COL 351 : Analysis & Design of Algorithms

LECTURE 1

INTRODUCTION TO ALGORITHMS

JULY 23, 2024 | ROHIT VAISH

COURSE WEBPAGE

WHAT IS AN ALGORITHM ?

WHAT IS AN ALGORITHM ?

WHAT IS AN ALGORITHM ?

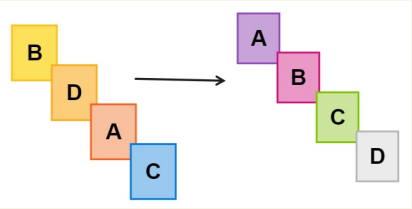
Any piece of code ?

WHAT IS AN ALGORITHM ?

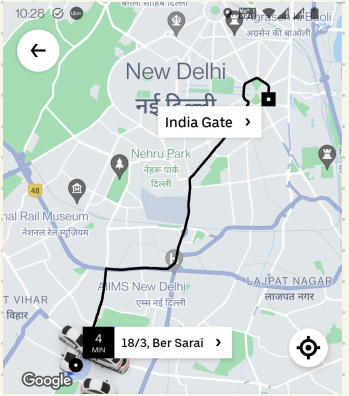
Any piece of code ?

♡ A recipe for solving some computational problem.

EXAMPLES



Sorting a list of objects

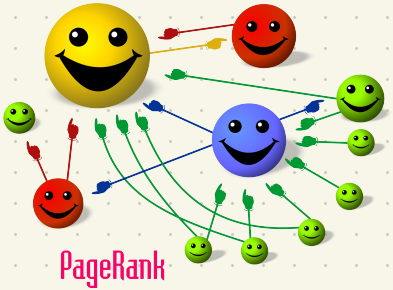


finding shortest path between two places

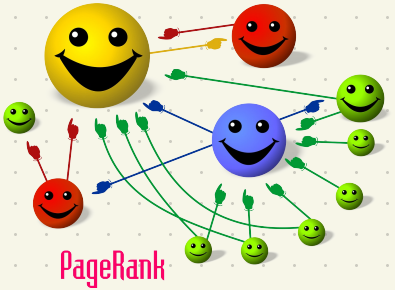
Completing tasks with deadlines

KILLER APPS

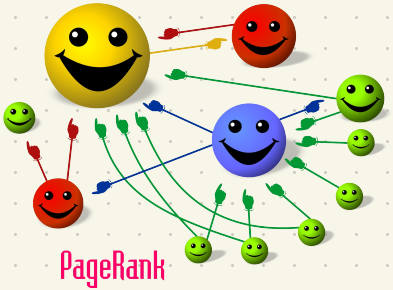
KILLER APPS



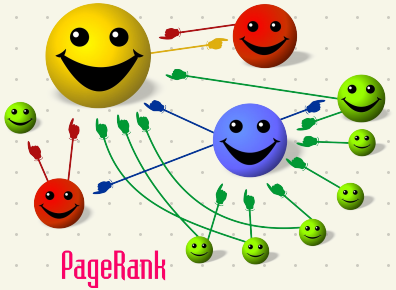
KILLER APPS



KILLER APPS



KILLER APPS



WHY SHOULD YOU STUDY ALGORITHMS ?

WHY SHOULD YOU STUDY ALGORITHMS ?

* Important for all branches of computer science

Networks	Cryptography	Graphics	Databases
shortest-path algorithms	number-theoretic algorithms	geometric algorithms	search tree data structures

WHY SHOULD YOU STUDY ALGORITHMS ?

* Important for all branches of computer science

Networks	Cryptography	Graphics	Databases
shortest-path algorithms	number-theoretic algorithms	geometric algorithms	search tree data structures

* A "lens" for non-CS areas

Physics	Economics	Biology
quantum computation	understanding "equilibria"	evolution as a search algorithm

WHY SHOULD YOU STUDY ALGORITHMS ?

* Important for all branches of computer science

Networks	Cryptography	Graphics	Databases
shortest-path algorithms	number-theoretic algorithms	geometric algorithms	search tree data structures

* A "lens" for non-CS areas

Physics	Economics	Biology
quantum computation	understanding "equilibria"	evolution as a search algorithm

* Challenging puzzles — you will feel smarter!

WHY SHOULD YOU STUDY ALGORITHMS ?

* Important for all branches of computer science

Networks	Cryptography	Graphics	Databases
shortest-path algorithms	number-theoretic algorithms	geometric algorithms	search tree data structures

* A "lens" for non-CS areas

Physics	Economics	Biology
quantum computation	understanding "equilibria"	evolution as a search algorithm

* Challenging puzzles — you will feel smarter!

* Fun !

CAKE - CUTTING

What is a fair way to divide a cake between two people?

CAKE - CUTTING

What is a fair way to divide a cake between two people?

Person 1 cuts the cake into two parts they consider equal in value.

Person 2 chooses their favorite part and Person 1 gets the leftovers.

