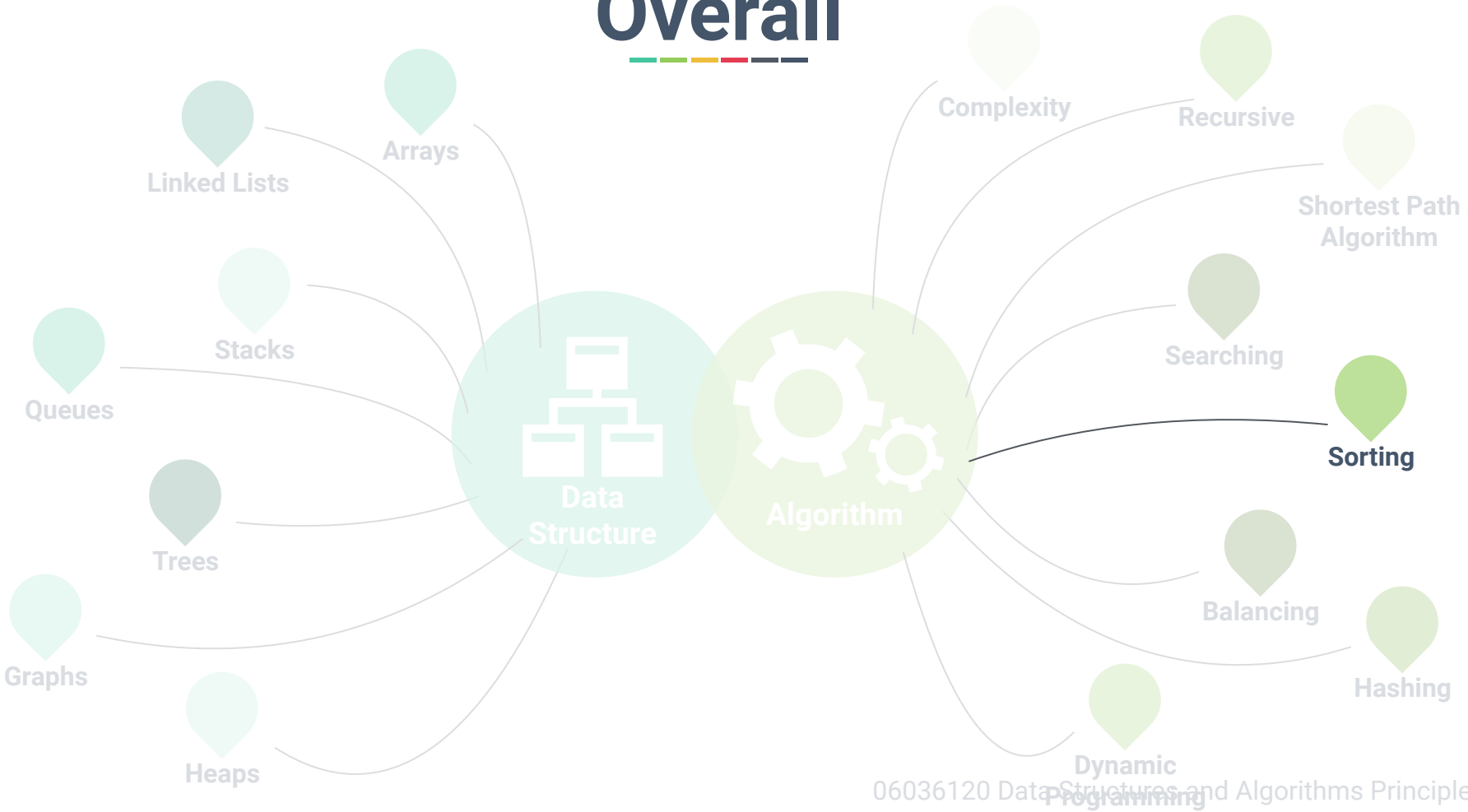
Chapter 10: Sorting

Dr. Sirasit Lochanachit

Overall

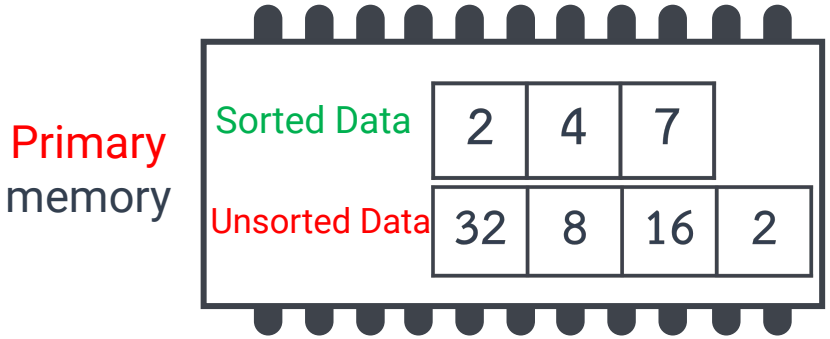

Linked Lists
Arrays
Stacks
Queues
Trees
Graphs
Heaps
Data Structure
Algorithm
Complexity
Recursive
Shortest Path Algorithm
Searching
Sorting
Balancing
Hashing
Dynamic Programming

Outline

Sorting

- Bubble Sort
- Selection Sort
- Insertion Sort

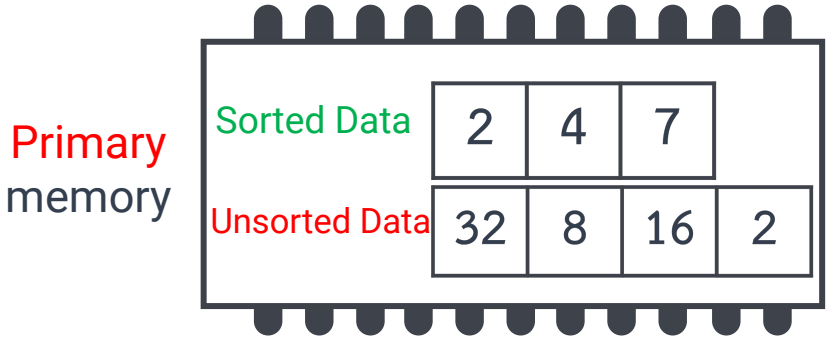
What is Sorting?



Sorting is the process of placing elements from a collection in some kind of order.

For instance, a list of words could be sorted by alphabet from a-z or z-a.

Sorting Operations



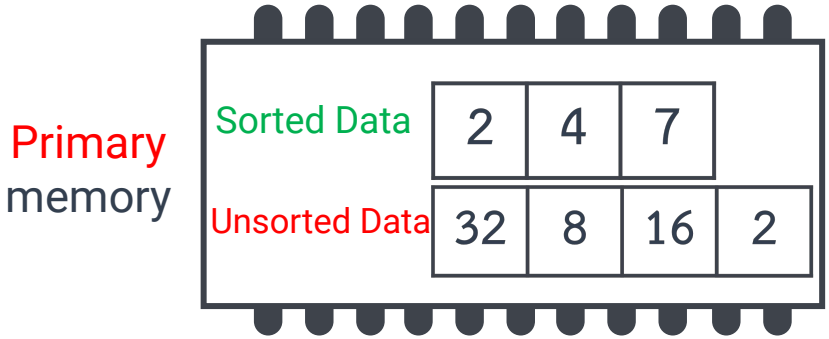
Generally, sorting has two operations:

1. **Compare** between two values to determine which is smaller (or larger).
   - The **total number of comparisons** is crucial to measure sorting efficiency.

Sorting Operations

Generally, sorting has two operations:

2. **Exchange** two values.
   - Exchange is a costly operation.
   - The **total number of exchanges** is also important for evaluating efficiency of the algorithm.



temp = alist[i]

alist[i] = alist[j]

alist[j] = temp

Python Built-in Sort Method

```
data = [10,1,3,4,9,2]
data.sort()
data
```

```
[1, 2, 3, 4, 9, 10]
```

```
[10]
data = [10,1,3,4,9,2]
sorted(data)
```

```
[1, 2, 3, 4, 9, 10]
```

```
data
```

```
[10, 1, 3, 4, 9, 2]
```

Bubble Sort

78	56	32	45	8	23	19

Bubble sort is a sorting algorithm which makes multiple iterations through a given list of unsorted elements.

