Design & Analysis of Algorithms

Algorithm: Well defined computational procedure that takes some inputs and provide some outputs.

Objective: Designing Algorithms Efficiently.

Ref: Introduction to Algorithms (Cormen et al.)

Example:

Algorithm: Sorting n numbers

Input: Some sequence $\langle a_1, a_2, \ldots, a_n \rangle$

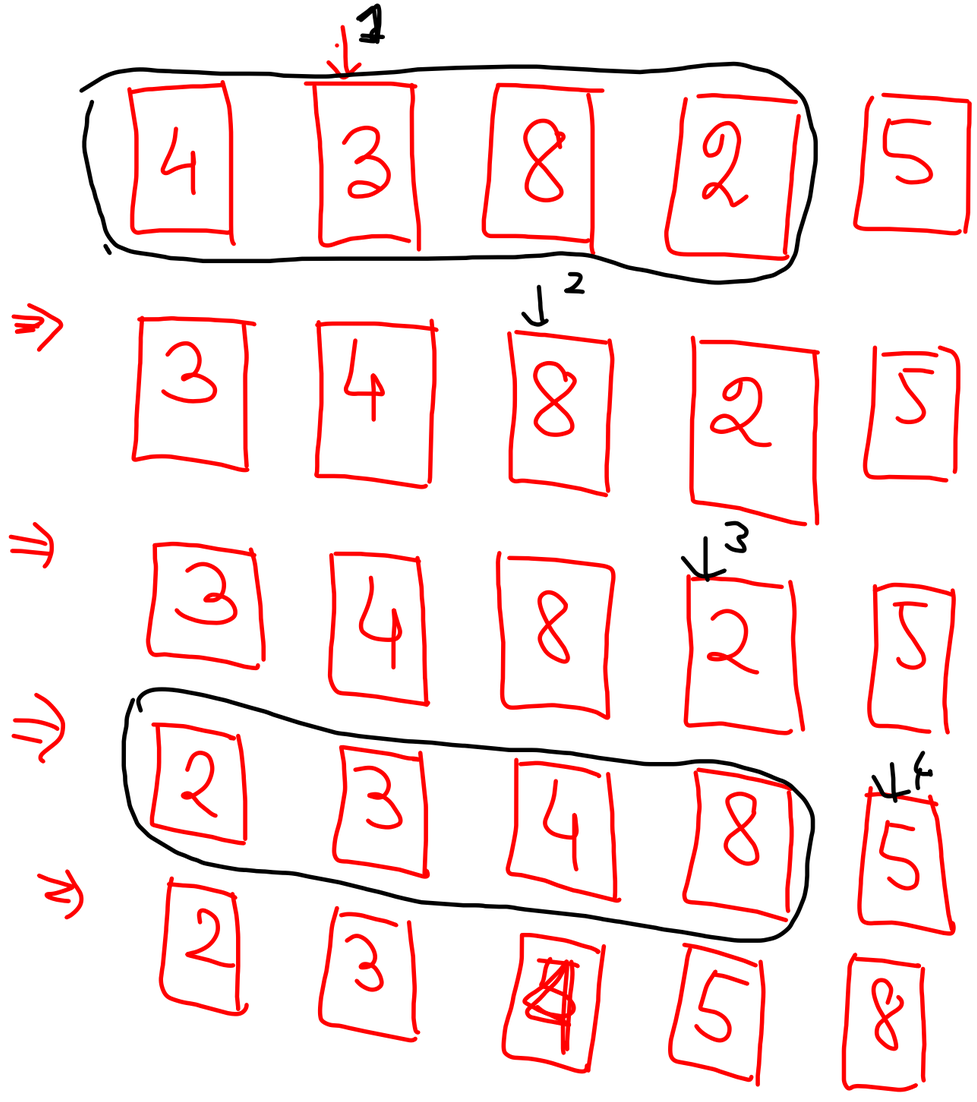
Output: A permutation of the numbers $\langle a_1', a_2', \ldots, a_n' \rangle$

s.t. $a_1' \leq a_2' \leq \ldots \leq a_n'$

5 4 6 8
↳ 4 5 6 8

- Correctness
- Time Complexity

Insertion Sort



$A[1..n]$

Pseudocode

```
1.  For j = 2 .. n do
2.      cur ← A[j]
3.      i ← j-1
4.      while (( A[i] > cur) & (i>0))
5.
            A[i+1] = A[i]
6.          i = i-1
7.      A[i+1] = cur
```

Correctness

Loop Invariant

Identify some properties (P)

- Initialization: 'P' holds the beginning of
- Maintenance: Show that if at j^{th} iteration → (P) holds
  then at the beginning of $(j+1)^{th}$ loop
- Termination: While terminating, (P) holds → (P) holds

N+1

$N \geq 8 \Rightarrow$ You can give an exact exchange

[3] [5]

Induction → base case, $8 = 5+3$

→ N → (*) — $x_1 \to 5$, $x_2 \to 3$ if $x_1 > 0$

Add 2 '3'-coin
Remove 1 '5'-coin
else

Add 2 '5'-coin
Remove 3 '3'-coin

Insertion Sort

Loop Invariant:

- At j^{th} step,
  - $A[1 \ldots (j-1)]$ contains the initial subarray $A[1 \ldots j-1]$
  - $A[1 \ldots (j-1)]$ is now sorted.
- Termination $\Rightarrow$ $j = n+1$
  $\Rightarrow A[1 \ldots n]$ is sorted

Selection Sort

$A[1, \ldots, n]$

At j^{th} step,

- Consider $A[j, \ldots, n]$
- Find min ; Say $A[k] \Rightarrow$ min
- Swap $(A[j], A[k])$.

