



# DESIGN AND ANALYSIS OF ALGORITHMS

---

**Shylaja S S**

Department of Computer Science  
& Engineering

# DESIGN AND ANALYSIS OF ALGORITHMS

---

## Brute Force: Selection Sort

Major Slides Content: Anany Levitin

**Shylaja S S**

Department of Computer Science & Engineering

A background image showing a long, multi-arched bridge spanning a body of water under a hazy, overcast sky. The bridge's arches are reflected in the calm water.

When in doubt, use  
brute force.

Ken Thompson

- Brute Force is a straightforward approach to solving a problem, usually directly based on the problem statement and definitions of the concepts involved

- In computer science, **brute-force search** or **exhaustive search**, also known as **generate and test**, is a very general problem-solving technique and algorithmic paradigm that consists of:
  - systematically enumerating all possible candidates for the solution
  - checking whether each candidate satisfies the problem's statement

- A brute-force algorithm to find the divisors of a natural number  $n$  would
  - enumerate all integers from 1 to  $n$
  - check whether each of them divides  $n$  without remainder
- A brute-force approach for the eight queens puzzle would
  - examine all possible arrangements of 8 pieces on the 64-square chessboard
  - check whether each (queen) piece can attack any other, for each arrangement
- The brute-force method for finding an item in a table (linear search)
  - checks all entries of the table, sequentially, with the item

- A brute-force search is simple to implement, and will always find a solution if it exists
- But, its cost is proportional to the number of candidate solutions – which in many practical problems tends to grow very quickly as the size of the problem increases (Combinatorial explosion)
- Brute-force search is typically used
  - when the problem size is limited
  - when there are problem-specific heuristics that can be used to reduce the set of candidate solutions to a manageable size
  - when the simplicity of implementation is more important than speed