- It compares each pair of adjacent elements and exchanges those that are out of order.

Ref : https://www.faceprep.in/bubble-sort-in-c/

Bubble Sort

78	56	32	45	8	23	19

- If there are *n* items in the list, then there are *n* - 1 pairs that needs to be compared on the first round.
- At the start of the 2nd round, there are *n* - 1 items left to sort which means there will be *n* - 2 pairs.
- Therefore, the total number of rounds is *n* - 1.

Great example: https://youtu.be/lyZQPjUT5B4?t=54

Bubble Sort Example

Round 1

Right-to-left

n - 1 pairs

78	56	32	45	8	23	19	Exchange
78	56	32	45	8	19	23	No Exchange
78	56	32	45	8	19	23	Exchange
78	56	32	8	45	19	23	Exchange
78	56	8	32	45	19	23	Exchange
78	8	56	32	45	19	23	Exchange
8	78	56	32	45	19	23	

Bubble Sort Example

Round 2

Right-to-left

n - 2 pairs

8	78	56	32	45	19	23	No Exchange
8	78	56	32	45	19	23	Exchange
8	78	56	32	19	45	23	Exchange
8	78	56	19	32	45	23	Exchange
8	78	19	56	32	45	23	Exchange
8	19	78	56	32	45	23	

Bubble Sort Example

Round 3

Right-to-left

n - 3 pairs

8	19	78	56	32	45	23	Exchange
8	19	78	56	32	23	45	Exchange
8	19	78	56	23	32	45	Exchange
8	19	78	23	56	32	45	Exchange

8	19	23	78	56	32	45

Bubble Sort Example

Round 4

Right-to-left

n - 4 pairs

8	19	23	78	56	32	45	No Exchange
8	19	23	78	56	32	45	Exchange
8	19	23	78	32	56	45	Exchange

8	19	23	32	78	56	45

Bubble Sort Example

Round 5

Right-to-left

n - 5 pairs

8	19	23	32	78	56	45	Exchange
8	19	23	32	78	45	56	Exchange
8	19	23	32	45	78	56	

Bubble Sort Example

Round 6

Right-to-left

n - 6 pairs

8	19	23	32	45	78	56

Exchange

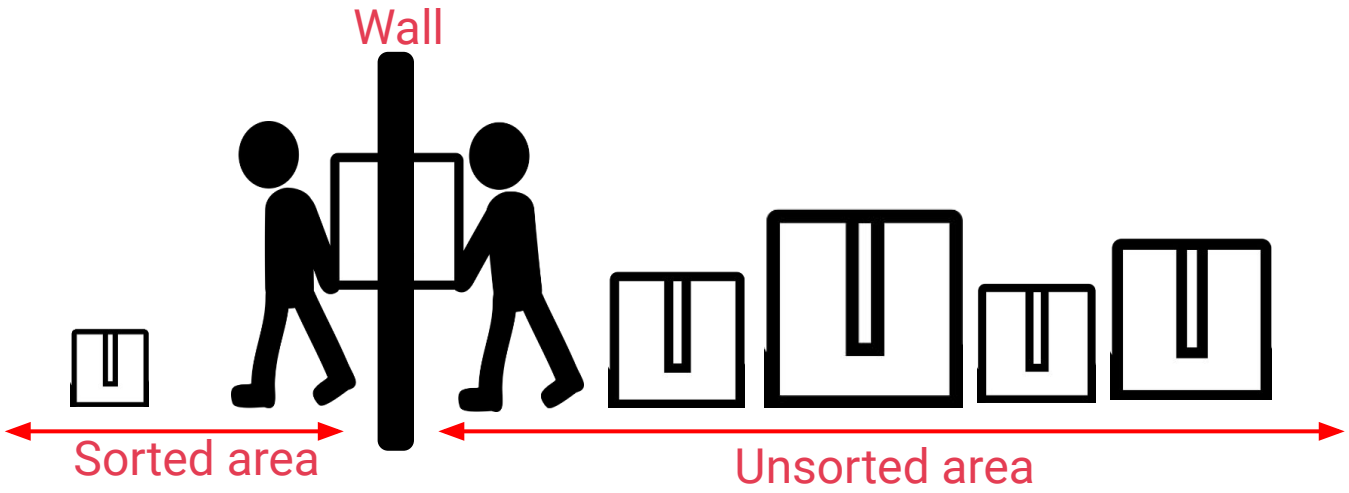
8	19	23	32	45	56	78

Bubble Sort Performance

Round	Number of Comparison
1	n - 1
2	n - 2
3	n - 3
...	...
n - 1	1

```
for round in range(0, n-1):
    for i in range(n-1, round, -1):
        if data[i-1] > data[i]:
            swap(data[i-1], data[i])
```

Selection Sort

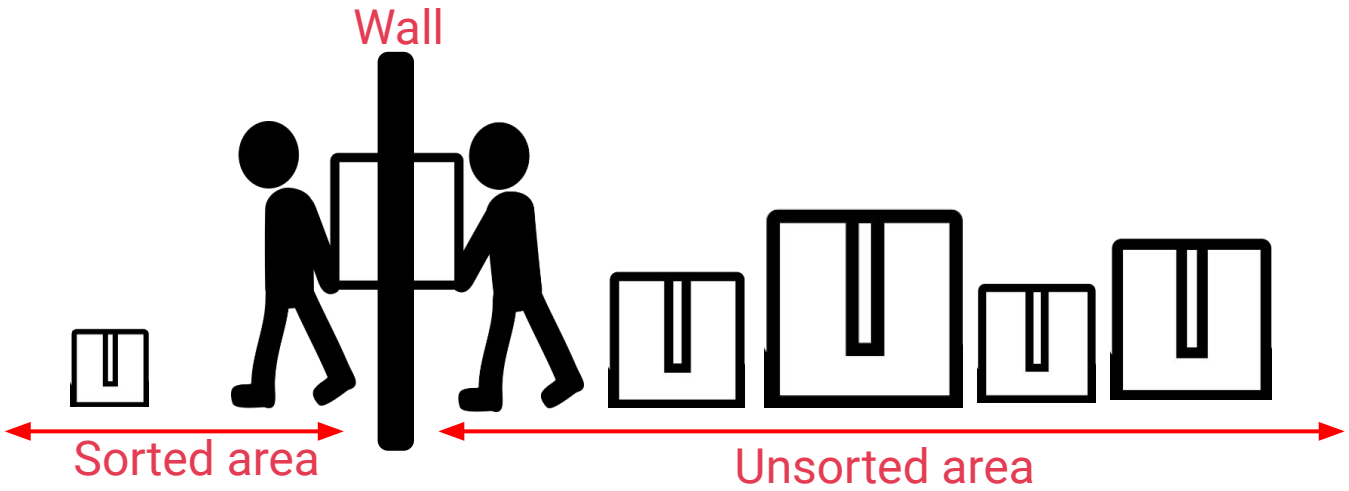


How to selection sort:

Given a list of unsorted data, it selects the smallest value and place it in a sorted list.

These steps are then repeated until all of the data are sorted.

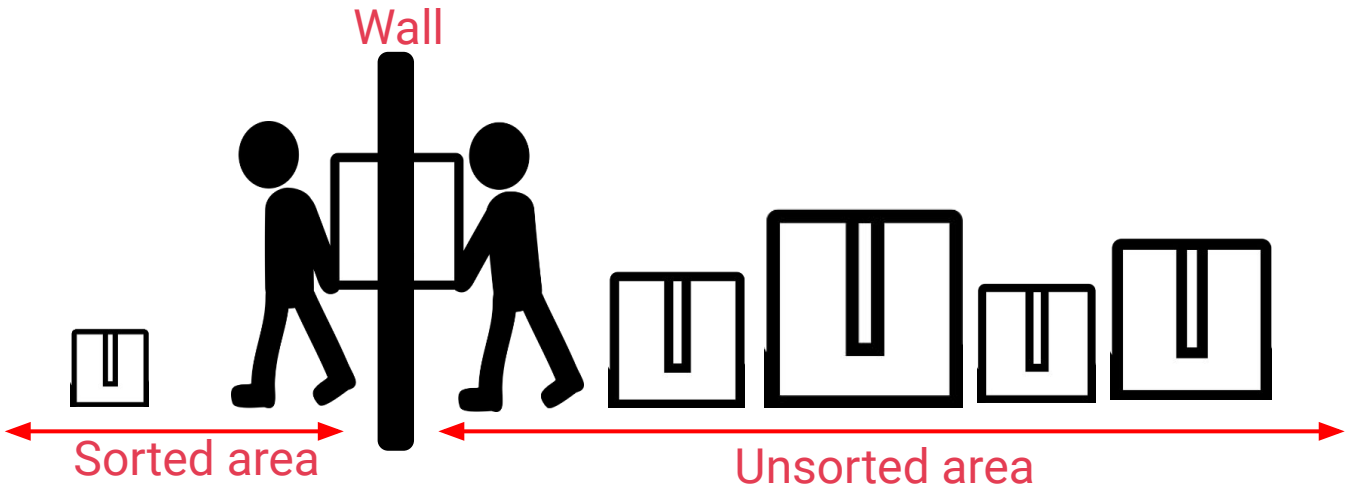
Selection Sort



In other words, the list is divided into two sub-lists, sorted and unsorted, which are divided by an imaginary wall.

