Brute Force

A straightforward approach, usually based directly on the problem's statement and definitions of the concepts involved

Examples:

1. Computing $a^n$ ($a > 0$, $n$ a nonnegative integer)

2. Computing $n!$

3. Multiplying two matrices

4. Searching for a key of a given value in a list
Sorting
Brute-Force Sorting Algorithm

**Selection Sort**  Scan the array to find its smallest element and swap it with the first element. Then, starting with the second element, scan the elements to the right of it to find the smallest among them and swap it with the second elements. Generally, on pass $i$ $(0 \leq i \leq n-2)$, find the smallest element in $A[i..n-1]$ and swap it with $A[i]$:

$$A[0] \leq \ldots \leq A[i-1] \ | \ A[i], \ldots, A[\text{min}], \ldots, A[n-1]$$

in their final positions
Analysis of Selection Sort

**ALGORITHM**  
\textit{SelectionSort}(A[0..n - 1])

// Sorts a given array by selection sort  
// Input: An array A[0..n - 1] of orderable elements  
// Output: Array A[0..n - 1] sorted in ascending order  
for $i \leftarrow 0$ to $n - 2$ do  
    $\textit{min} \leftarrow i$  
    for $j \leftarrow i + 1$ to $n - 1$ do  
        if $A[j] < A[\textit{min}]$  
            $\textit{min} \leftarrow j$  
    swap $A[i]$ and $A[\textit{min}]$

Time efficiency:

Space efficiency:

Stability:
Bubble Sort

5 1 4 2 8
(51428) → (15428),
(15428) → (14528),
(14528) → (14258),
(14258) → (14258),
(14258) → (14258)
(14258) → (12458),
(12458) → (12458)
(12458) → (12458)
(1 2 4 5 8) → (1 2 4 5 8)
(1 2 4 5 8) → (1 2 4 5 8)
(1 2 4 5 8) → (1 2 4 5 8)
(1 2 4 5 8) → (1 2 4 5 8)
Brute-Force String Matching

- **pattern**: a string of \( m \) characters to search for
- **text**: a (longer) string of \( n \) characters to search in
- **problem**: find a substring in the text that matches the pattern

**Brute-force algorithm**

**Step 1** Align pattern at beginning of text

**Step 2** Moving from left to right, compare each character of pattern to the corresponding character in text until
  - all characters are found to match (successful search); or
  - a mismatch is detected

**Step 3** While pattern is not found and the text is not yet exhausted, realign pattern one position to the right and repeat Step 2
Examples of Brute-Force String Matching

Pattern: 001011
Text: 10010101101001100101111010

Pattern: happy
Text: It is never too late to have a happy childhood.
Pseudocode and Efficiency

```
ALGORITHM BruteForceStringMatch(T[0..n − 1], P[0..m − 1])

// Implements brute-force string matching
// Input: An array T[0..n − 1] of n characters representing a text and
// an array P[0..m − 1] of m characters representing a pattern
// Output: The index of the first character in the text that starts a
// matching substring or −1 if the search is unsuccessful
for i ← 0 to n − m do
    j ← 0
    while j < m and P[j] = T[i + j] do
        j ← j + 1
    if j = m return i
return −1
```

Efficiency:
Examples of Brute-Force String Matching

Pattern: 0010011
Text: 10010101101001100101111010