Integer Addition

INTEGER ADDITION

input: two n digit numbers x and y

output: the sum $x + y$

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,

$$\begin{array}{r} 5\ 6\ 7\ 8 \\ 1\ 2\ 3\ 4 \\ \hline \\ \hline \end{array}$$

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,

$$\begin{array}{cccc} 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline \\ \hline \end{array}$$

Grade - school addition

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,

$$\begin{array}{cccc} & & 1 & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline & & & 2 \\ \hline \end{array}$$

Grade - school addition

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,

$$\begin{array}{cccc} & {\scriptstyle 1} & {\scriptstyle 1} & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline & & 1 & 2 \\ \hline \end{array}$$

Grade - school addition

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,

$$\begin{array}{cccc} & {\scriptstyle 1} & {\scriptstyle 1} & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline & 9 & 1 & 2 \\ \hline \end{array}$$

Grade - school addition

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,

$$\begin{array}{cccc} & {\scriptstyle 1} & {\scriptstyle 1} & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline 6 & 9 & 1 & 2 \\ \hline \end{array}$$

Grade - school addition

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,

$$\begin{array}{cccc} & {\scriptstyle 1} & {\scriptstyle 1} & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline 6 & 9 & 1 & 2 \end{array}$$

Grade - school addition

This algorithm is correct.

INTEGER ADDITION

input: two n digit numbers x and y

output: the sum $x + y$

e.g.,

$$\begin{array}{cccc} & {\scriptstyle 1} & {\scriptstyle 1} & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline 6 & 9 & 1 & 2 \\ \hline \end{array}$$

Grade-school addition

This algorithm is correct.

How much time does it take?

INTEGER ADDITION

input: two n digit numbers x and y

output: the sum $x + y$

e.g.,

$$\begin{array}{cccc} & {\scriptstyle 1} & {\scriptstyle 1} & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline 6 & 9 & 1 & 2 \\ \hline \end{array}$$

basic operation

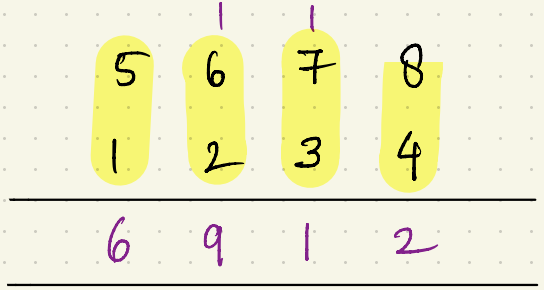
adding two single-digit numbers

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,



basic operation

adding two single-digit numbers

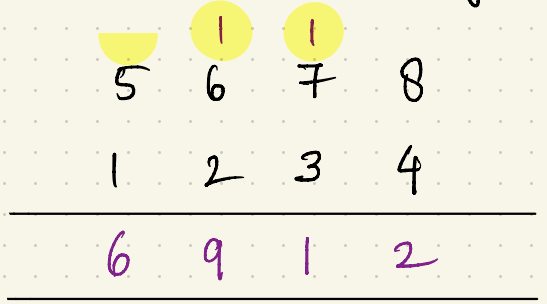
n single-digit addition operations

INTEGER ADDITION

input: two n digit numbers x and y

output: the sum $x + y$

e.g.,



basic operation

adding two single-digit numbers

n single-digit addition operations

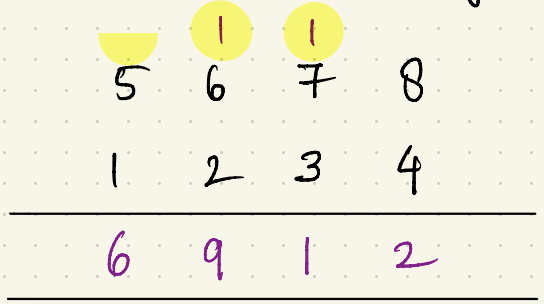
$n-1$ carry operations (at most)

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum $x + y$

e.g.,



basic operation

adding two single-digit numbers

n single-digit addition operations

$n-1$ carry operations (at most)

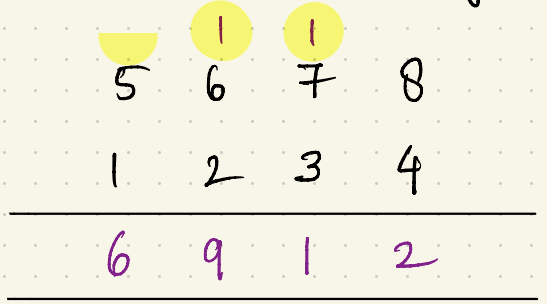
adding a 2-digit and a 1-digit number

INTEGER ADDITION

input : two n digit numbers x and y

output : the sum x + y

e.g.,



basic operation

adding two single-digit numbers

n single-digit addition operations

{ n-1 carry operations (at most)
{ adding a 2-digit and a 1-digit number

$\leq 2(n-1)$ single-digit addition operations

INTEGER ADDITION

input: two n digit numbers x and y

output: the sum $x+y$

e.g.,

$$\begin{array}{cccc} & {\scriptstyle 1} & {\scriptstyle 1} & \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline 6 & 9 & 1 & 2 \end{array}$$

basic operation

adding two single-digit numbers

n single-digit addition operations

$\leq 3n$ operations overall

$\begin{cases} n-1 \text{ carry operations (at most)} \\ \text{adding a 2-digit and a 1-digit number} \end{cases}$

$\leq 2(n-1)$ single-digit addition operations

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$\begin{array}{cccc} 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline \end{array}$$

Grade-school algorithm

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$\begin{array}{cccc} 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ \hline 2 & 2 & 7 & 1 & 2 \end{array}$$

Grade-school algorithm

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$\begin{array}{r} 5\ 6\ 7\ 8 \\ 1\ 2\ 3\ 4 \\ \hline 2\ 2\ 7\ 1\ 2 \\ 1\ 7\ 0\ 3\ 4 \end{array}$$

Grade-school algorithm

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{cccc}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 &
\end{array}
$$

Grade-school algorithm

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$\begin{array}{r} 5\ 6\ 7\ 8 \\ \times\ 1\ 2\ 3\ 4 \\ \hline 2\ 2\ 7\ 1\ 2 \\ 1\ 7\ 0\ 3\ 4 \\ 1\ 1\ 3\ 5\ 6 \\ 5\ 6\ 7\ 8 \end{array}$$

Grade-school algorithm

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 & \\
5 & 6 & 7 & 8 & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2
\end{array}
$$

Grade-school algorithm

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 & \\
5 & 6 & 7 & 8 & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2
\end{array}
$$

Grade-school algorithm

This algorithm is correct.