- Selection Sort
- Bubble Sort

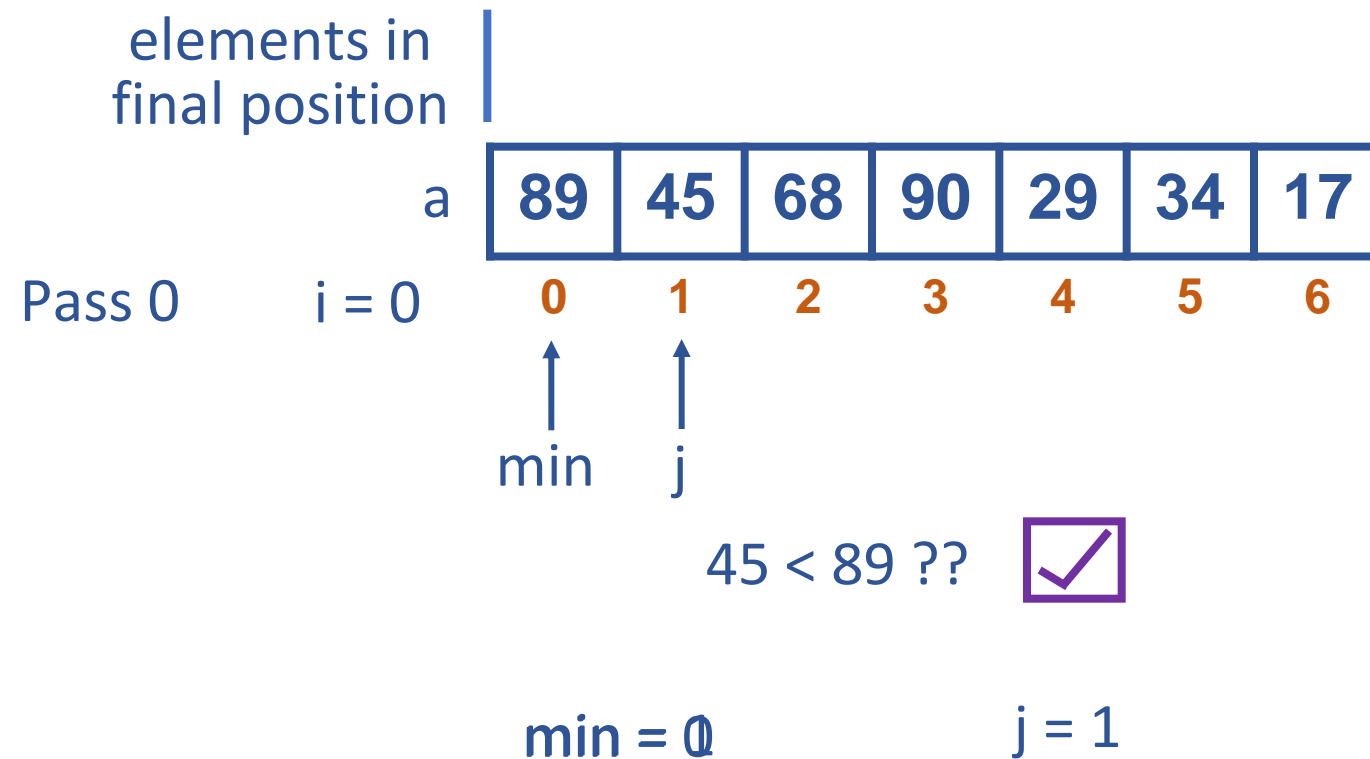


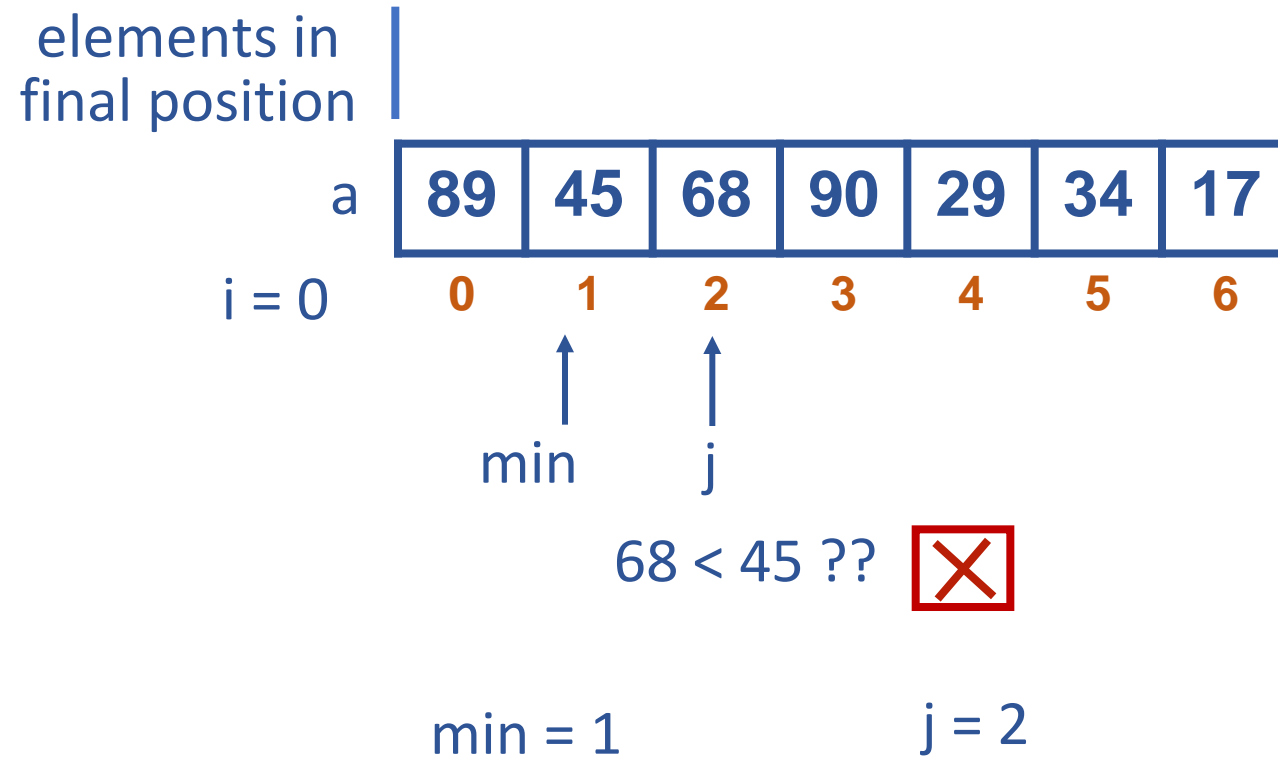
## Selection Sort

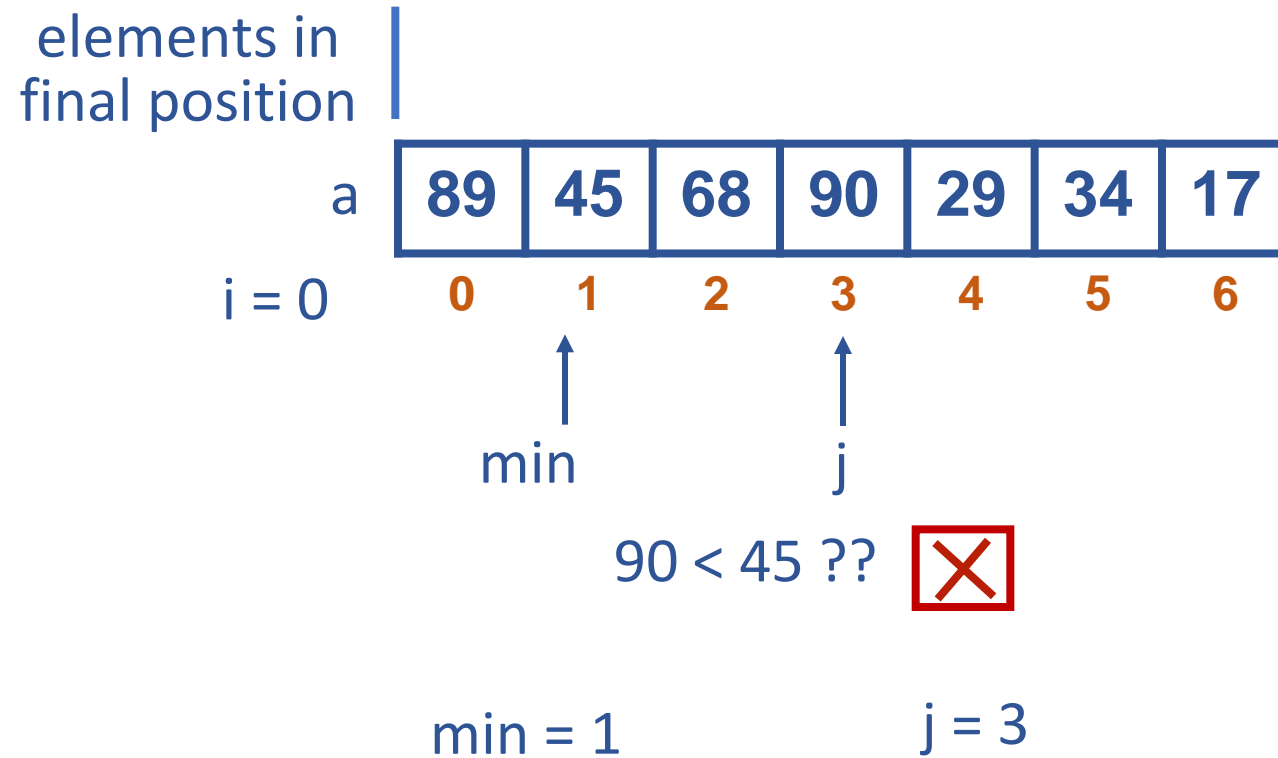
- Scan the array to find its smallest element and swap it with the first element, putting the smallest element in its final position in the sorted list
- Then, starting with the second element, scan the elements to the right of it to find the smallest among them and swap it with the second element, putting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

