving $\Omega(n)$ digits = $\Omega(n10^n)$ single digit operations.

The largest decimal number representable using n digits is $10^n - 1$

Attempt # 2: Long Multiplication

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)



$\sim 10^n$ additions each involving $\Omega(n)$ digits = $\Omega(n10^n)$ single digit operations.

The largest decimal number representable using n digits is $10^n - 1$

Solution 2. Long Multiplication.

Example: $2311 \times 4301 =$

$$\begin{array}{r} 2311 \\ \times 4301 \\ \hline 2311 \end{array}$$

Diagram illustrating the long multiplication process. Red arrows point from the digits of the multiplier (4, 3, 0, 1) to the corresponding digits of the multiplicand (2, 3, 1, 1), showing the alignment of partial products.

Attempt # 2: Long Multiplication

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)



$\sim 10^n$ additions each involving $\Omega(n)$ digits = $\Omega(n10^n)$ single digit operations.

The largest decimal number representable using n digits is $10^n - 1$

Solution 2. Long Multiplication.

Example: $2311 \times 4301 =$

$$\begin{array}{r} 2311 \\ \times 4301 \\ \hline 2311 \\ 0000 \\ 0000 \\ 0000 \end{array}$$

Diagram illustrating the long multiplication process. Red arrows point from the digits 2, 3, 1, and 1 of the first number to the corresponding digits 4, 3, 0, and 1 of the second number, showing the alignment of partial products.

Attempt # 2: Long Multiplication

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)



$\sim 10^n$ additions each involving $\Omega(n)$ digits = $\Omega(n10^n)$ single digit operations.

The largest decimal number representable using n digits is $10^n - 1$

Solution 2. Long Multiplication.

Example: $2311 \times 4301 =$

$$\begin{array}{r} 2311 \\ \times 4301 \\ \hline 2311 \\ 0000 \\ 6933 \end{array}$$

Diagram illustrating the long multiplication process. Red arrows point from the digits 2, 3, 1, and 1 of the first number to the digit 3 of the second number, indicating the multiplication steps.

Attempt # 2: Long Multiplication

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)



$\sim 10^n$ additions each involving $\Omega(n)$ digits = $\Omega(n10^n)$ single digit operations.

The largest decimal number representable using n digits is $10^n - 1$

Solution 2. Long Multiplication.

Example: $2311 \times 4301 =$

$$\begin{array}{r} \times 2311 \\ \hline 2311 \\ 0000 \\ 6933 \\ + 9244 \\ \hline 9939611 \end{array}$$

Diagram illustrating the long multiplication process for 2311×4301 . Red arrows point from the digit 4 in the multiplier to the digits 2, 3, 1, and 1 in the multiplicand, indicating the partial products.

Attempt # 2: Long Multiplication

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)



$\sim 10^n$ additions each involving $\Omega(n)$ digits = $\Omega(n10^n)$ single digit operations.

The largest decimal number representable using n digits is $10^n - 1$

Solution 2. Long Multiplication.

Example: $2311 \times 4301 =$



$\Theta(n^2)$ single digit operations.

					2	3	1	1	
					↑	↑	↑	↑	
x					4	3	0	1	
					2	3	1	1	
				0	0	0	0		
			6	9	3	3			
		+	9	2	4	4			
			9	9	3	9	6	1	1

Attempt # 2: Long Multiplication

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Solution 1. Repeated addition.

Example: $200 \times 300 = 300 + 300 + \dots$ (200 times)



$\sim 10^n$ additions each involving $\Omega(n)$ digits = $\Omega(n10^n)$ single digit operations.

The largest decimal number representable using n digits is $10^n - 1$

Solution 2. Long Multiplication.

Example: $2311 \times 4301 =$



$\Theta(n^2)$ single digit operations.

🤔 Can we do better?

		2	3	1	1	
		↑	↑	↑	↑	
x		4	3	0	1	
		2	3	1	1	
		0	0	0	0	
		6	9	3	3	
+	9	2	4	4		
	9	9	3	9	6	1 1

Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

$$\begin{array}{ccc|ccc} \mathbf{4} & \mathbf{5} & \mathbf{3} & \mathbf{9} & \mathbf{6} & \mathbf{1} \\ \hline & a & & b & & \end{array}$$

Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

$$\begin{array}{ccc|ccc} \mathbf{4} & \mathbf{5} & \mathbf{3} & \mathbf{9} & \mathbf{6} & \mathbf{1} \\ \hline & a & & b & & \end{array}$$