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrrrrr}
 & & & 5 & 6 & 7 & 8 & \\
 & & & 1 & 2 & 3 & 4 & \\
\hline
 & & 2 & 2 & 7 & 1 & 2 & \\
 & 1 & 7 & 0 & 3 & 4 & & \\
1 & 1 & 3 & 5 & 6 & & & \\
5 & 6 & 7 & 8 & & & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2 &
\end{array}
$$

Grade-school algorithm

This algorithm is correct.

How much time does it take?

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 & \\
5 & 6 & 7 & 8 & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2
\end{array}
$$

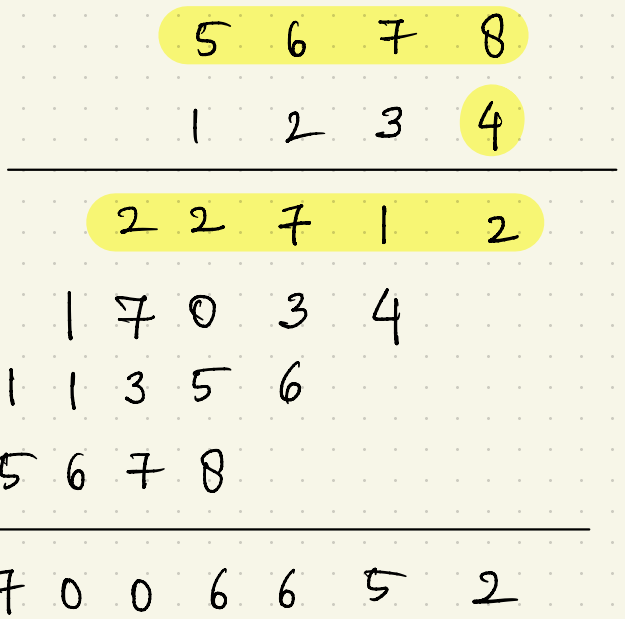
basic operations: add or multiply two single-digit numbers

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,



basic operations: add or multiply two single-digit numbers

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{r}
5\ 6\ 7\ 8 \\
\times\ \ 1\ 2\ 3\ 4 \\
\hline
2\ 2\ 7\ 1\ 2 \\
1\ 7\ 0\ 3\ 4 \\
1\ 1\ 3\ 5\ 6 \\
5\ 6\ 7\ 8 \\
\hline
7\ 0\ 0\ 6\ 6\ 5\ 2
\end{array}
$$

basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations

- n multiplications
- $\leq 2n$ additions (including carry)

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{r}
5\ 6\ 7\ 8 \\
1\ 2\ 3\ 4 \\
\hline
2\ 2\ 7\ 1\ 2 \\
1\ 7\ 0\ 3\ 4 \\
1\ 1\ 3\ 5\ 6 \\
5\ 6\ 7\ 8 \\
\hline
7\ 0\ 0\ 6\ 6\ 5\ 2
\end{array}
$$

basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{r}
5\ 6\ 7\ 8 \\
1\ 2\ 3\ 4 \\
\hline
2\ 2\ 7\ 1\ 2 \\
1\ 7\ 0\ 3\ 4 \\
1\ 1\ 3\ 5\ 6 \\
5\ 6\ 7\ 8 \\
\hline
7\ 0\ 0\ 6\ 6\ 5\ 2
\end{array}
$$

basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{r}
5\ 6\ 7\ 8 \\
1\ 2\ 3\ 4 \\
\hline
2\ 2\ 7\ 1\ 2 \\
1\ 7\ 0\ 3\ 4 \\
1\ 1\ 3\ 5\ 6 \\
5\ 6\ 7\ 8 \\
\hline
7\ 0\ 0\ 6\ 6\ 5\ 2
\end{array}
$$

basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 & \\
5 & 6 & 7 & 8 & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2
\end{array}
$$