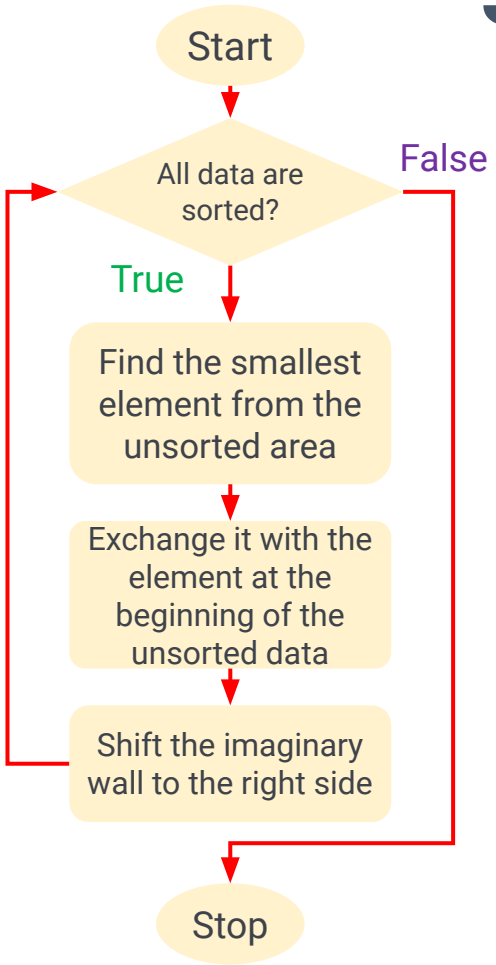
The smallest element from the unsorted sub-list are selected and exchange it with the element at the beginning of the unsorted data.