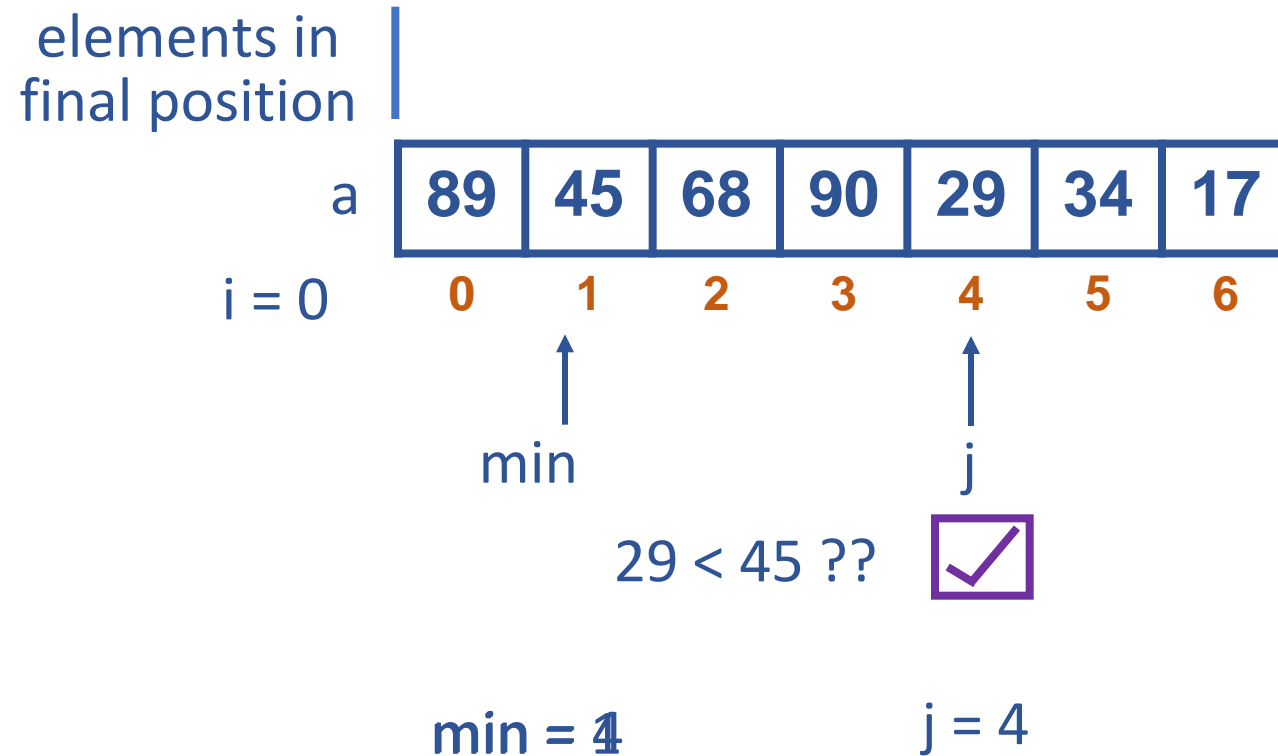


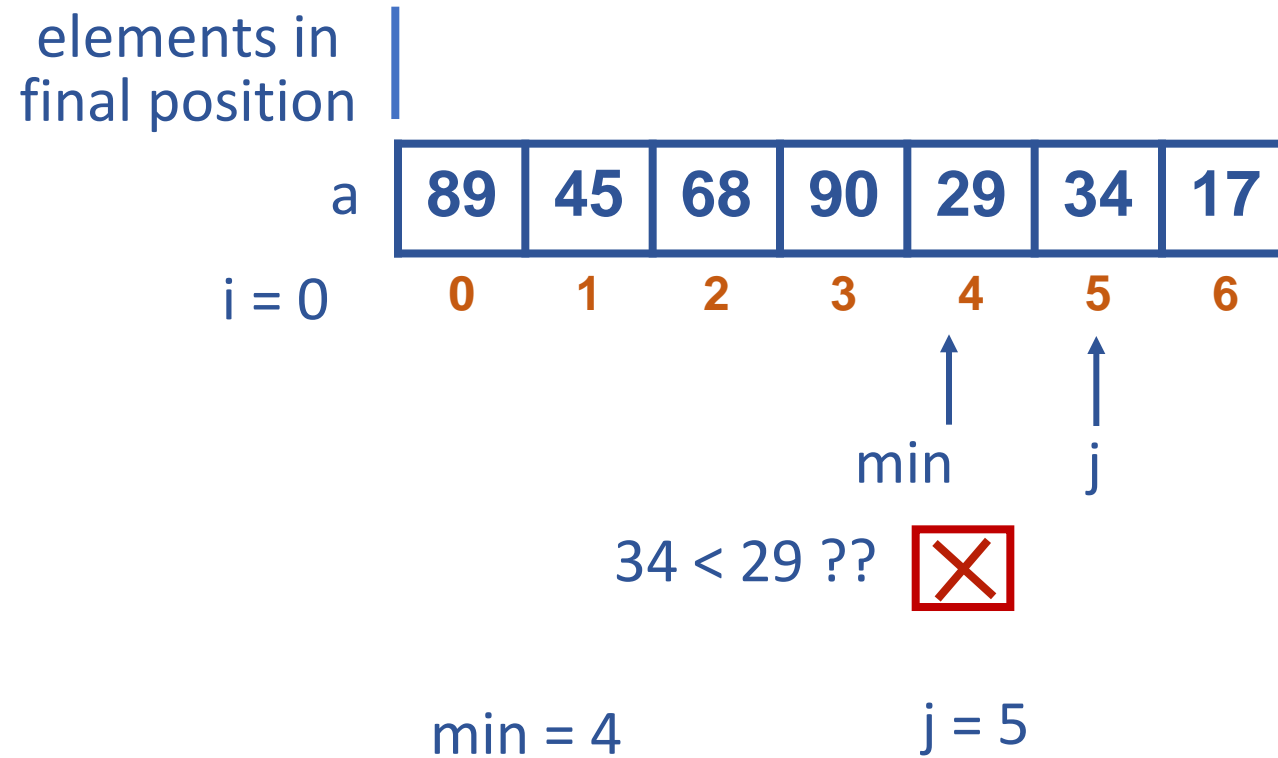
- $A[0] \leq A[1] \leq A[2] \dots \leq A[i-1] \mid A[i], \dots, A[\text{min}], \dots, A[n-1]$
- in their final positions      the last  $n - i$  elements