basic operations: add or multiply two single-digit numbers

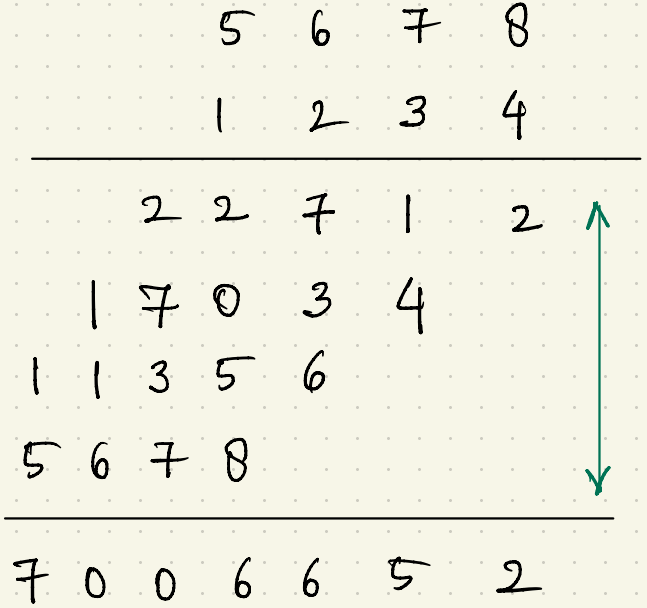
$\leq 3n$ operations per row

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,



basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations per row

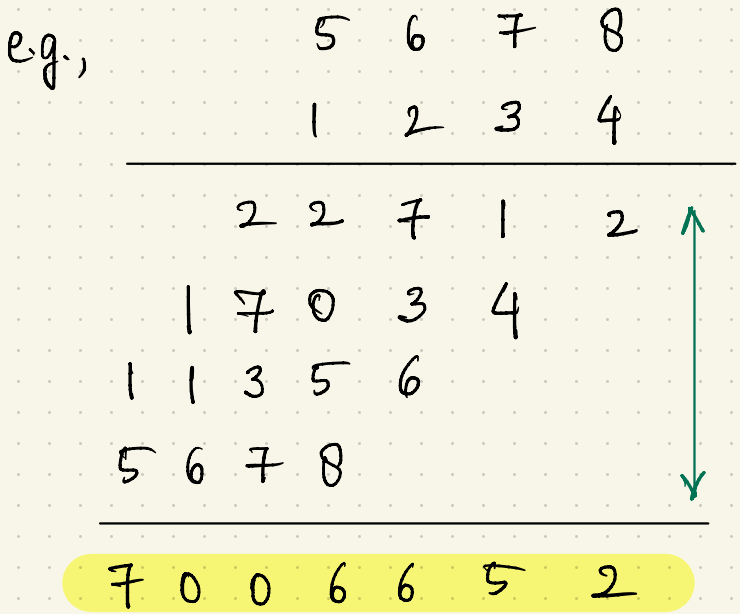
n rows

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,



basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations per row

n rows

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 \quad 2 \\
 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 \\
5 & 6 & 7 & 8 & \\
\hline
7 & 0 & 0 & 6 & 6 \quad 5 \quad 2
\end{array}
$$

basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations per row

n rows

$\leq 6n^2$ operations (Exercise)

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

```
          5 6 7 8
          1 2 3 4
-----------------
        2 2 7 1 2
      1 7 0 3 4
    1 1 3 5 6
  5 6 7 8
-----------------
7 0 0 6 6 5 2
```

basic operations: add or multiply two single-digit numbers

$\leq 3n$ operations per row

n rows

$\leq 9n^2$ operations overall

$\leq 6n^2$ operations (Exercise)

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 & \\
5 & 6 & 7 & 8 & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2
\end{array}
$$

High-level message

$$\#\text{ operations} \leq \underbrace{\text{constant}}_{\leq 9} \cdot n^2$$

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 & \\
5 & 6 & 7 & 8 & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2
\end{array}
$$

High-level message

$$\#\text{ operations} \leq \underbrace{\text{constant}}_{\leq 9} \cdot n^2$$

double the input size

$\Rightarrow$ no. of operations quadruples (at most)

Integer Multiplication

input: two n digit numbers x and y

output: the product $x \cdot y$

e.g.,

$$
\begin{array}{rrrrr}
 & 5 & 6 & 7 & 8 \\
 & 1 & 2 & 3 & 4 \\
\hline
 & 2 & 2 & 7 & 1 & 2 \\
 & 1 & 7 & 0 & 3 & 4 \\
1 & 1 & 3 & 5 & 6 & \\
5 & 6 & 7 & 8 & & \\
\hline
7 & 0 & 0 & 6 & 6 & 5 & 2
\end{array}
$$

High-level message

$$\#\text{ operations} \leq \underbrace{\text{constant}}_{\leq 9} \cdot n^2$$

Can we do better?

The algorithm design space is **surprisingly rich**!

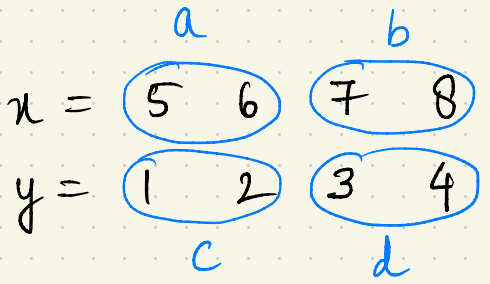
KARATSUBA MULTIPLICATION

KARATSUBA MULTIPLICATION

$x = 5\ 6\ 7\ 8$

$y = 1\ 2\ 3\ 4$

KARATSUBA MULTIPLICATION



KARATSUBA MULTIPLICATION

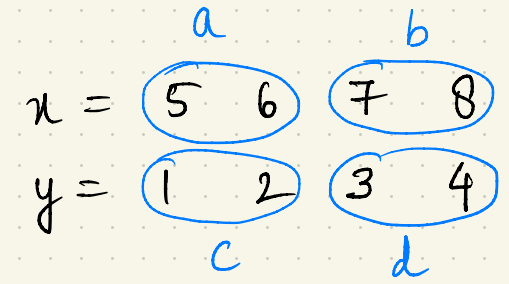
(1) compute $a \cdot c = 56 \times 12 = 672$

$x =$ (5 6) (7 8) — a, b

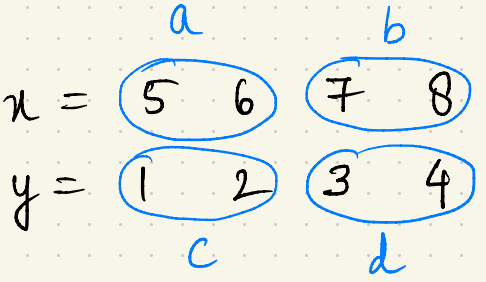
$y =$ (1 2) (3 4) — c, d

KARATSUBA MULTIPLICATION

1. compute $a \cdot c = 56 \times 12 = 672$
2. compute $b \cdot d = 78 \times 34 = 2652$