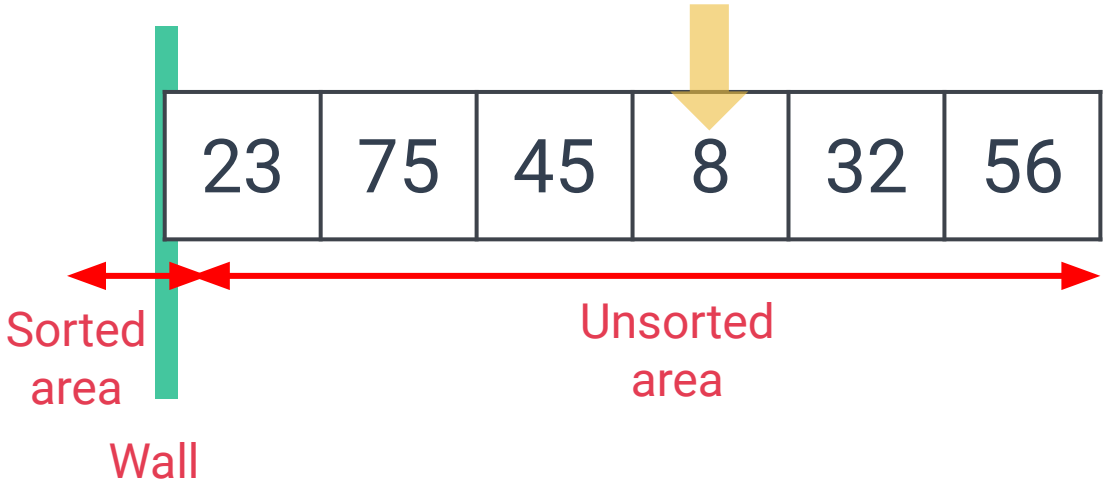
Selection Sort Real-life Demo



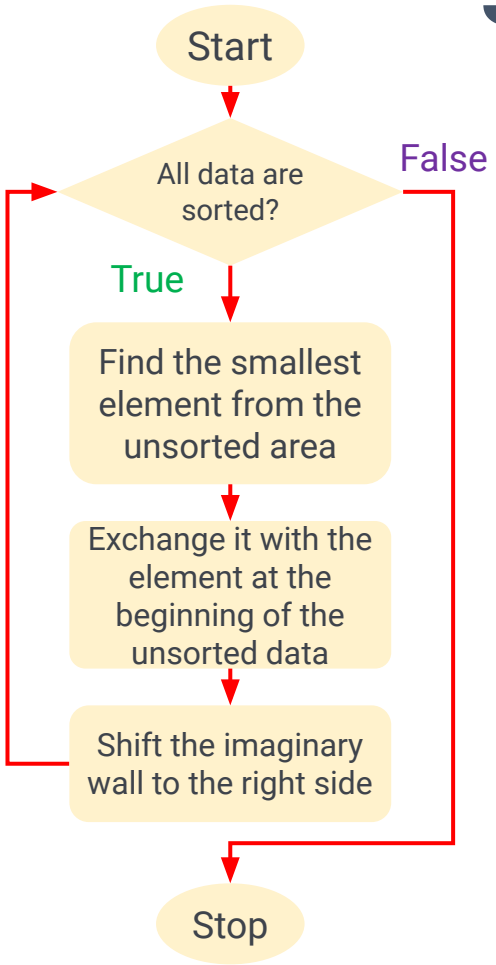
Select-sort with Gypsy folk dance

Selection Sort Example

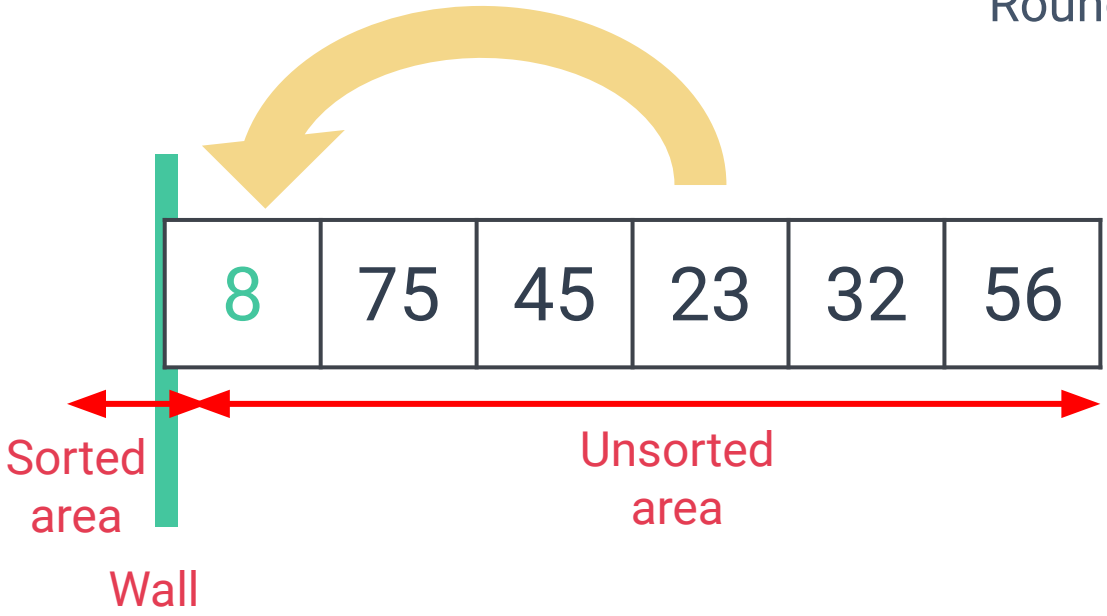




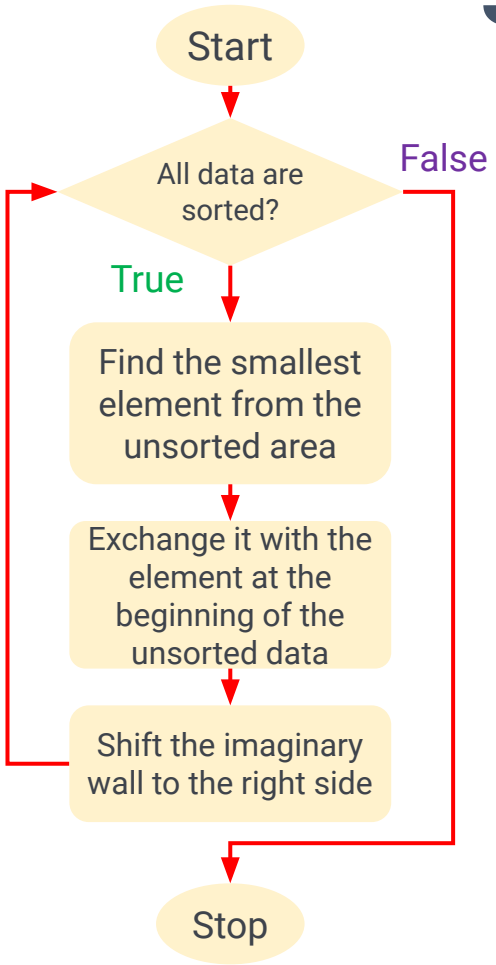
Selection Sort Example

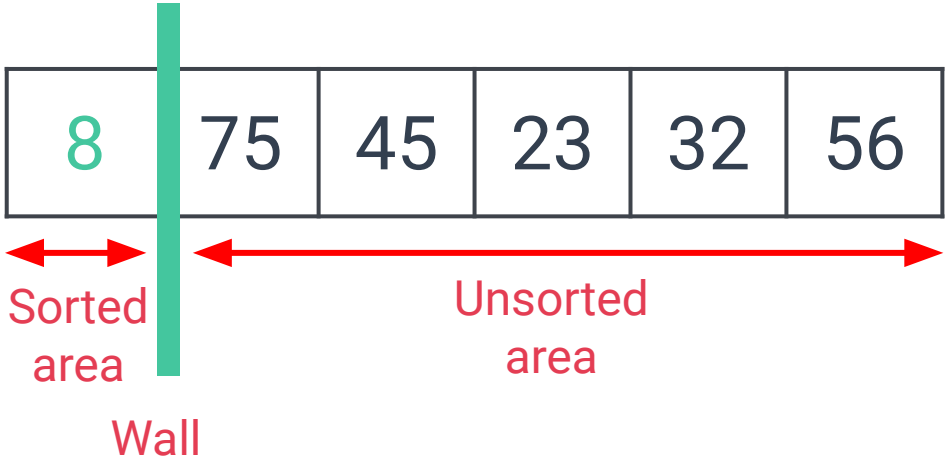


Round 1

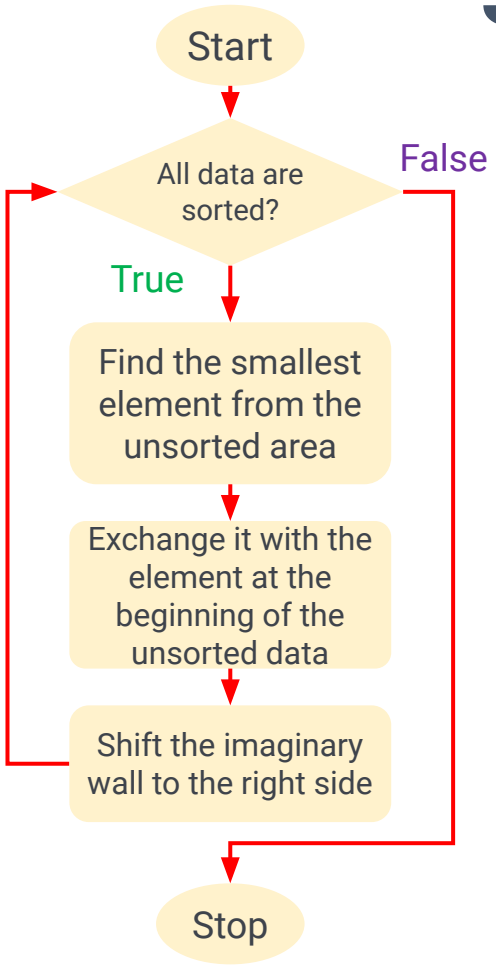


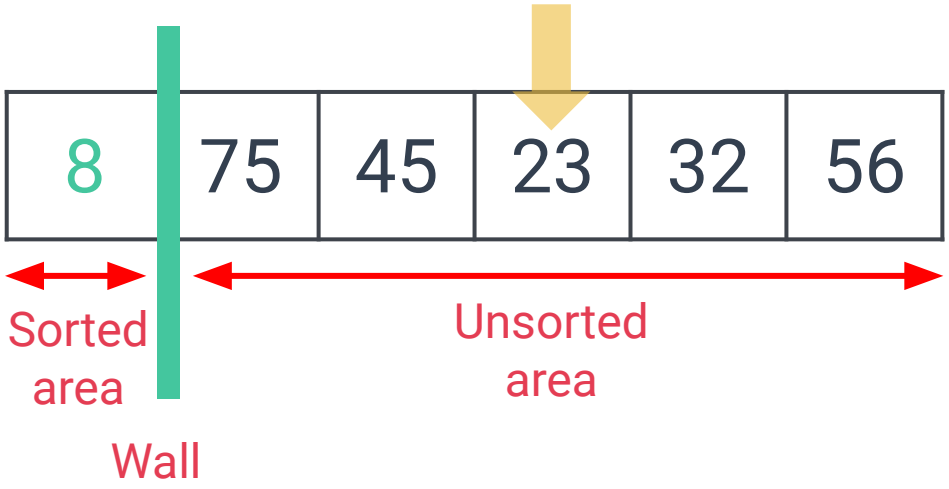
Selection Sort Example



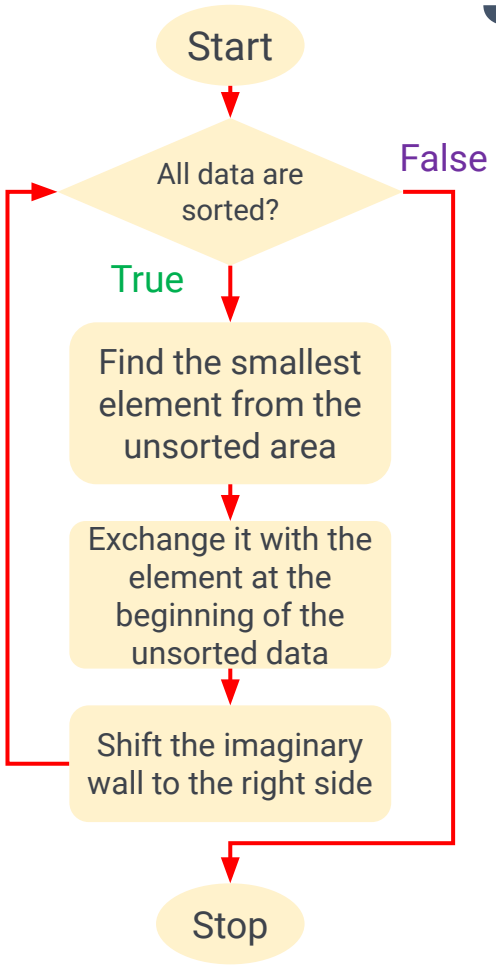


Selection Sort Example

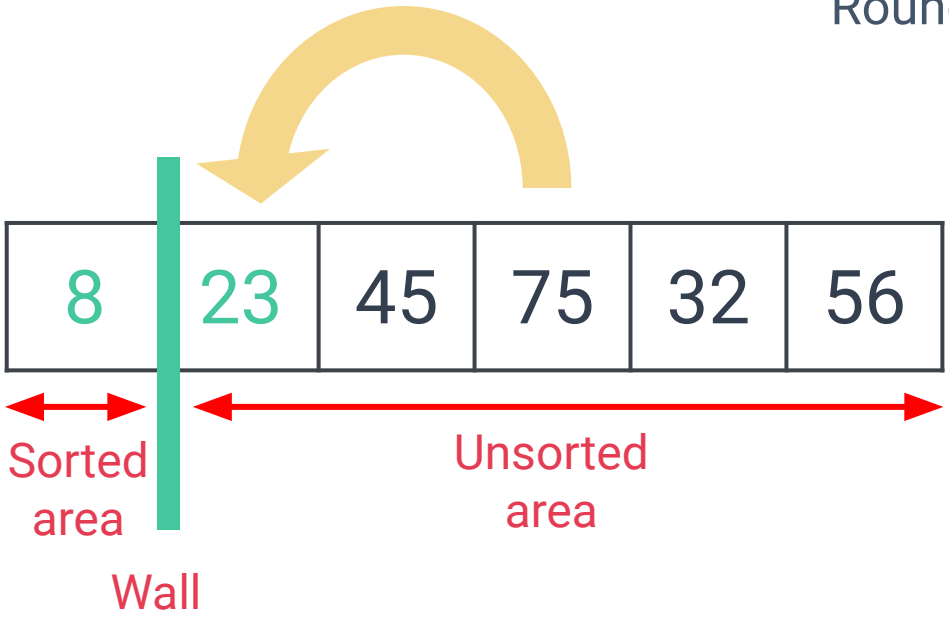




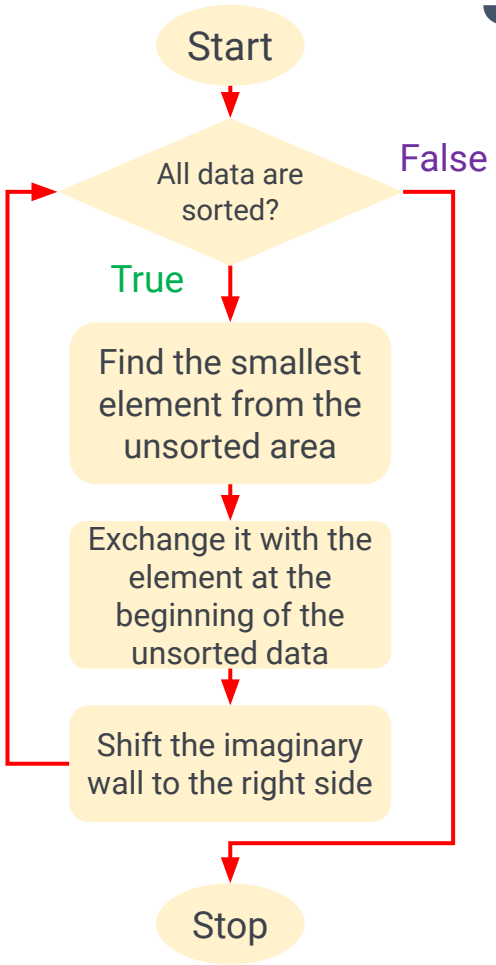