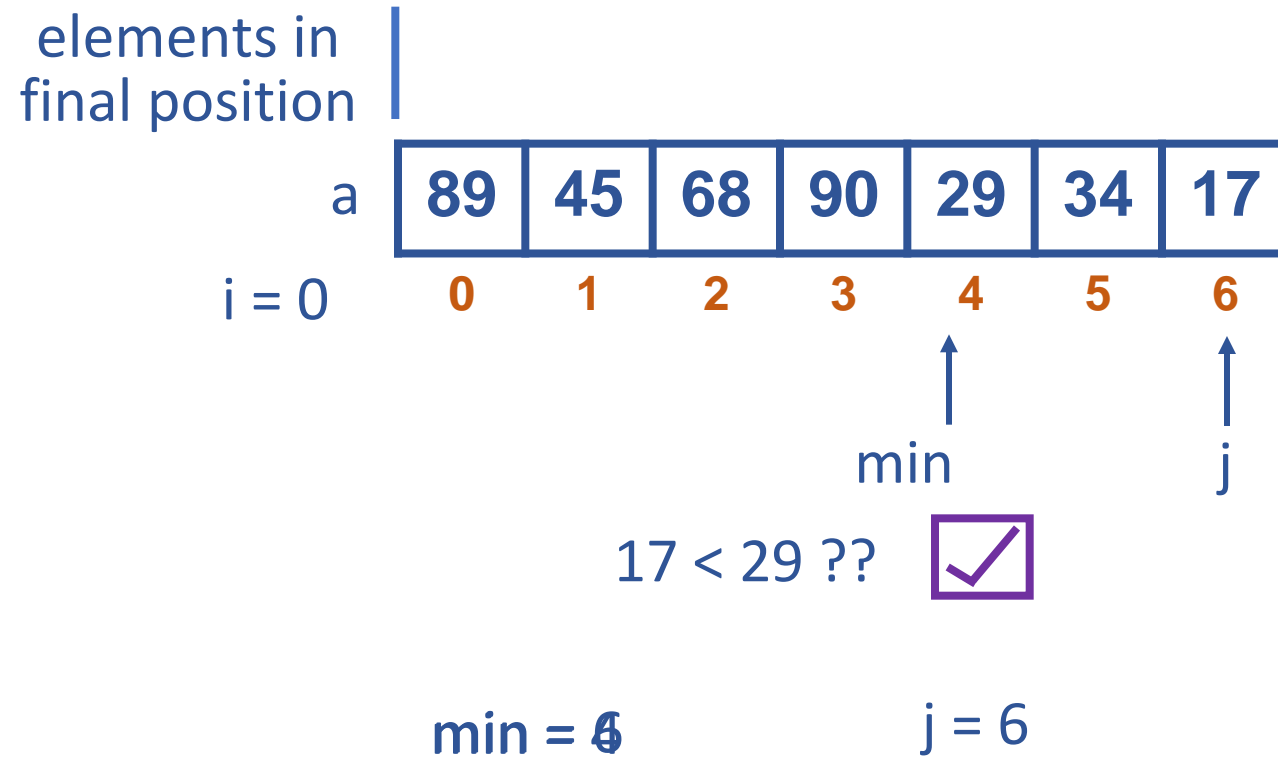


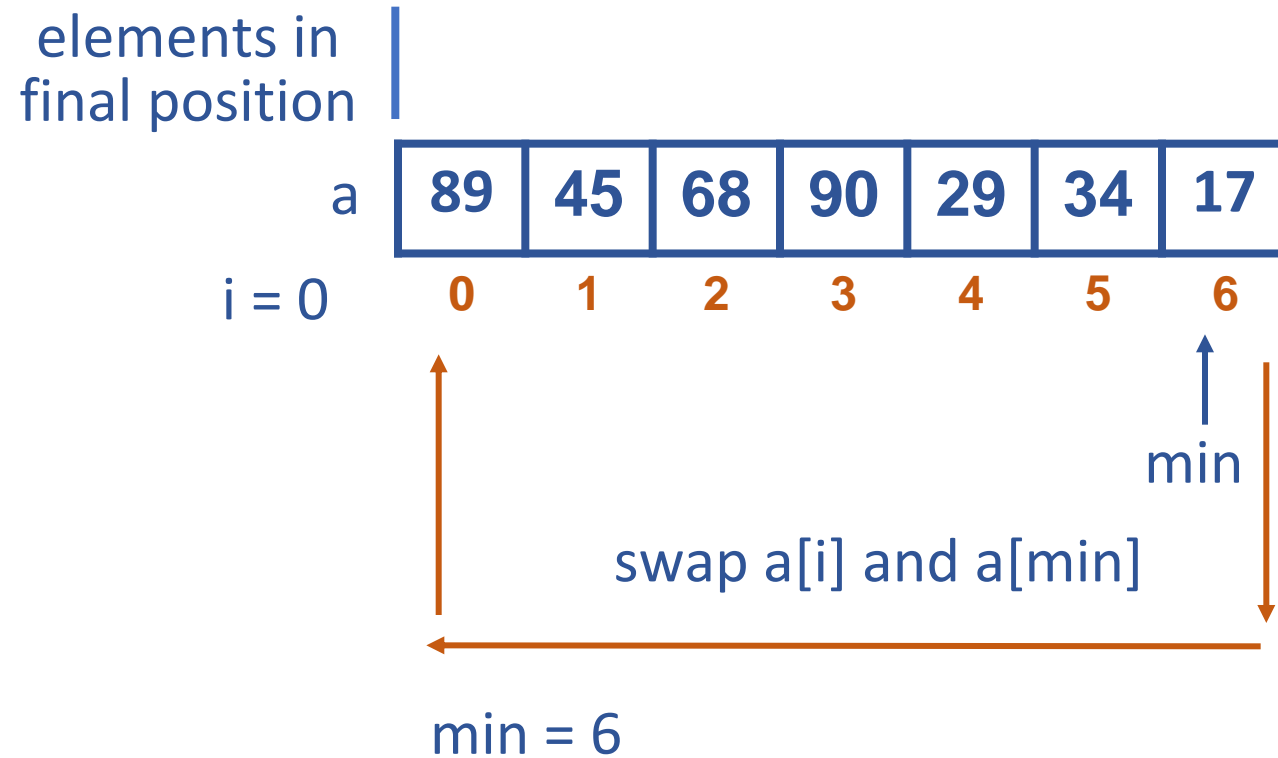








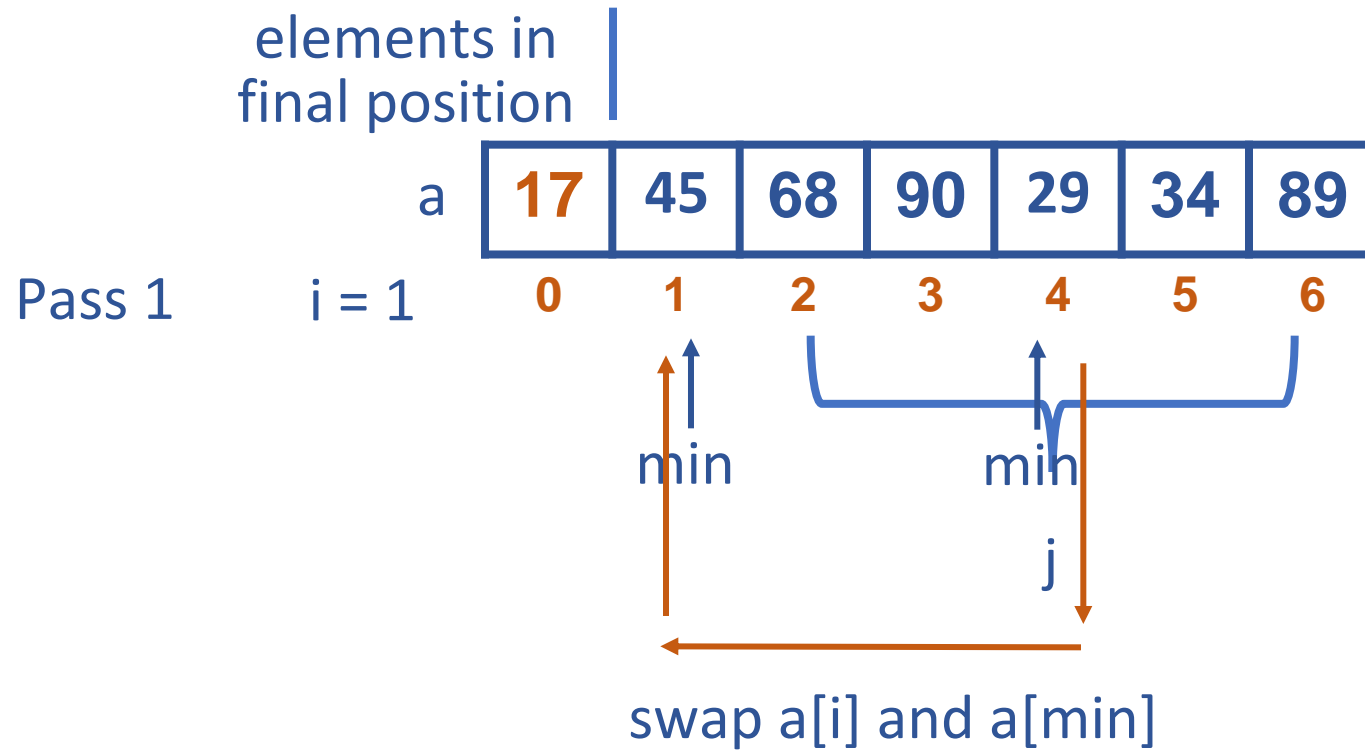




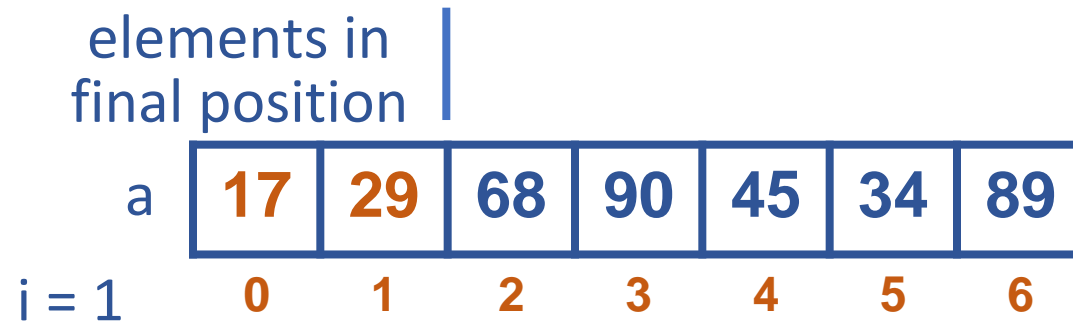


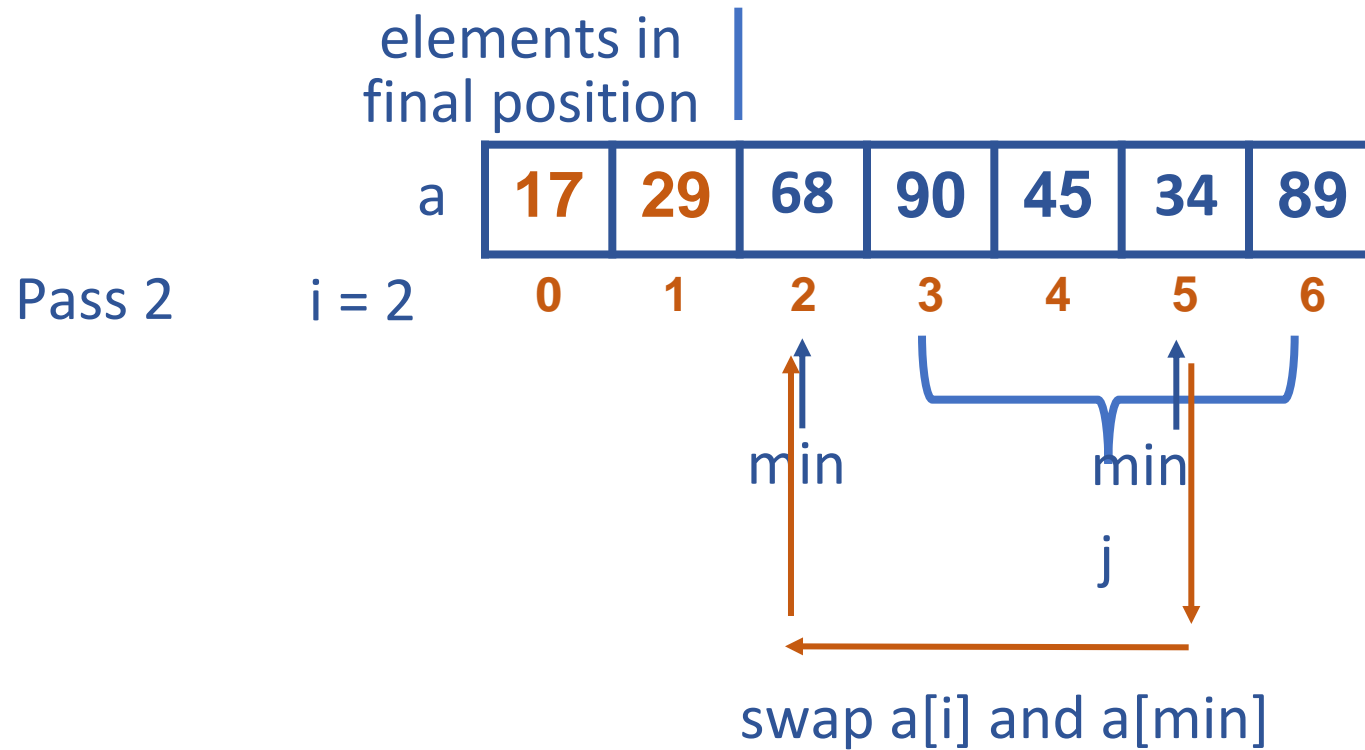
elements in  
final position |

a	17	45	68	90	29	34	89
i = 0	0	1	2	3	4	5	6

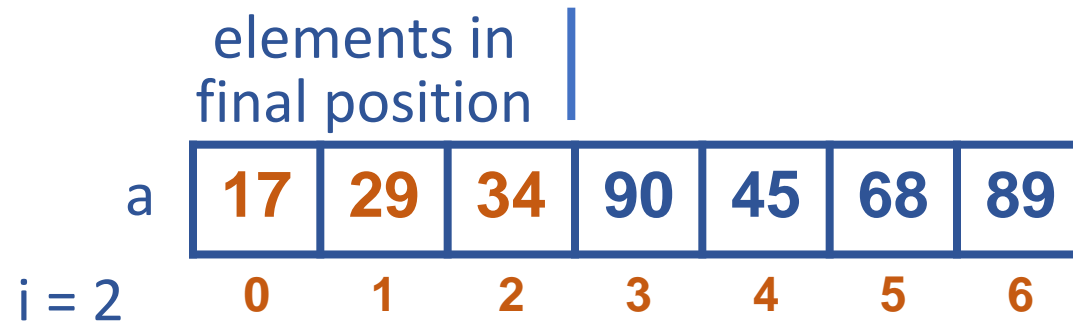