Left half: Shift right $\frac{n}{2}$ steps: 0 0 0 **4 5 3**

Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

	<u>4</u>	<u>5</u>	<u>3</u>	<u>9</u>	<u>6</u>	<u>1</u>
	a			b		
Left half: Shift right $\frac{n}{2}$ steps:	0	0	0	4	5	3
Right half: Shift left $\frac{n}{2}$ steps:	9	6	1	0	0	0
then Shift right $\frac{n}{2}$ steps	0	0	0	9	6	1

Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

$$\begin{array}{ccc|ccc} \mathbf{4} & \mathbf{5} & \mathbf{3} & \mathbf{9} & \mathbf{6} & \mathbf{1} \\ \hline & a & & b & & \end{array}$$

Left half: Shift right $\frac{n}{2}$ steps: 0 0 0 **4 5 3**

Right half: Shift left $\frac{n}{2}$ steps: **9 6 1** 0 0 0
then Shift right $\frac{n}{2}$ steps: 0 0 0 **9 6 1**

The number x can be written as $a \cdot 10^{\frac{n}{2}} + b$

Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

4

5

3

9

6

1

0

0

0

4

5

3

9

6

1

0

0

0

0

0

0

9

6

1

a

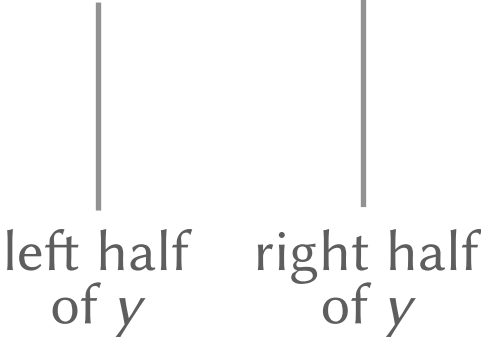
b

Left half: Shift right $\frac{n}{2}$ steps:

Right half: Shift left $\frac{n}{2}$ steps:
then Shift right $\frac{n}{2}$ steps

The number x can be written as $a \cdot 10^{\frac{n}{2}} + b$

Observation 2. $x \times y$ can be written as: $(a \cdot 10^{\frac{n}{2}} + b) \times (c \cdot 10^{\frac{n}{2}} + d)$



Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

$$\begin{array}{ccc|ccc} \mathbf{4} & \mathbf{5} & \mathbf{3} & \mathbf{9} & \mathbf{6} & \mathbf{1} \\ \hline & & & & & \\ & a & & b & & \end{array}$$

Left half: Shift right $\frac{n}{2}$ steps: 0 0 0 **4 5 3**

Right half: Shift left $\frac{n}{2}$ steps: **9 6 1** 0 0 0
then Shift right $\frac{n}{2}$ steps 0 0 0 **9 6 1**

The number x can be written as $a \cdot 10^{\frac{n}{2}} + b$

Observation 2. $x \times y$ can be written as: $(a \cdot 10^{\frac{n}{2}} + b) \times (c \cdot 10^{\frac{n}{2}} + d)$
 $= ac \cdot 10^n + ad \cdot 10^{\frac{n}{2}} + bc \cdot 10^{\frac{n}{2}} + bd$

Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

$$\begin{array}{ccc|ccc} \mathbf{4} & \mathbf{5} & \mathbf{3} & \mathbf{9} & \mathbf{6} & \mathbf{1} \\ \hline & a & & b & & \end{array}$$

Left half: Shift right $\frac{n}{2}$ steps: 0 0 0 **4 5 3**

Right half: Shift left $\frac{n}{2}$ steps: **9 6 1** 0 0 0
then Shift right $\frac{n}{2}$ steps 0 0 0 **9 6 1**

The number x can be written as $a \cdot 10^{\frac{n}{2}} + b$

Observation 2. $x \times y$ can be written as:

$$\begin{aligned} & (a \cdot 10^{\frac{n}{2}} + b) \times (c \cdot 10^{\frac{n}{2}} + d) \\ &= ac \cdot 10^n + ad \cdot 10^{\frac{n}{2}} + bc \cdot 10^{\frac{n}{2}} + bd \\ &= ac \cdot 10^n + (ad + bc) \cdot 10^{\frac{n}{2}} + bd \end{aligned}$$

Attempt # 4: A Divide & Conquer Algorithm

Problem. Given two positive integers x and y of length n digits each, find $x \times y$.