KARATSUBA MULTIPLICATION



1. compute $a \cdot c = 56 \times 12 = 672$
2. compute $b \cdot d = 78 \times 34 = 2652$
3. compute $(a+b) \cdot (c+d) = 134 \times 46 = 6164$

KARATSUBA MULTIPLICATION

$x = \overbrace{5\ 6}^{a}\ \overbrace{7\ 8}^{b}$

$y = \overbrace{1\ 2}^{c}\ \overbrace{3\ 4}^{d}$

1. compute $a \cdot c = 56 \times 12 = 672$
2. compute $b \cdot d = 78 \times 34 = 2652$
3. compute $(a+b) \cdot (c+d) = 134 \times 46 = 6164$
4. compute (3) − (2) − (1) = 2840

KARATSUBA MULTIPLICATION

$x = \overbrace{5\ 6}^{a}\ \overbrace{7\ 8}^{b}$

$y = \underbrace{1\ 2}_{c}\ \underbrace{3\ 4}_{d}$

① compute $a \cdot c = 56 \times 12 = 672$

② compute $b \cdot d = 78 \times 34 = 2652$

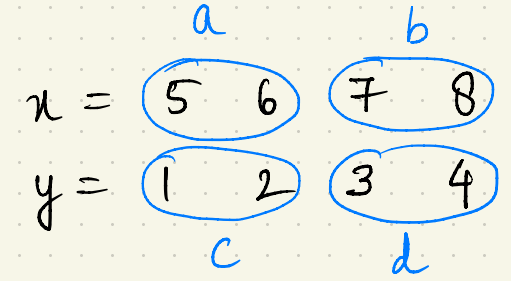
③ compute $(a+b) \cdot (c+d) = 134 \times 46 = 6164$

④ compute ③ − ② − ① = 2840

⑤ compute

$$\begin{array}{rl} 6720000 & \text{from } ① \\ 2652 & \text{from } ② \\ 284000 & \text{from } ④ \\ \hline \end{array}$$

KARATSUBA MULTIPLICATION



1. compute $a \cdot c = 56 \times 12 = 672$
2. compute $b \cdot d = 78 \times 34 = 2652$
3. compute $(a+b) \cdot (c+d) = 134 \times 46 = 6164$
4. compute (3) − (2) − (1) = 2840
5. compute

$$
\begin{array}{rl}
6720000 & \text{from (1)} \\
2652 & \text{from (2)} \\
2840000 & \text{from (4)} \\
\hline
7006652 &
\end{array}
$$

KARATSUBA MULTIPLICATION

$x = \overbrace{56}^{a}\ \overbrace{78}^{b}$

$y = \overbrace{12}^{c}\ \overbrace{34}^{d}$

① compute $a \cdot c = 56 \times 12 = 672$

② compute $b \cdot d = 78 \times 34 = 2652$

③ compute $(a+b) \cdot (c+d) = 134 \times 46 = 6164$

④ compute ③ − ② − ① = 2840

⑤ compute

$$
\begin{array}{rl}
6720000 & \text{from ①} \\
2652 & \text{from ②} \\
2840000 & \text{from ④} \\
\hline
7006652 & = x \cdot y
\end{array}
$$