Find Correctness
- Loop Invariant
-

Time Complexity

Complexity

Time Complexity

- In terms of size of the input.

Program
⇕
Prog. Languages
⇕
Computer

Algorithms
⇕
Pseudocode
⇕
Computational Model

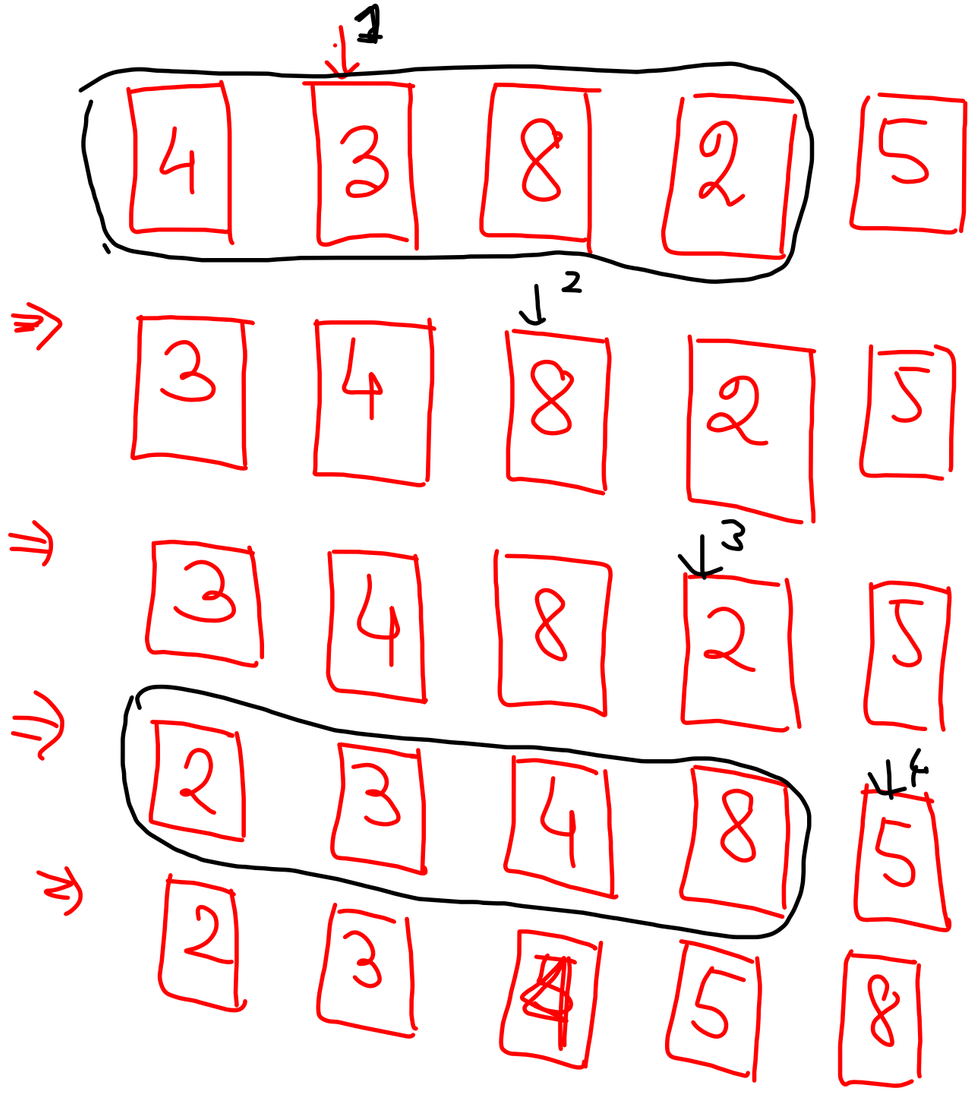
Random Access Machine Model (RAM)

- All the instructions / statements in the pseudo code executes sequentially.

- Operations → - Arithmatic operations
  - Branch statements
  - load, store

} All these operations take constant time.

Insertion Sort



$A[1..n]$

Pseudocode

1.	For $j = 2..n$ do	(n)
2.	$cur \leftarrow A[j]$	$(n-1)$
3.	$i \leftarrow j-1$	$(n-1)$
4.	while $((A[i] > cur) \& (i>0))$	$\hookrightarrow \sum_{j=2}^{n} t_j$
5.	$A[i+1] = A[i]$	$\hookrightarrow \sum_{j=2}^{n} (t_j - 1)$
6.	$i = i-1$	
7.	$A[i+1] = cur$	$\hookrightarrow (n-1)$

Time Complexity of Insertion Sort

$$T(n) = c_1 \cdot n + c_2 \cdot (n-1) + c_3 \cdot (n-1) + c_4 \cdot \sum_{j=2}^{n} t_j + c_5 \cdot \sum_{j=2}^{n} (t_j - 1) + c_6 \cdot \sum_{j=2}^{n} (t_j - 1) + c_7 \cdot (n-1)$$

$$= \left(c \cdot n + d \cdot \sum_{j=2}^{n} t_j \right) + e$$

Best Case: Initial array Sorted $\Rightarrow (t_j = 1) \Rightarrow T(n) = d_1 \cdot n + d_2 = \Theta(n)$

Worst Case: Initial array reversely sorted $(t_j = j) \Rightarrow T(n) = d_4 \cdot n^2 + d_2 \cdot n + d_3 = \Theta(n^2)$