

# Design and Analysis of Algorithms

CS202

# Today's class

- Asymptotic Notation
- Local minimum exercises
- Loop Invariants

## Find the input size

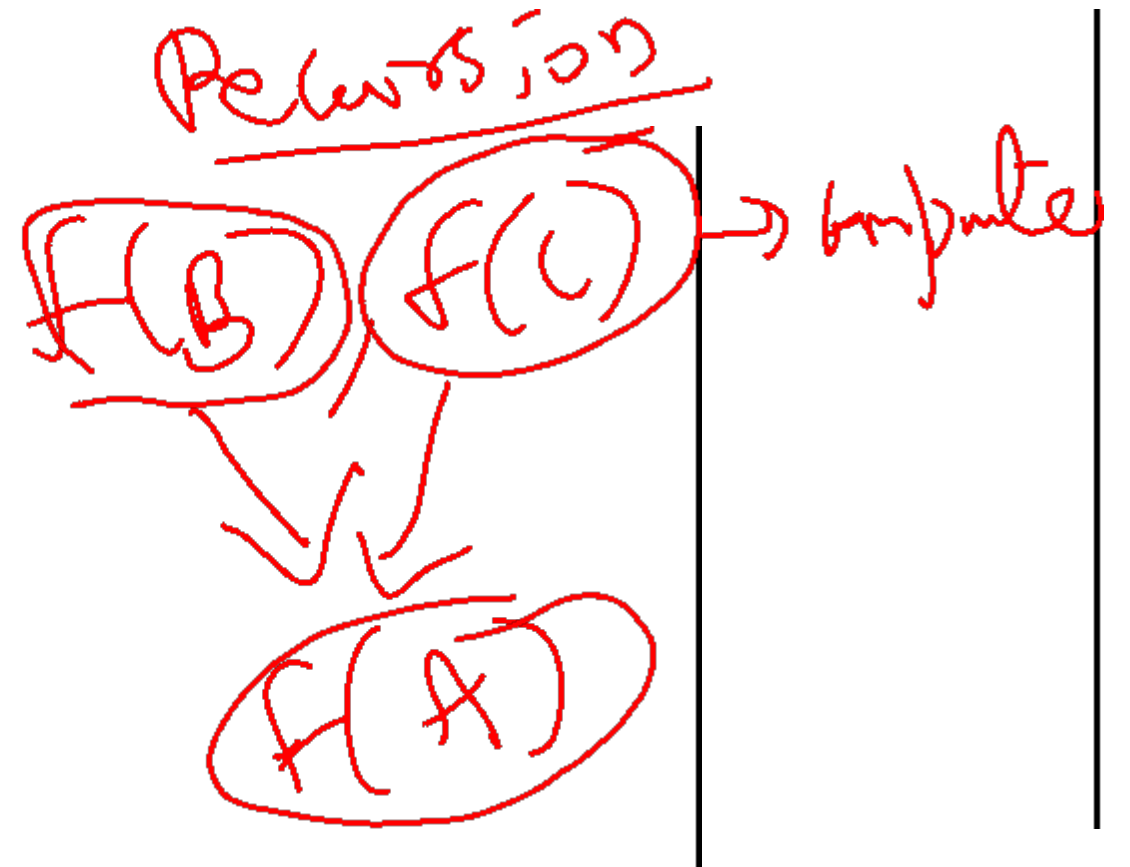
1. Input:  $x, y$  in  $\{1, 2, \dots, n\}$ . Output:  $xy$   
 $2 \log_2 n$