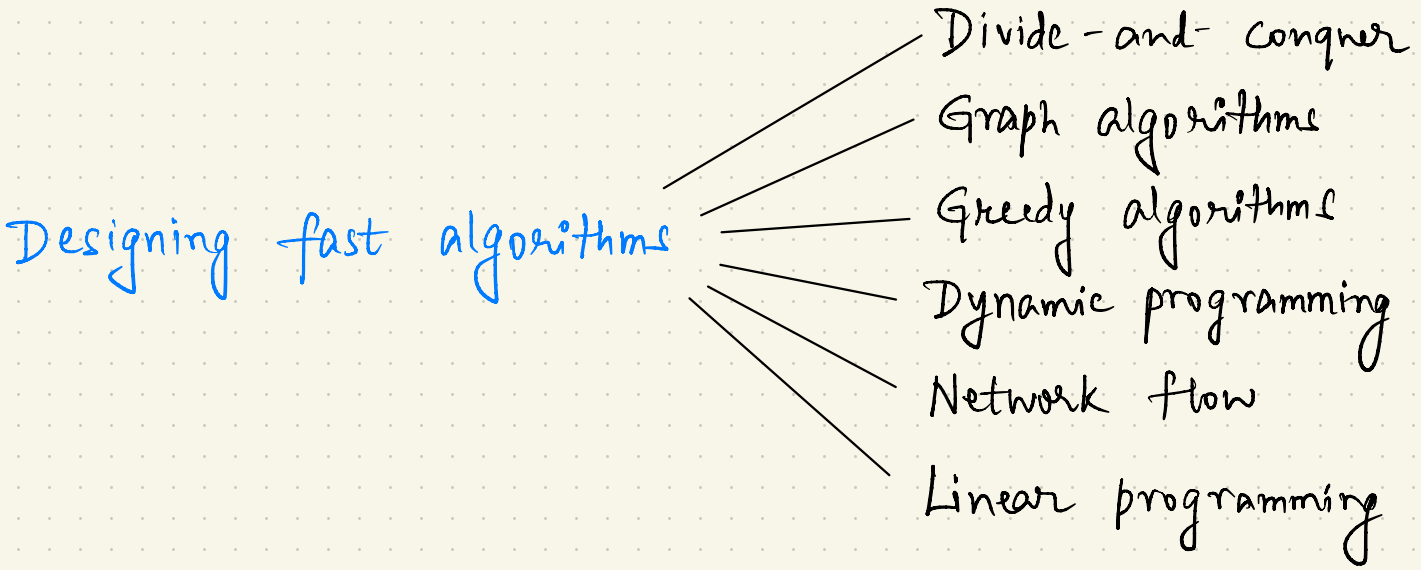
KARATSUBA MULTIPLICATION

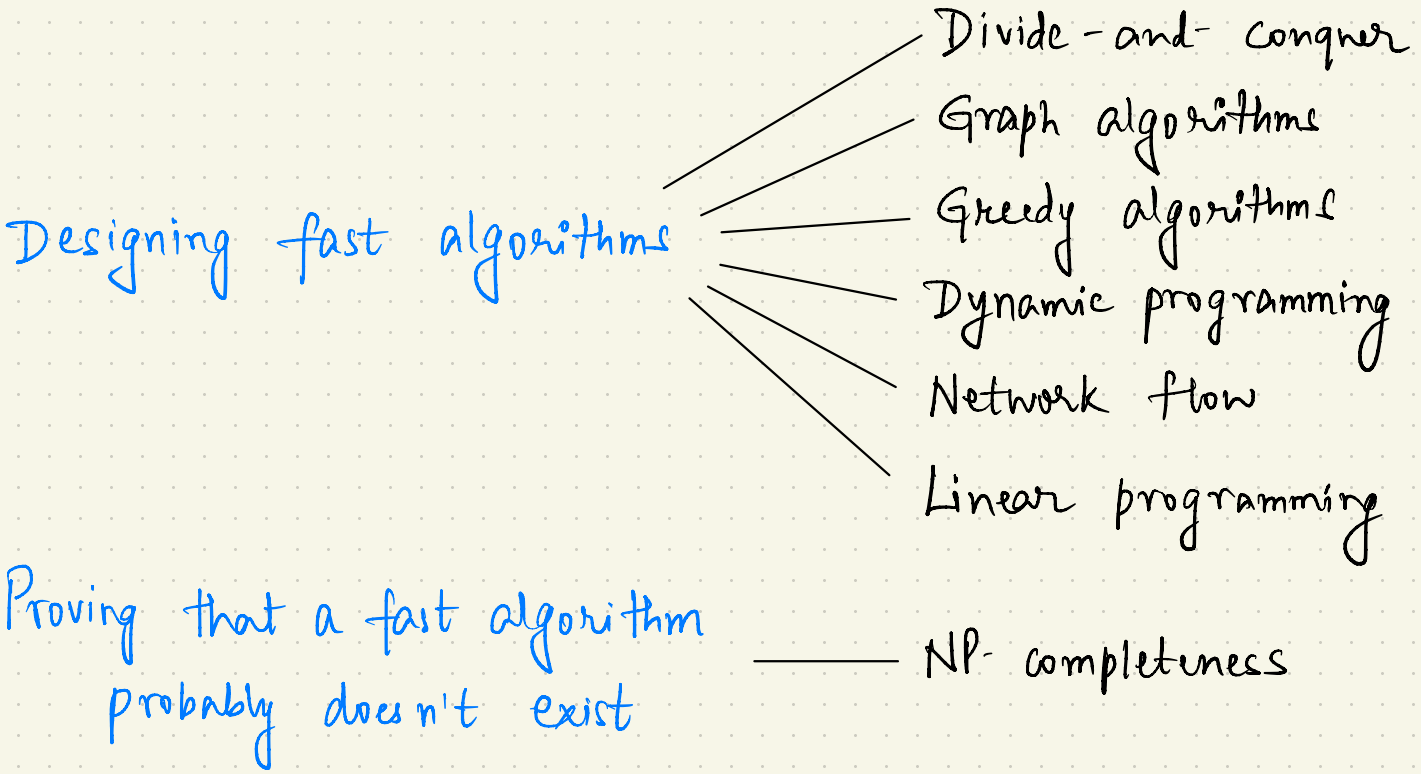
$x = \overbrace{56}^{a}\ \overbrace{78}^{b}$