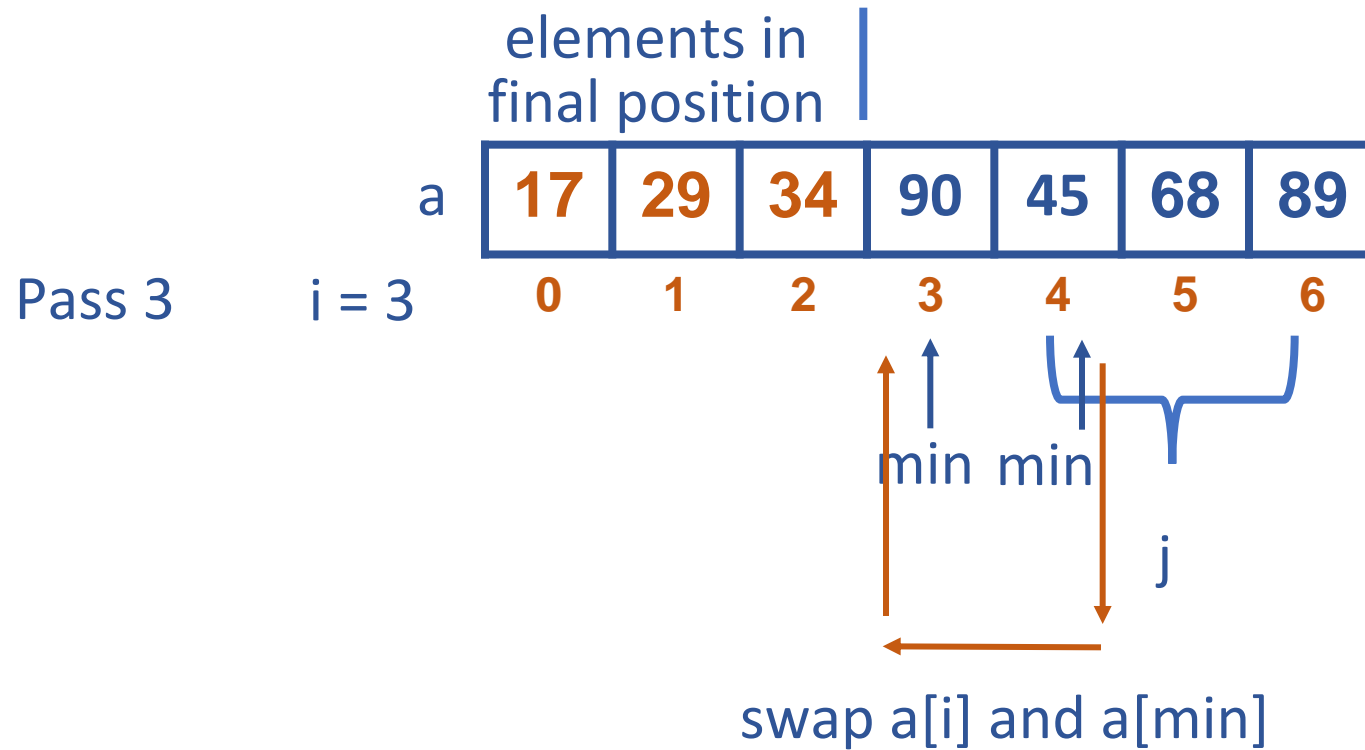
$\text{min} = 4$



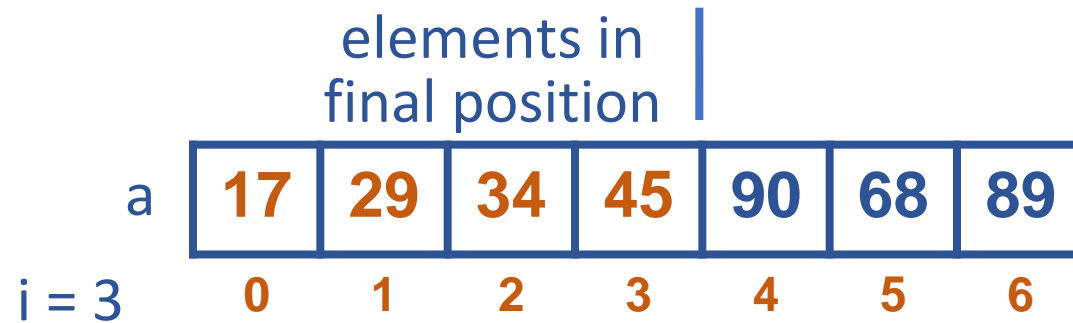


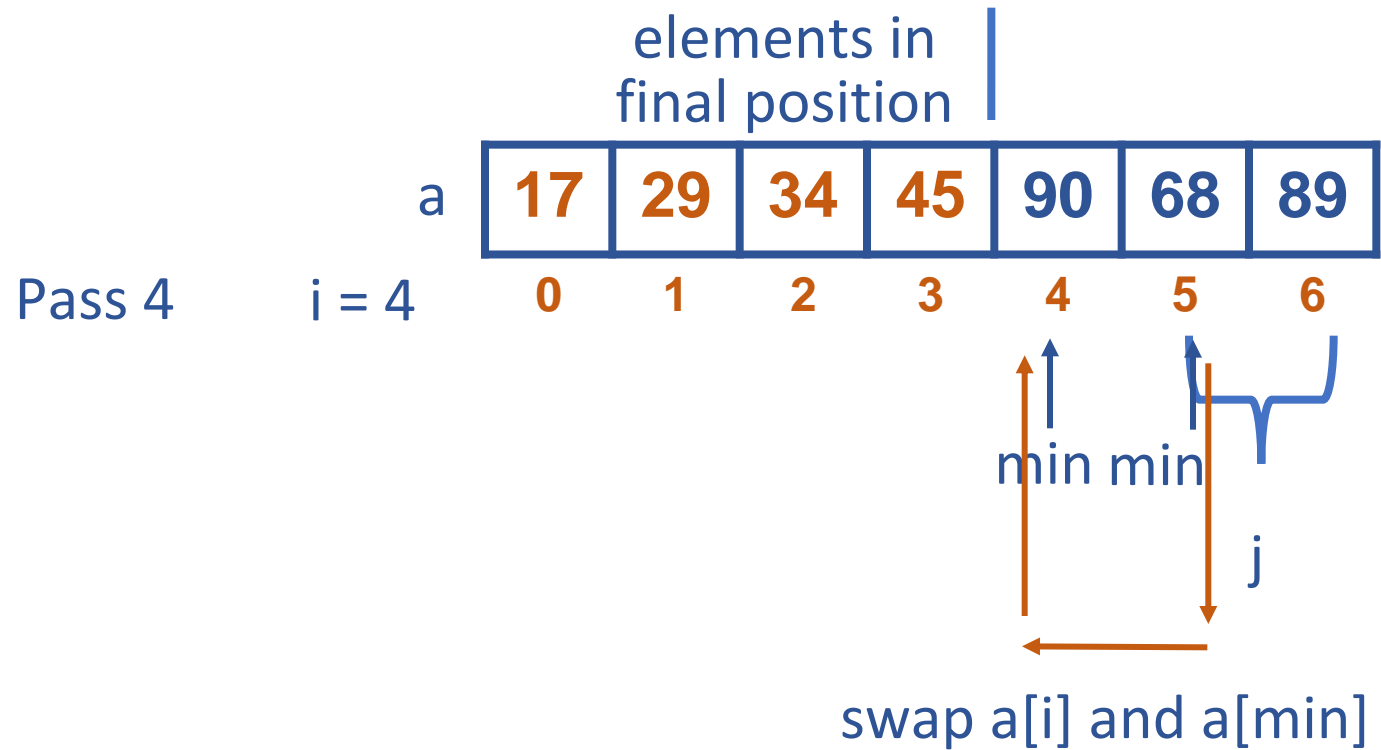
$\text{min} = 5$





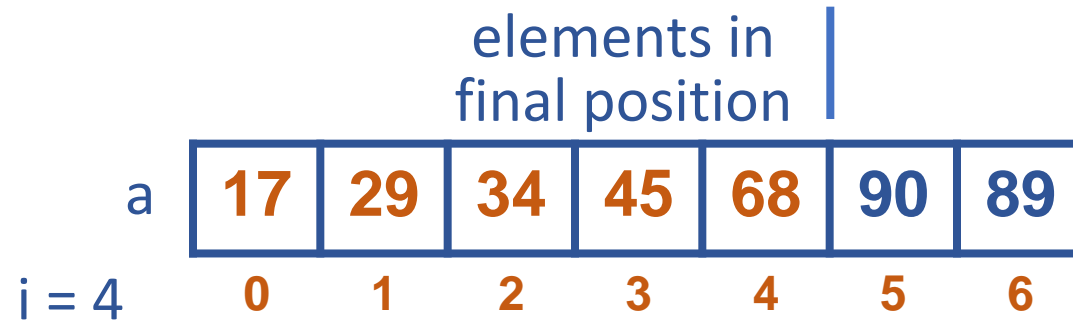
$\text{min} = 4$

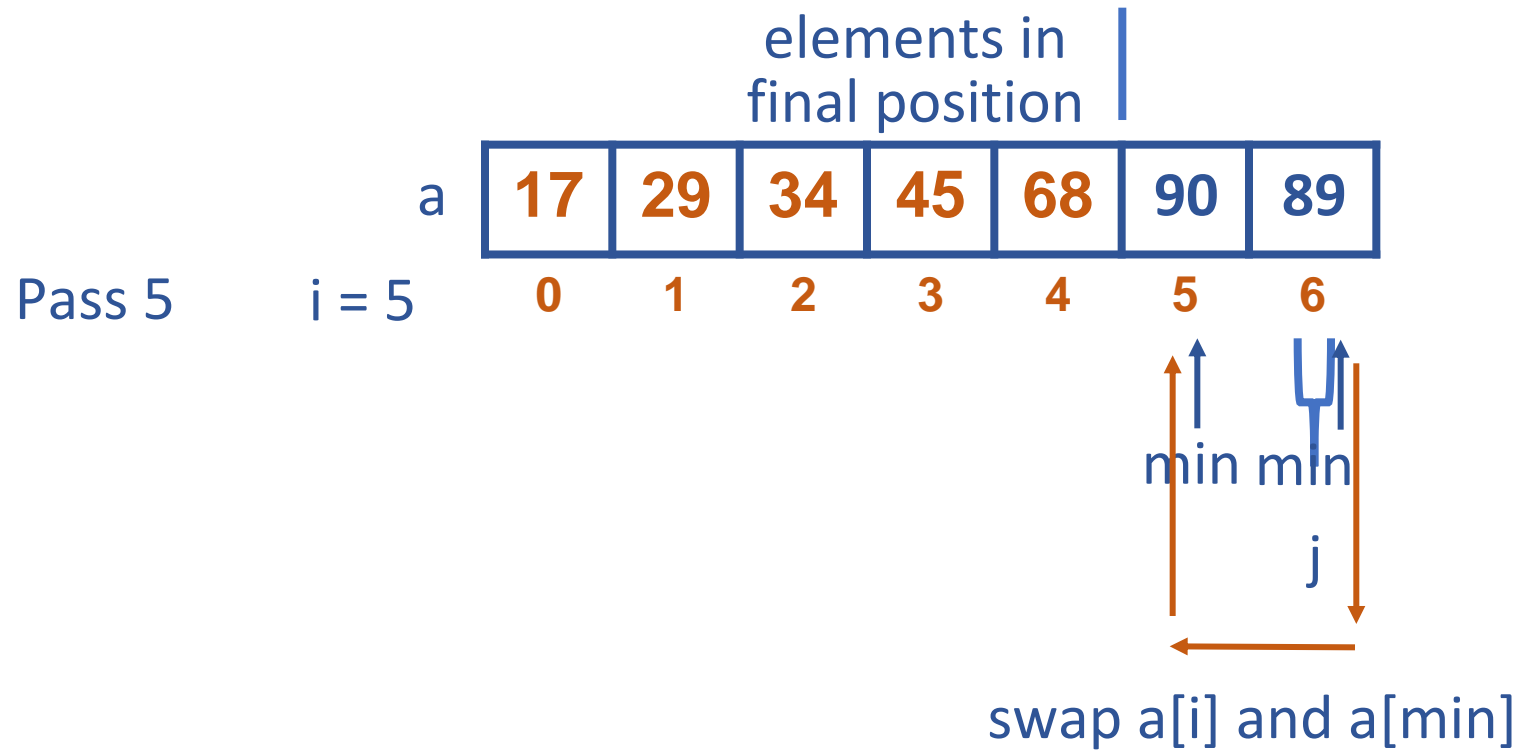




$\text{min} = 5$







$\text{min} = 6$

elements in  
final position |

a