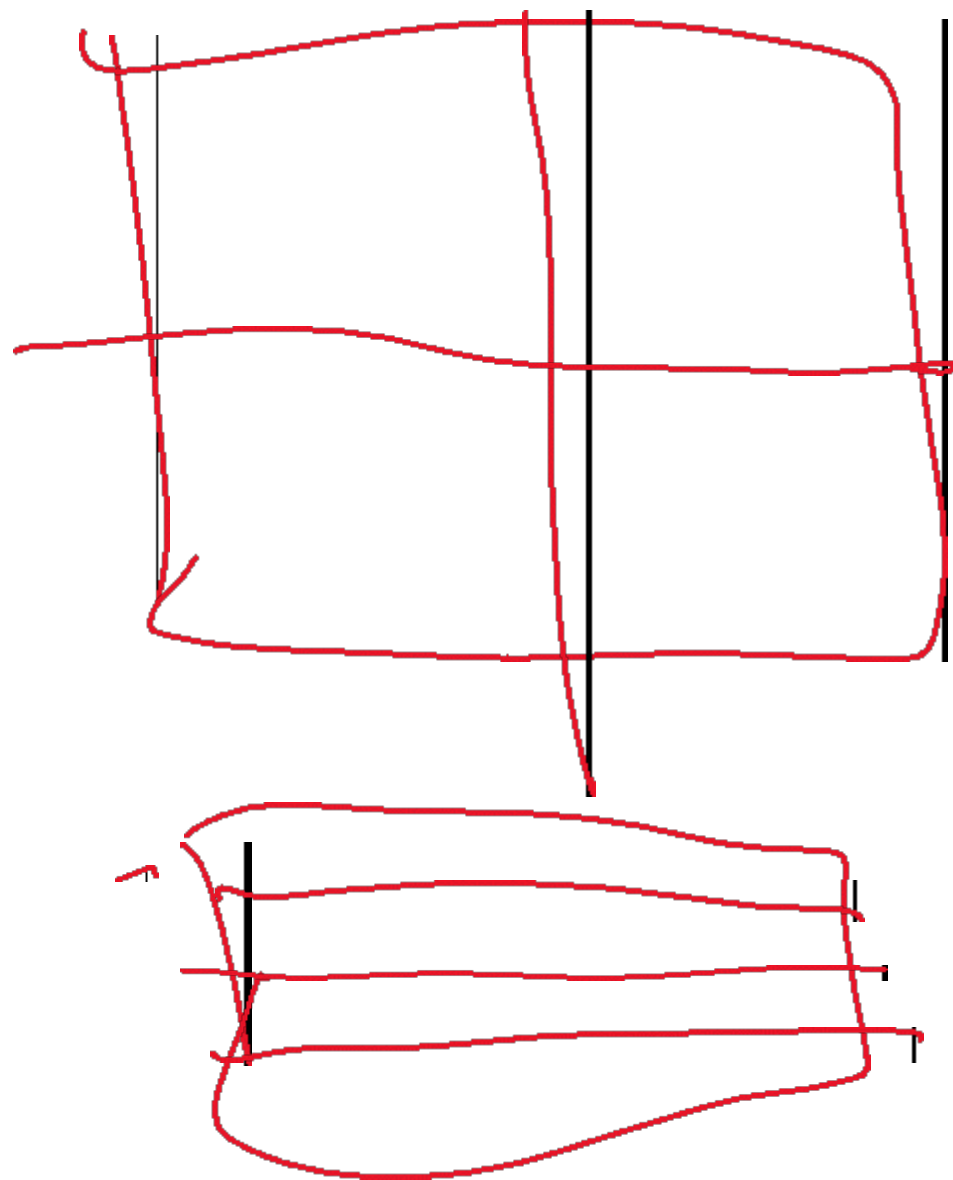
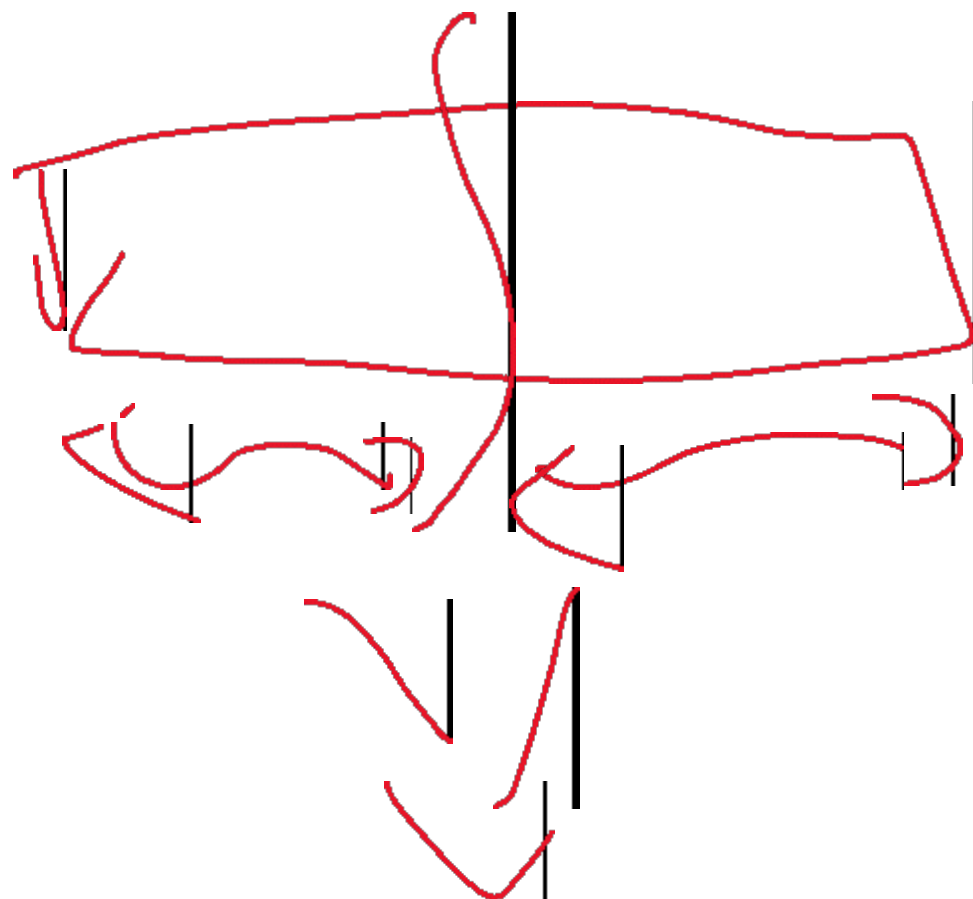
2. Input:  $A, B$ :  $n$  by  $n$  matrix. Output:  $AB$   
 $(n^2)$