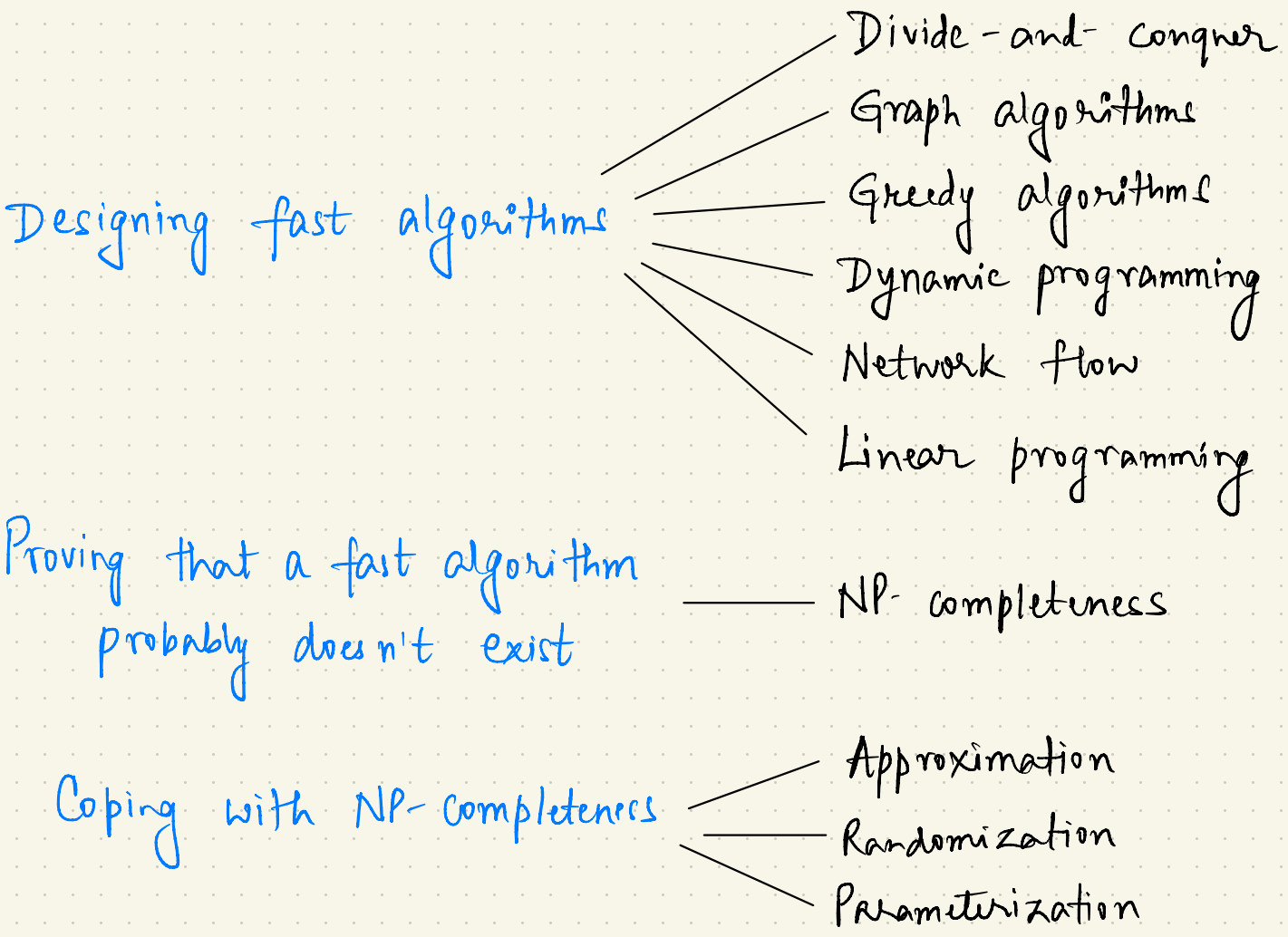
$y = \underbrace{12}_{c}\ \underbrace{34}_{d}$

1. compute $a \cdot c = 56 \times 12 = 672$
2. compute $b \cdot d = 78 \times 34 = 2652$
3. compute $(a+b) \cdot (c+d) = 134 \times 46 = 6164$
4. compute (3) − (2) − (1) = 2840
5. compute

$$
\begin{array}{rl}
6720000 & \text{from (1)} \\
2652 & \text{from (2)} \\
284000 & \text{from (4)} \\
\hline
7006652 & = x \cdot y
\end{array}
$$


Designing fast algorithms
Divide-and-conquer
Graph algorithms
Greedy algorithms
Dynamic programming
Network flow
Linear programming


Designing fast algorithms
Divide-and-conquer
Graph algorithms
Greedy algorithms
Dynamic programming
Network flow
Linear programming
Proving that a fast algorithm probably doesn't exist
NP-completeness


Designing fast algorithms
Divide-and-conquer
Graph algorithms
Greedy algorithms
Dynamic programming
Network flow
Linear programming
Proving that a fast algorithm probably doesn't exist
NP-completeness
Coping with NP-completeness
Approximation
Randomization
Parameterization

COURSE LOGISTICS

COURSE LOGISTICS

INSTRUCTOR : ROHIT VAISH (Please call me Rohit)

→ Lectures on Tue/Wed/Fri 10-11 AM LH 121

COURSE LOGISTICS

INSTRUCTOR : ROHIT VAISH (Please call me Rohit)

↳ Lectures on Tue/Wed/Fri 10-11 AM LH 121

TAs : LAKSHAY SAGGI

RAMANUJ GOEL

GEETANSH JUNEJA

SAMYAK JAIN

PRATIK BULANI

GAURAV RAJPUT

ADITHYA NARAYAN

SHIVANSH GARG

VAIBHAV KATENDRA

Tutorials on Mon/Tue/Thurs/Fri 1-2 PM LH 408 + Office hours

COURSE LOGISTICS

INSTRUCTOR : ROHIT VAISH (Please call me Rohit)

↳ Lectures on Tue/Wed/Fri 10-11 AM LH 121

TAs : LAKSHAY SAGGI

RAMANUJ GOEL

GEETANSH JUNEJA

SAMYAK JAIN

PRATIK BULANI

GAURAV RAJPUT

ADITHYA NARAYAN

SHIVANSH GARG

VAIBHAV KATENDRA

Tutorials on Mon/Tue/Thurs/Fri 1-2 PM LH 408 + Office hours

Check your tutorial group on the website (by July 27)

EVALUATION

~ 2 hrs	MINOR	20 %	
~ 2 hrs	MAJOR	20 %	
~ 45 mins	QUIZZES	36 %	[12% × best 3 out of 4]
~ 10 mins	TUTORIALS	24 %	[3% × best 8 out of 12]

EVALUATION

~ 2 hrs	MINOR	20 %	
~ 2 hrs	MAJOR	20 %	
~ 45 mins	QUIZZES	36 %	[12% × best 3 out of 4]
~ 10 mins	TUTORIALS	24 %	[3% × best 8 out of 12]