Selection Sort Example

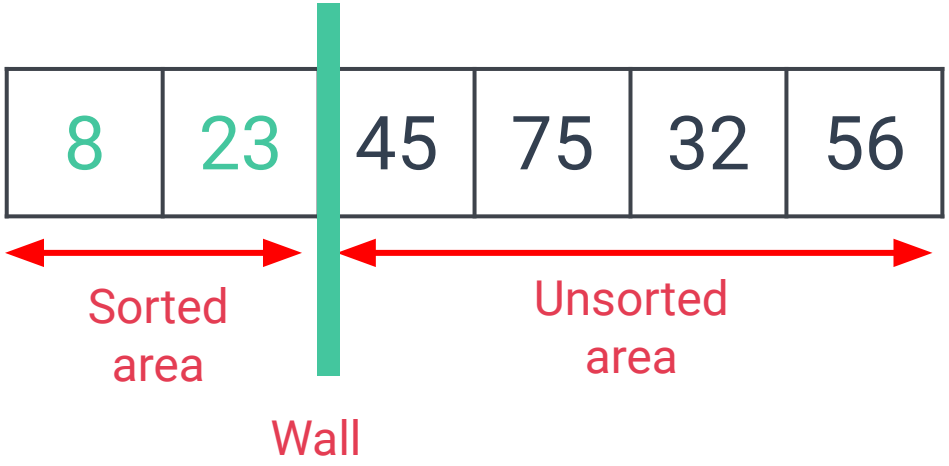


Round 2

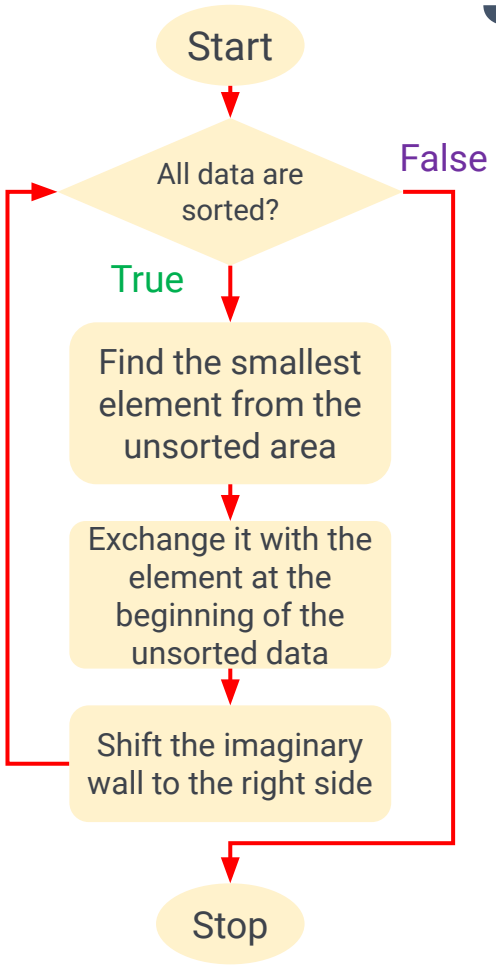


Selection Sort Example

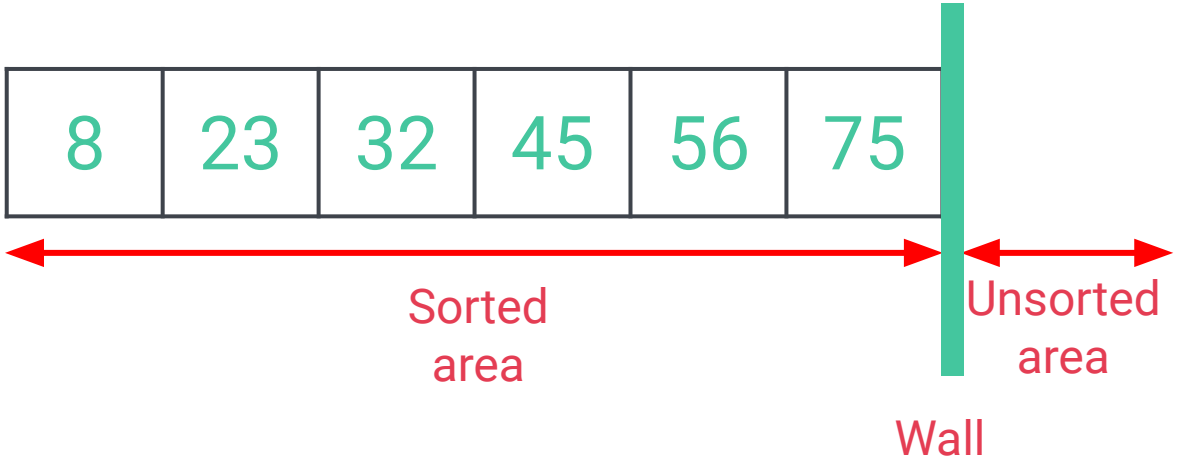




Selection Sort Example



After Round 5



Selection Sort Performance

Round	Number of Comparison
1	n - 1
2	n - 2
3	n - 3
...	...
n - 1	1

```
for ??:
    minIndex = ?
    for ??:
        if ??
            Update minIndex
    Swap between the smallest item and
the first item in unsorted area.
```

Insertion Sort

Similar to selection sort, a list is divided to two parts: sorted and unsorted.