Input size:  $2n^2$  (number of elements)\*

# Divide & Conquer Strategy

Identical to original  
 $b$   $c$   
 $\swarrow \searrow$   
 $f(A)$





E.g. 1:

Given a positive integer  $n$ , find  $\sqrt{n}$ .

$i = 1$

While ( $i * i < n$ )

Increment  $i$

Return ( $i - 1$ )

Example

4, 9, 16, 25, 36

$O(\sqrt{n})$

Linear search

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50

E.x. 1:

Finding a local minimum'

3, 7, 0, -2, 12, 8, 10, 1, 4, 2

A local minimum is an element  $A[i]$   
such that it is:

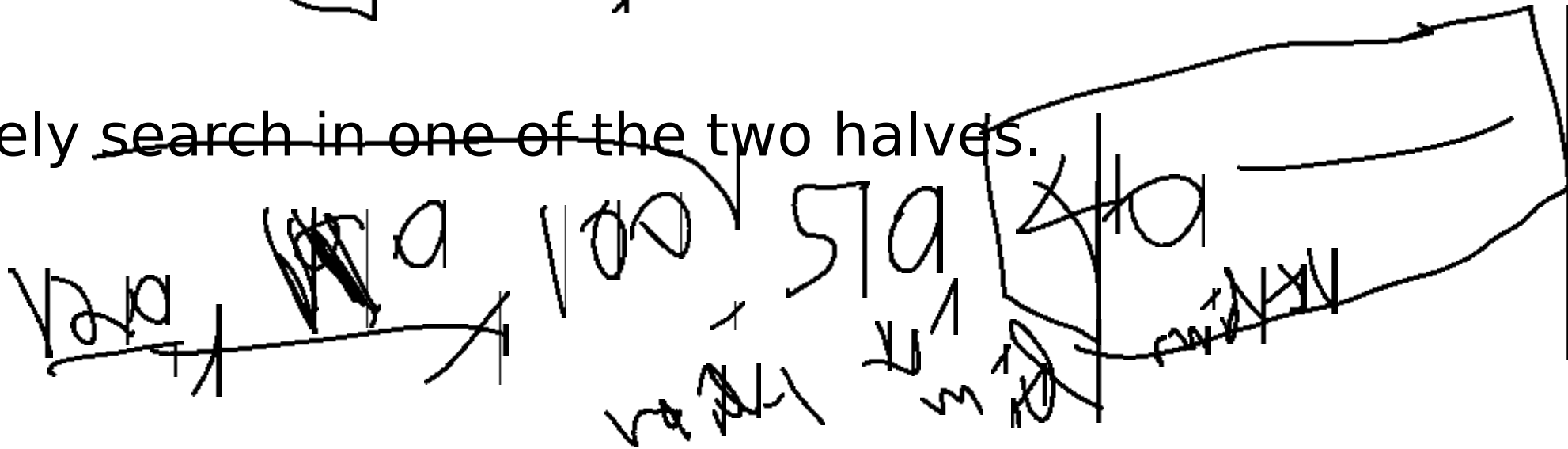
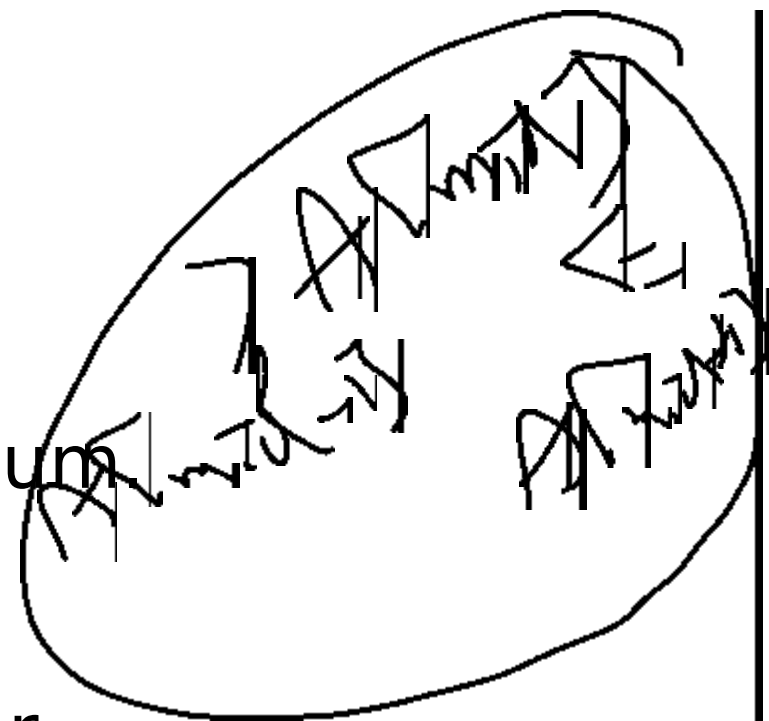
$A[i] \leq A[i-1]$  and  $A[i] \leq A[i+1]$

## Algorithm for local minimum

1. Compare the middle element with its neighbors to check if it is a local minimum.
2. Suppose it is not.

$A[mid] > A[mid-1]$  or  $A[mid] > A[mid+1]$  (or both)

Recursively search in one of the two halves.





If  $A[mid+1] < A[mid]$ , then the right half of the array contains at least one local minimum.