Observation 1. A positive integer x of length n digits can be split into 2 halves as follows:

$$\begin{array}{ccc|ccc} \mathbf{4} & \mathbf{5} & \mathbf{3} & \mathbf{9} & \mathbf{6} & \mathbf{1} \\ \hline & a & & b & & \end{array}$$

Left half: Shift right $\frac{n}{2}$ steps: 0 0 0 **4 5 3**

Right half: Shift left $\frac{n}{2}$ steps: **9 6 1** 0 0 0
then Shift right $\frac{n}{2}$ steps: 0 0 0 **9 6 1**

The number x can be written as $a \cdot 10^{\frac{n}{2}} + b$

Observation 2. $x \times y$ can be written as: $(a \cdot 10^{\frac{n}{2}} + b) \times (c \cdot 10^{\frac{n}{2}} + d)$
 $= ac \cdot 10^n + ad \cdot 10^{\frac{n}{2}} + bc \cdot 10^{\frac{n}{2}} + bd$



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn$

to shift and perform 3
addition operations

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$ using the master method

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$ using the master method



Can we reduce the number of subproblems?

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$ using the master method



Can we reduce the number of subproblems?

karatsuba

ENTER

Karatsuba

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$



Can we reduce the number of subproblems?

karatsuba

Observation. If we know what $(a + b)(c + d)$ we know what $(ad + bc)$ is!

Explanation. $(a + b)(c + d) = \underline{ac} + \underline{ad + bc} + \underline{bd}$

we want
this sum

we know
these

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$



Can we reduce the number of subproblems?

karatsuba

Observation. If we know what $(a + b)(c + d)$ we know what $(ad + bc)$ is!

Explanation. $(a + b)(c + d) = ac + ad + bc + bd$

$$(a + b)(c + d) - (ac + bd) = (ac + ad + bc + bd) - (ac + bd)$$

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$



Can we reduce the number of subproblems?

karatsuba

Observation. If we know what $(a + b)(c + d)$ we know what $(ad + bc)$ is!

Explanation. $(a + b)(c + d) = ac + ad + bc + bd$

bingo!

$$(a + b)(c + d) - (\cancel{ac} + \cancel{bd}) = (\cancel{ac} + ad + bc + \cancel{bd}) - (\cancel{ac} + \cancel{bd}) = (ad + bc)$$



only **one** multiplication is needed to find $(ad + bc)$

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$



Can we reduce the number of subproblems?

karatsuba

Observation. If we know what $(a + b)(c + d)$ we know what $(ad + bc)$ is!

Explanation. $(a + b)(c + d) = ac + ad + bc + bd$

bingo!

$$(a + b)(c + d) - (\cancel{ac} + \cancel{bd}) = (\cancel{ac} + ad + bc + \cancel{bd}) - (\cancel{ac} + \cancel{bd}) = (ad + bc)$$



only **one** multiplication is needed to find $(ad + bc)$

$$\begin{aligned} x \times y &= ac \cdot 10^n + (\boxed{ad + bc}) \cdot 10^{\frac{n}{2}} + bd \\ &= ac \cdot 10^n + \boxed{((a + b)(c + d) - (ac + bd))} \cdot 10^{\frac{n}{2}} + bd \end{aligned}$$

Attempt # 4: A Divide & Conquer Algorithm



$$= \underline{ac} \cdot 10^n + (\underline{ad} + \underline{bc}) \cdot 10^{\frac{n}{2}} + \underline{bd}$$

Multiplying two n -digit numbers requires **4** multiplications between numbers of length $\frac{n}{2}$ each!

Running Time. $T(n) = 4T(\frac{n}{2}) + cn = \Theta(n^2)$



Can we reduce the number of subproblems?

karatsuba

Observation. If we know what $(a + b)(c + d)$ we know what $(ad + bc)$ is!

Explanation. $(a + b)(c + d) = ac + ad + bc + bd$

bingo!

$$(a + b)(c + d) - (\cancel{ac} + \cancel{bd}) = (\cancel{ac} + ad + bc + \cancel{bd}) - (\cancel{ac} + \cancel{bd}) = (ad + bc)$$



only **one** multiplication is needed to find $(ad + bc)$

$$x \times y = ac \cdot 10^n + (ad + bc) \cdot 10^{\frac{n}{2}} + bd$$

$$= \boxed{ac} \cdot 10^n + \boxed{((a + b)(c + d)) - (ac + bd)} \cdot 10^{\frac{n}{2}} + \boxed{bd}$$



only **three** multiplications are needed to find $x \times y$

Karatsuba's Algorithm: Implementation

```
// assuming x and y have n digits each and n is a power of 2
MULTIPLY(x, y):
    n = number of digits in x and y
    if (n == 1) return x * y
```

Karatsuba's Algorithm: Implementation

```
// assuming x and y have n digits each and n is a power of 2
MULTIPLY(x, y):
    n = number of digits in x and y
    if (n == 1) return x * y

    a = n/2 left-most digits of x
    b = n/2 right-most digits of x
    c = n/2 left-most digits of y
    d = n/2 right-most digits of y
```