In each round, the first element of the unsorted sublist is transferred to the sorted sublist by inserting it at the appropriate place.

A real example is sorting cards by card players. As they pick up each card, they insert it into the proper sequence in their hand.

Insertion Sort

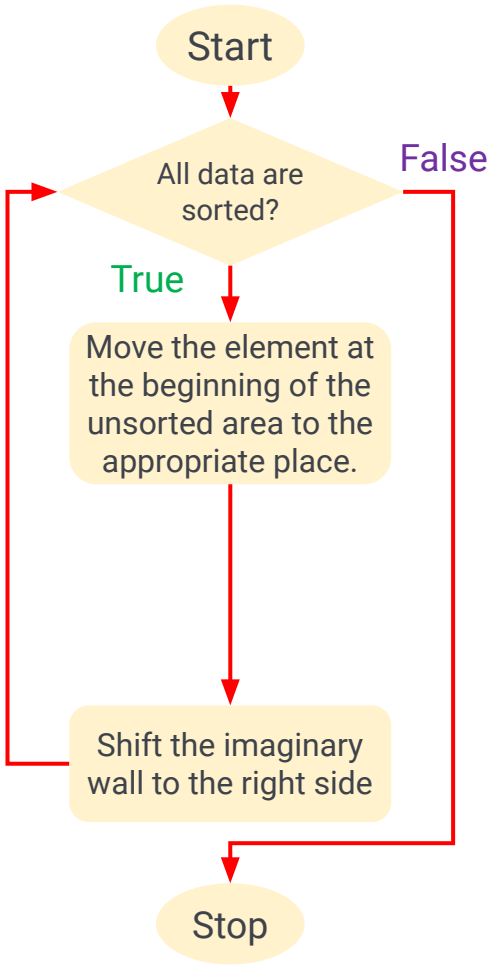
Where to insert?

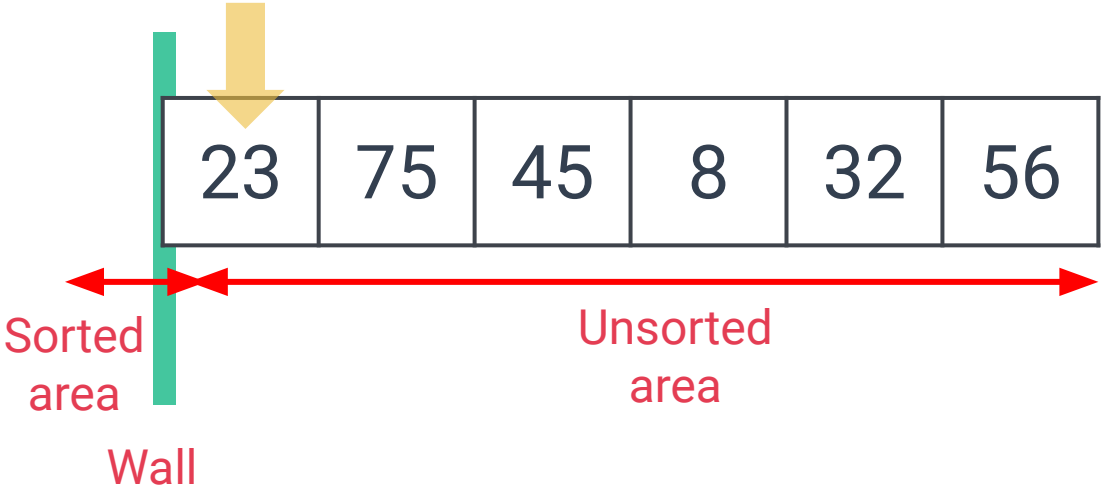
- The selected item is checked against items in the sorted sublist.
- The items in the sorted sublist that have greater value are shifted to the right.
- When reach a smaller item or the start of the sublist, the selected item can be inserted.