AUDITS Not allowed

EVALUATION

~ 2 hrs	MINOR	20 %	
~ 2 hrs	MAJOR	20 %	
~ 45 mins	QUIZZES	36 %	[12% × best 3 out of 4]
~ 10 mins	TUTORIALS	24 %	[3% × best 8 out of 12]

AUDITS — Not allowed

ATTENDANCE — Not monitored by me (but might be required by the institute)

↓

based on above-mentioned evaluations

RE- EVALUATION

RE- EVALUATION

* Missed QUIZ :
  in-class / tutorial

* Missed MINOR :
  and/or MAJOR

* RE-MAJOR :
  with E-grade

RE- EVALUATION

* Missed QUIZ : No re-quiz
in-class / tutorial

* Missed MINOR
and/or MAJOR :

* RE-MAJOR
with E-grade :

RE- EVALUATION

* Missed QUIZ : No re-quiz
in-class / tutorial

* Missed MINOR : A Common make-up exam (20%)
and/or MAJOR Prior approval before the date of exam(s)
and ≥ 75% attendance

* RE-MAJOR :
with E-grade

RE- EVALUATION

* Missed QUIZ : No re-quiz
in-class / tutorial

* Missed MINOR : A Common make-up exam (20%)
and/or MAJOR Prior approval before the date of exam(s)
and ≥ 75% attendance

* RE-MAJOR : ≥ 75% attendance
with E-grade

TUTORIALS

* Once (almost) every week

TUTORIALS

* Once (almost) every week
* First ~ 10 mins: Quiz

TUTORIALS

* Once (almost) every week
* First ~ 10 mins: Quiz 1 problem from weekly problem set

TUTORIALS

* Once (almost) every week

* First ~ 10 mins: Quiz 1 problem from weekly problem set

announced by Thursday of previous week

announced by TA at the start of the tutorial

TUTORIALS

* Once (almost) every week

* First ~ 10 mins: Quiz — 1 problem from weekly problem set

  (weekly problem set: announced by Thursday of previous week)

  (1 problem: announced by TA at the start of the tutorial)

* Remaining time: Discussion (Ask questions!)

TUTORIALS

* Once (almost) every week
* First ~ 10 mins: Quiz — 1 problem from weekly problem set
  - weekly problem set: announced by Thursday of previous week
  - 1 problem: announced by TA at the start of the tutorial
* Remaining time: Discussion (Ask questions!)
* Each tutorial submission is worth 3% (best eight).

TUTORIALS

* Once (almost) every week
* First ~ 10 mins: Quiz 1 problem from weekly problem set
  - weekly problem set: announced by Thursday of previous week
  - 1 problem: announced by TA at the start of the tutorial
* Remaining time: Discussion (Ask questions!)
* Each tutorial submission is worth 3% (best eight).
* Problem not asked in quiz $\neq$ Not relevant

REGRADING

* Gradescope : Sign up only with @ dept.iitd, not @ iitd

* "Frivolous" requests : −2% in overall score for first one

  Subsequently −4%, −8%, −16%, ...

If in doubt, talk to your group's TA.

PLAGIARISM

Any kind, at any stage : F grade

If in doubt, talk to your TA or me.

NO GARBAGE

(Credit : Allan Borodin, Nisarg Shah)

NO GARBAGE

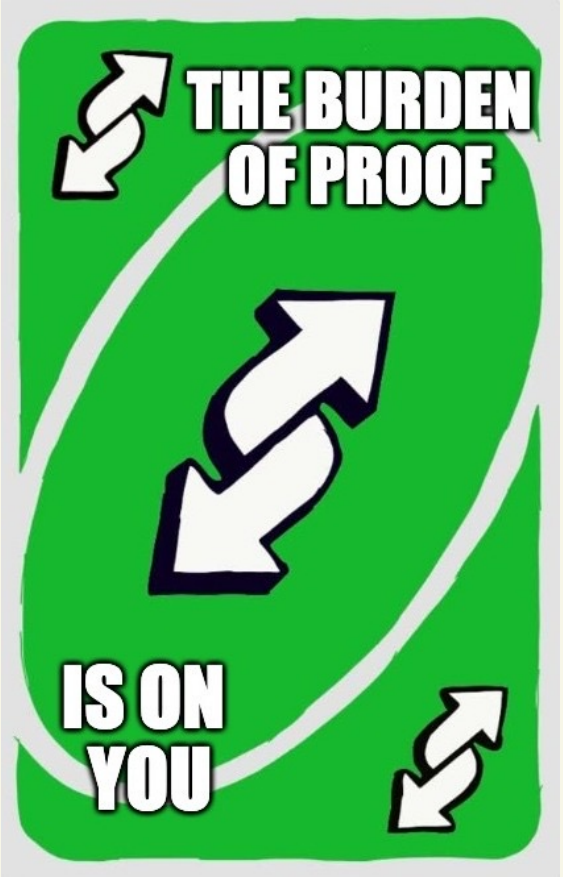
Partial marks for promising approaches [can be extended to give correct answer]

Zero marks if answer makes no sense or is hard to read

20% marks for "I do not know how to approach this problem."

NOTE : $20 > 0$

(Credit : Allan Borodin, Nisarg Shah)


THE BURDEN
OF PROOF
IS ON
YOU

COMMUNICATION

* **Most Important!**

* Please reach out to me or TAs about ANY issues at any time. Do not wait until the majors.

* We want you to enjoy learning about algorithms and do well in the course.

All the best!