Karatsuba's Algorithm: Implementation

```
// assuming x and y have n digits each and n is a power of 2  
MULTIPLY(x, y):
```

```
    n = number of digits in x and y
```

```
    if (n == 1) return x * y
```

```
    a = n/2 left-most digits of x
```

```
    b = n/2 right-most digits of x
```

```
    c = n/2 left-most digits of y
```

```
    d = n/2 right-most digits of y
```

```
1  ac = MULTIPLY(a, c)
```

```
2  bd = MULTIPLY(b, d)
```

```
3  temp = MULTIPLY(a+b, c+d)
```

Karatsuba's Algorithm: Implementation

```
// assuming x and y have n digits each and n is a power of 2
MULTIPLY(x, y):
```

```
    n = number of digits in x and y
```

```
    if (n == 1) return x * y
```

```
    a = n/2 left-most digits of x
```

```
    b = n/2 right-most digits of x
```

```
    c = n/2 left-most digits of y
```

```
    d = n/2 right-most digits of y
```

```
1 ac = MULTIPLY(a, c)
```

```
2 bd = MULTIPLY(b, d)
```

```
3 temp = MULTIPLY(a+b, c+d)
```

```
return [ac * 10^n] + [(temp - ac - bd) * 10^(n/2)] + bd
```

1

3

1

2

2

Karatsuba's Algorithm: Implementation and Analysis

```
// assuming x and y have n digits each and n is a power of 2
MULTIPLY(x, y):
    n = number of digits in x and y
    if (n == 1) return x * y

    a = n/2 left-most digits of x
    b = n/2 right-most digits of x
    c = n/2 left-most digits of y
    d = n/2 right-most digits of y

    ac = MULTIPLY(a, c)
    bd = MULTIPLY(b, d)
    temp = MULTIPLY(a+b, c+d)

    return [ac * 10^n] + [(temp - ac - bd) * 10^(n/2)] + bd
```



Running Time.

$T(n) = 3T(\frac{n}{2}) + cn$ — time to add, subtract, shift, etc.

Karatsuba's Algorithm: Implementation and Analysis

```
// assuming x and y have n digits each and n is a power of 2
MULTIPLY(x, y):
    n = number of digits in x and y
    if (n == 1) return x * y

    a = n/2 left-most digits of x
    b = n/2 right-most digits of x
    c = n/2 left-most digits of y
    d = n/2 right-most digits of y

    ac = MULTIPLY(a, c)
    bd = MULTIPLY(b, d)
    temp = MULTIPLY(a+b, c+d)

    return [ac * 10^n] + [(temp - ac - bd) * 10^(n/2)] + bd
```



Running Time.

$$T(n) = 3T\left(\frac{n}{2}\right) + cn = \Theta(n^{\log_2 3}) \quad \text{using the master method}$$
$$= O(n^{1.59})$$



Karatsuba's Algorithm: Implementation and Analysis

```
// assuming x and y have n digits each and n is a power of 2
MULTIPLY(x, y):
    n = number of digits in x and y
    if (n == 1) return x * y

    a = n/2 left-most digits of x
    b = n/2 right-most digits of x
    c = n/2 left-most digits of y
    d = n/2 right-most digits of y

    ac = MULTIPLY(a, c)
    bd = MULTIPLY(b, d)
    temp = MULTIPLY(a+b, c+d)

    return [ac * 10^n] + [(temp - ac - bd) * 10^(n/2)] + bd
```



Running Time.

$$\begin{aligned} T(n) &= 3T\left(\frac{n}{2}\right) + cn = \Theta(n^{\log_2 3}) \\ &= O(n^{1.59}) \end{aligned}$$



Is it worth the effort?

Yes if n is large (>100)

No if n is small.

Integer Multiplication: A Race for Efficiency!

year	algorithm	bit operations
12xx	grade school	$O(n^2)$
1962	Karatsuba-Ofman	$O(n^{1.585})$
1963	Toom-3, Toom-4	$O(n^{1.465}), O(n^{1.404})$
1966	Toom-Cook	$O(n^{1+\epsilon})$
1971	Schönhage-Strassen	$O(n \log n \cdot \log \log n)$
2007	Fürer	$n \log n 2^{O(\log^* n)}$
2018	Harvey-van der Hoeven	$O(n \log n \cdot 2^{2 \lg^* n})$
2026	 OpenAI	$O(n \log^{1-\kappa} n), \quad \kappa = 2^{-182}$
	???	$O(n)$

Remark. GNU Multiple Precision library uses one of first five algorithms depending on n .