Insertion Sort Real-life Demo

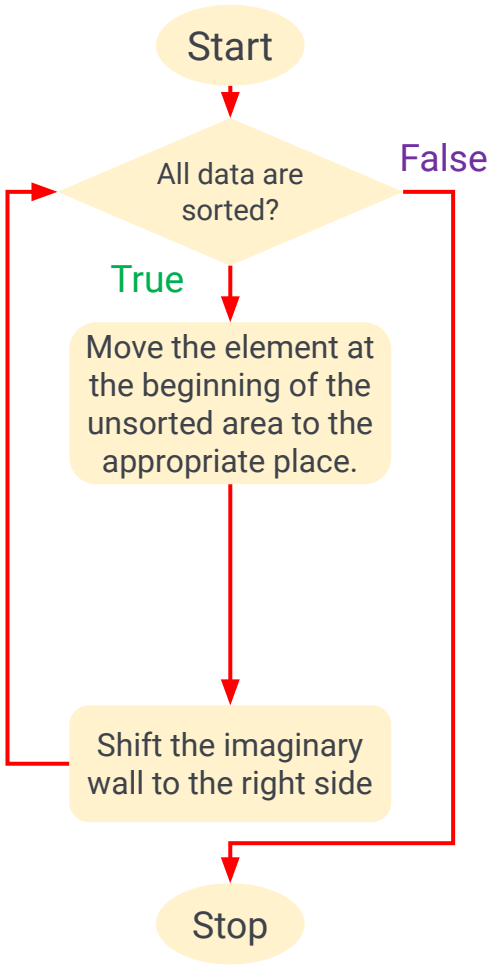
Insert-sort with Romanian folk dance

Insertion Sort Example



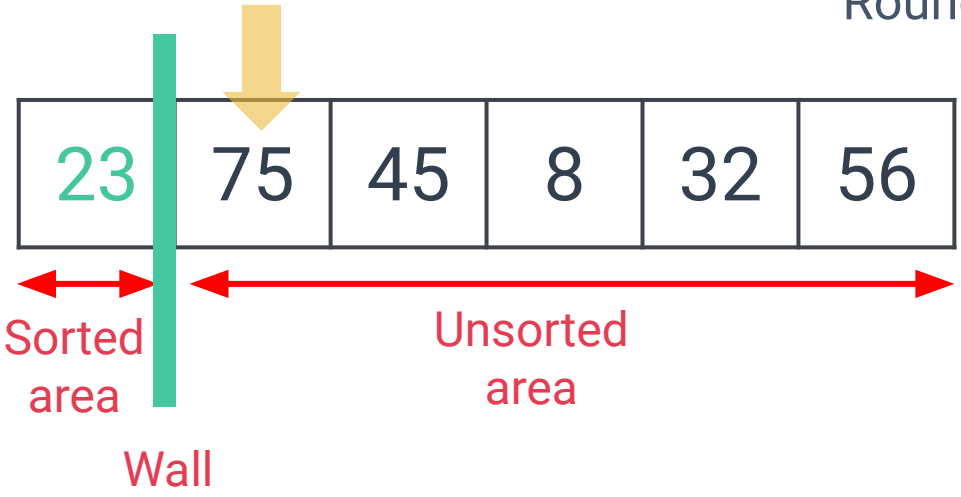


Insertion Sort Example

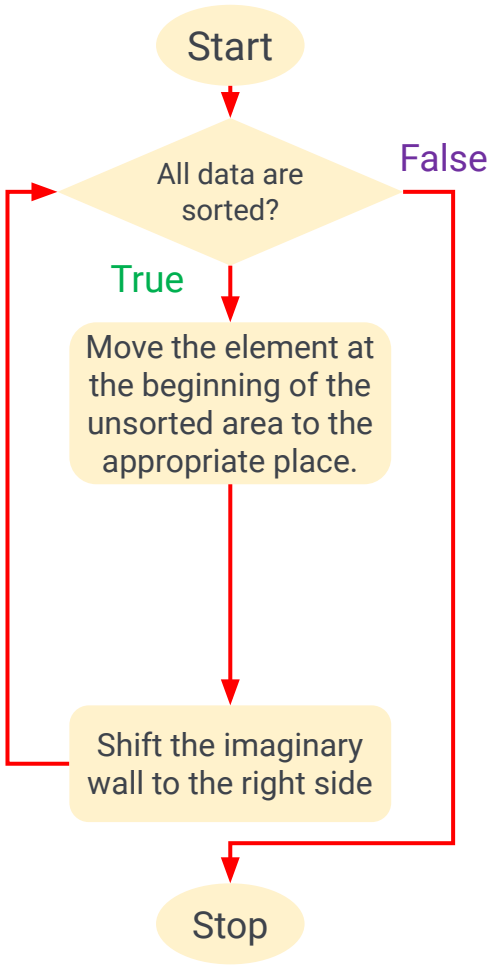


Inserted 23

Round 1

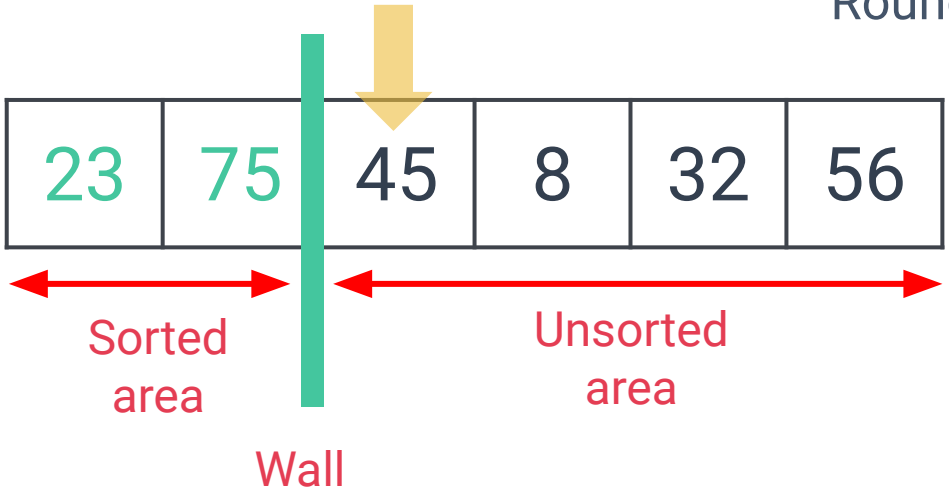


Insertion Sort Example

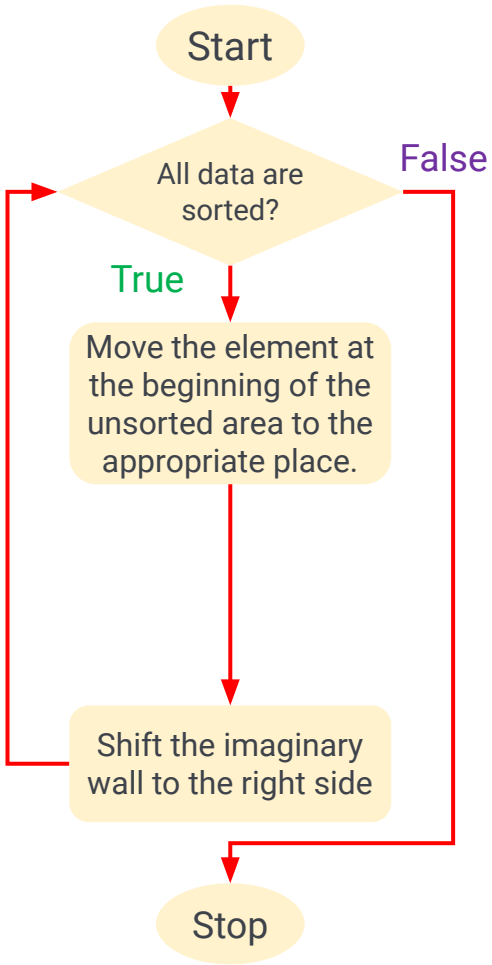


Inserted 75

Round 2

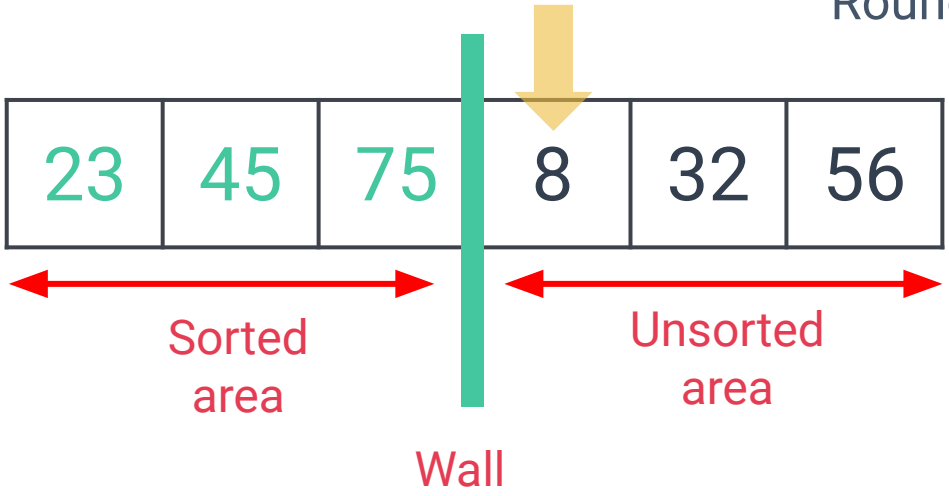


Insertion Sort Example

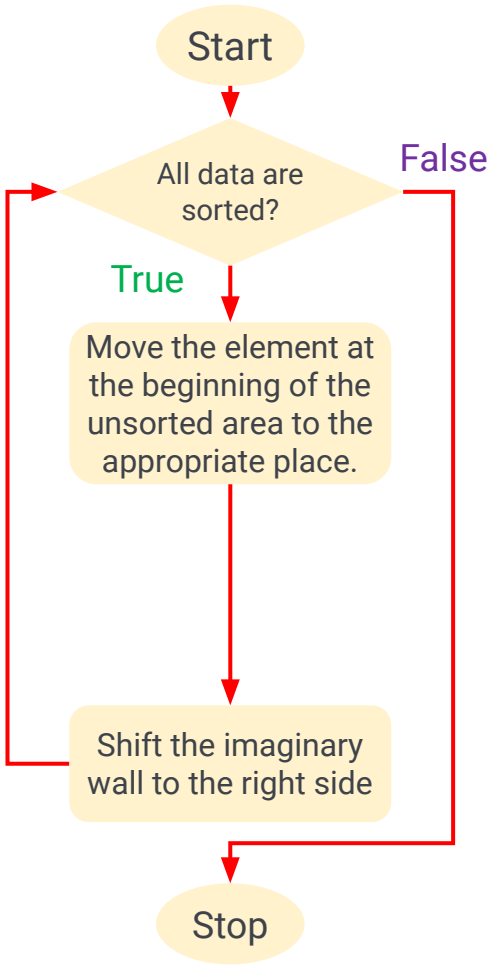


Inserted 45

Round 3

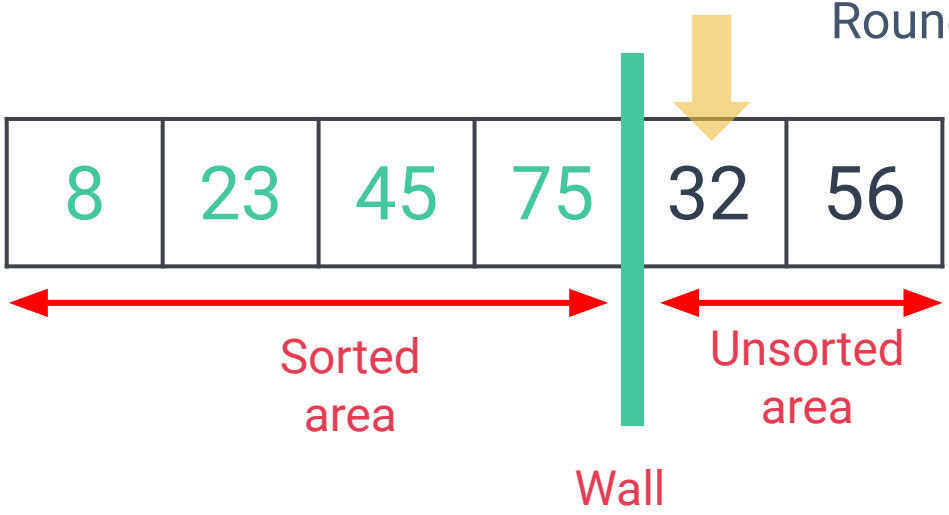