Proof:

The smallest element of the right half is a local minimum.

## Algo to find local minimum

1. Compare middle element with its neighbors. If it is the local minimum, we are done.
2. Else if  $A[\text{mid}-1] < A[\text{mid}]$ , then recursively search in  $A[\text{first}, \text{mid}-1]$
3. Else ( $A[\text{mid}+1] < A[\text{mid}]$ ), recursively search in  $A[\text{mid}+1, \text{last}]$

Handwritten notes at the top:  $n, 1, 2, 3, 4, 1$

Algo to find local minimum

First=1, last=n

While (first < last)

If  $A[mid]$  is a local minimum, return mid.

Else if  $A[mid-1] < A[mid]$ , set last=mid-1

Else if  $A[mid+1] < A[mid]$ , set first=mid+1

Return first.

Handwritten notes on the left: A diagram of a binary search tree with nodes 1, 2, 3, 4, 1. The root is 1, with children 2 and 4. Node 2 has children 3 and 1. Node 4 has children 2 and 1. The tree is drawn with arrows indicating the search path.

Handwritten notes at the bottom:  $1, 2, 3, 4, 1$

Algo for standard binary search:  $A[1] \leq \dots \leq A[n]$   
and  $x$

First=1, last=n

While (first<last)

    If ( $x=A[\text{mid}]$ ), return mid.

    Else if  $x < A[\text{mid}]$ , set last=mid-1

    Else if  $x > A[\text{mid}]$ , set first=mid+1

If ( $x=A[\text{first}]$ ) return first, else return Not found.

Modify it to find the least index  $I$  such that  $A[i] \leq x$   
2,4,6,8,10,12 and  $x=11$ . Output: 5

# Comparisons in finding

minimum:  $n-1$

Every comparison

1  
2  
3

eliminated

Min (height)

low/sea

Min (height)

## Asymptotic Notation: Examples

1.  $4n^2 + 5n - 10 = O(n^2)$

2.  $n^3 - n^2 + 5n + 6 = O(n^3)$

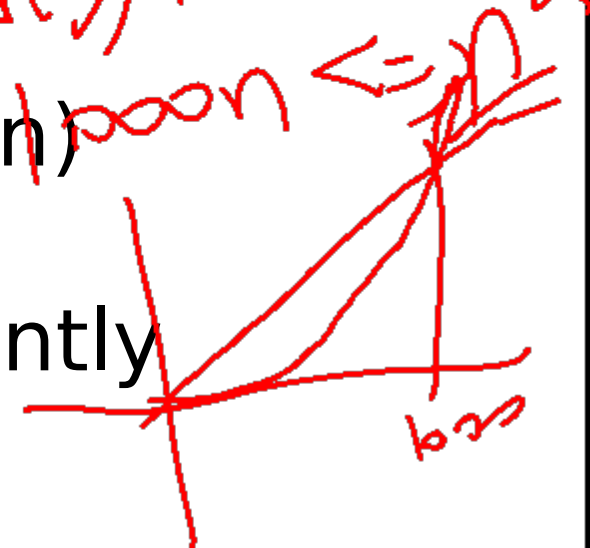
3.  $n^2 + n \log n + \sqrt{n} = O(n^2)$

4.  $2^n + n^{10} + \log n = O(2^n)$

5.  $\log n = o(n)$ ;  $n = o(n^2)$ ;  $n^2 = o(2^n)$

$F(n) = O(g(n))$   $f(n) \leq cg(n)$  for sufficiently large  $n$

$\log n = o(n)$



# Asymptotic Notation: Arrange in increasing order

1.  $n^2 + n$  2.  $\log^2 n$  3.  $n / \log n$

4.  $5n + 4$  5.  $\log \log n$  6.  $\sqrt{n}$

7.  $2^n$  8.  $(\log n)^{\log n}$

Sqrt(n) vs  $n / \log n$

1 vs  $\sqrt{n} / \log n$