<b>17</b>	<b>29</b>	<b>34</b>	<b>45</b>	<b>68</b>	<b>89</b>	<b>90</b>
0	1	2	3	4	5	6

i = 5

Original array

<b>89</b>	<b>45</b>	<b>68</b>	<b>90</b>	<b>29</b>	<b>34</b>	<b>17</b>
0	1	2	3	4	5	6

Array after sorting

<b>17</b>	<b>29</b>	<b>34</b>	<b>45</b>	<b>68</b>	<b>89</b>	<b>90</b>
0	1	2	3	4	5	6

ALGORITHM SelectionSort( $A[0 \dots n-1]$ )

//Sorts a given array by selection sort

//Input: An array  $A[0 \dots n-1]$  of orderable elements

//Output: Array  $A[0 \dots n-1]$  sorted in ascending order

for  $i \leftarrow 0$  to  $n-2$  do

$\text{min} \leftarrow i$

    for  $j \leftarrow i+1$  to  $n-1$  do

        if  $A[j] < A[\text{min}]$   $\text{min} \leftarrow j$

    swap  $A[i]$  and  $A[\text{min}]$

### Selection Sort Analysis

$$C(n) = \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1 = \sum_{i=0}^{n-2} (n-1-i) = \frac{(n-1)n}{2}$$

Selection Sort is a  $\Theta(n^2)$  algorithm



# THANK YOU

---

**Shylaja S S**

Department of Computer Science  
& Engineering

**[shylaja.sharath@pes.edu](mailto:shylaja.sharath@pes.edu)**