Insertion Sort Example

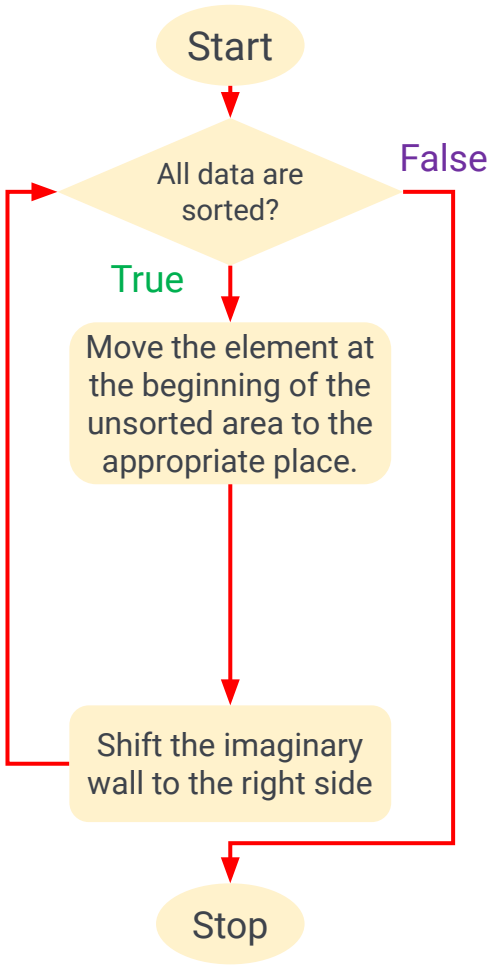


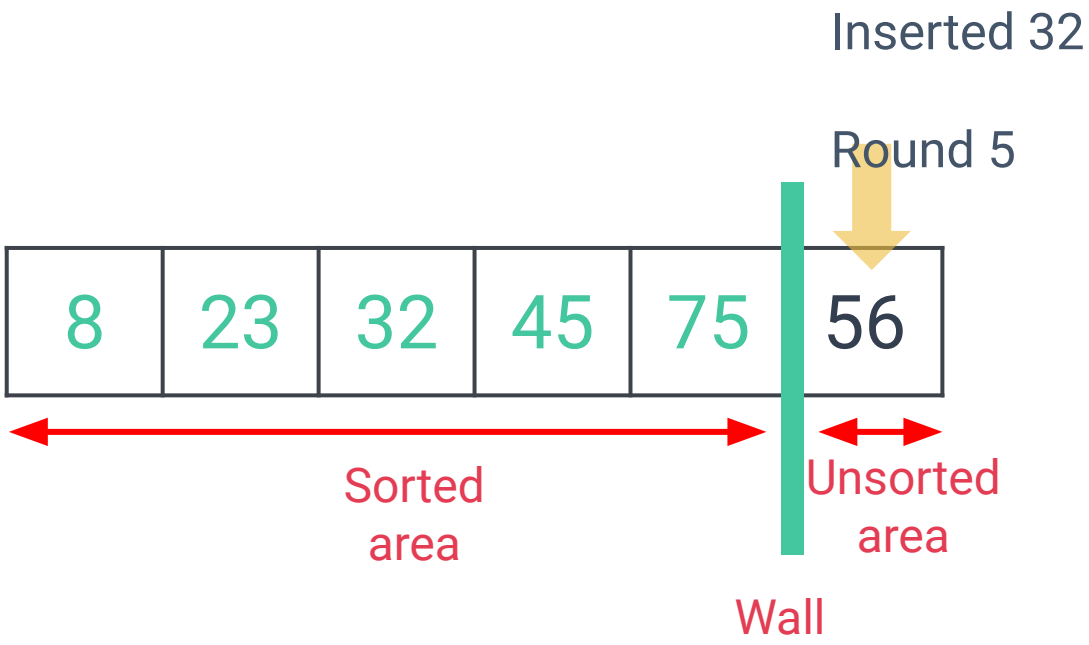
Inserted 8

Round 4

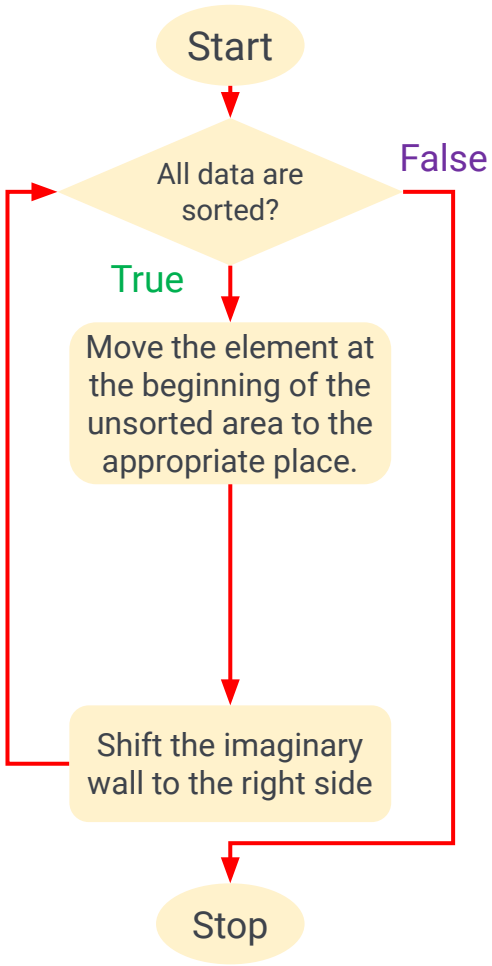


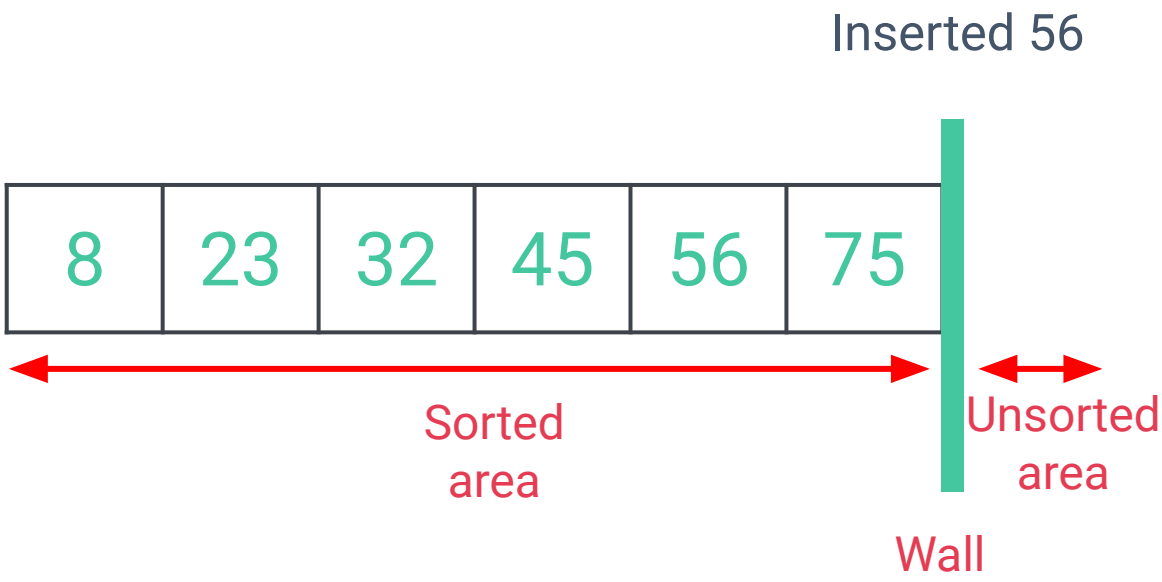
Insertion Sort Example





Insertion Sort Example





Insertion Sort Performance

Round	Number of Comparison
1	1
2	2
3	3
...	...
n - 1	n - 1

for ??:

currentValue = ?

position = ?

while ??:

Swap position if the item to insert is smaller than the item in sorted sublist.

Insert the selected item at the proper place.