Selection sort

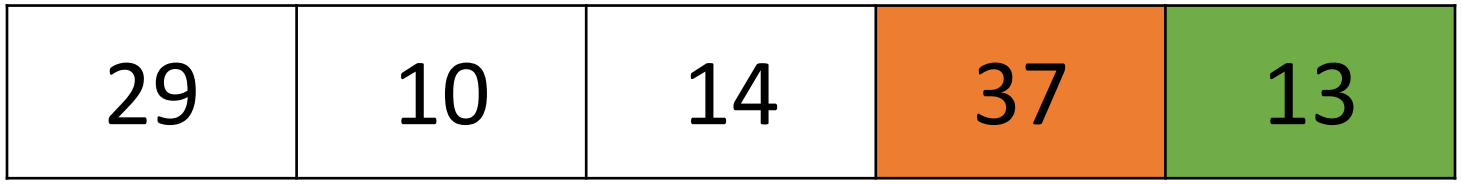
Selection Sort
Idea

- Given an array of n items:
  - Find the largest item x, in the range of $[0 \ldots n-1]$.
  - Swap x with the $(n-1)$-th item.
  - Reduce n by 1 and go to $step$ 1.

Selection Sort Example

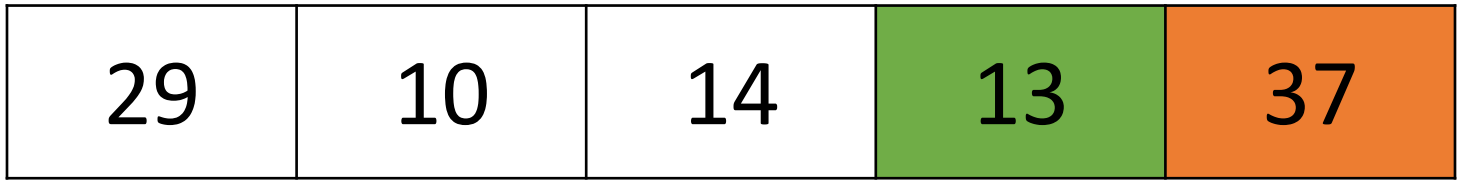
29	10	14	37	13

Selection Sort Example



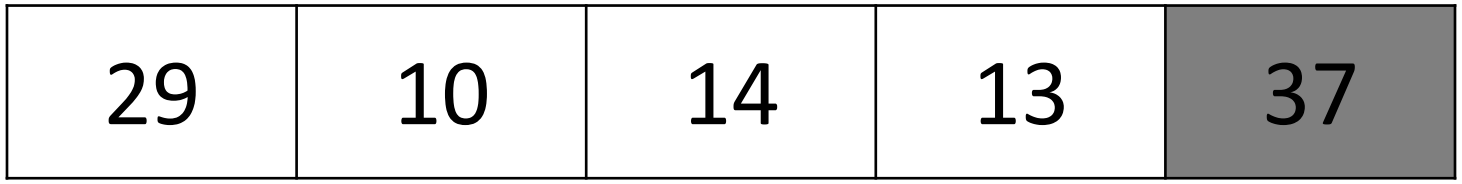
37 is the largest, swap it with the last element, i.e. 13.

Selection Sort Example

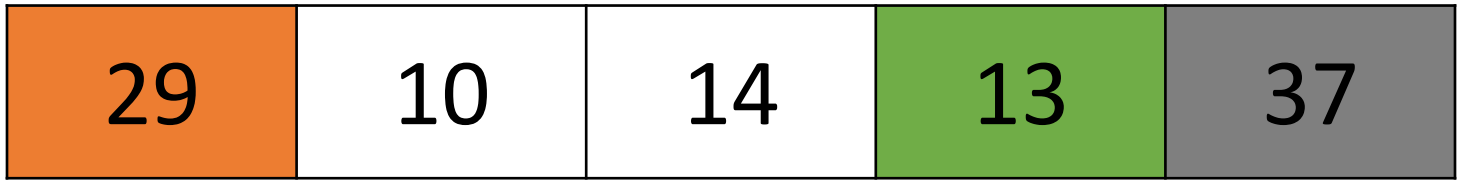


37 is the largest, swap it with the last element, i.e. 13.

Selection Sort Example

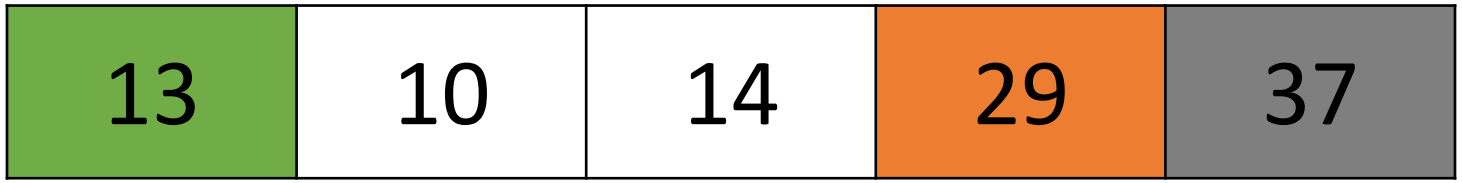


Selection Sort Example



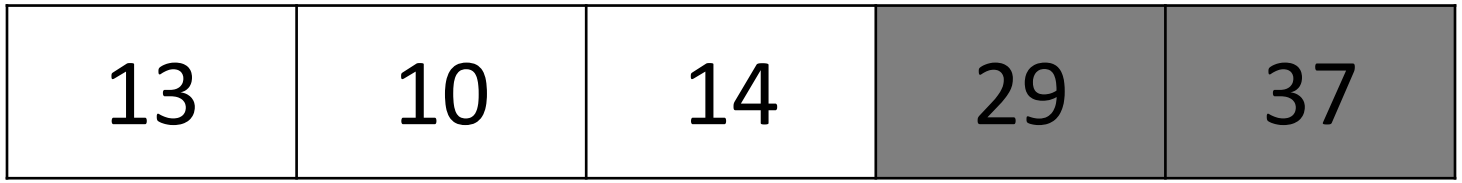
29 is the next largest, swap it with the appropriate element, i.e. again 13.

Selection Sort Example



29 is the next largest, swap it with the appropriate element, i.e. again 13.

Selection Sort Example



Selection Sort Example

14 is the next largest, swap it with the appropriate element, i.e. with himself.

Selection Sort Example

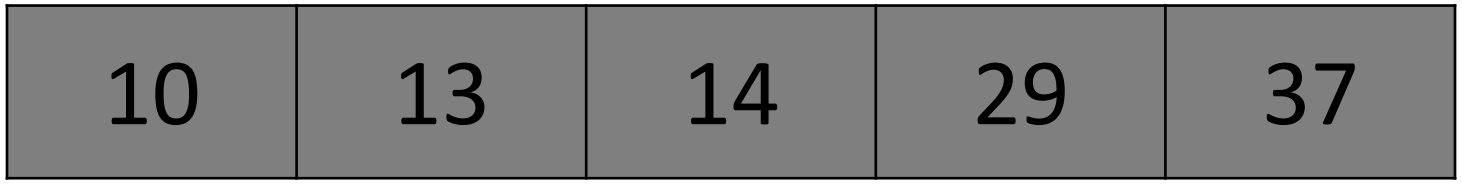
Selection Sort Example

13 is the next largest, swap it with the appropriate element, i.e. with 10.

Selection Sort Example

13 is the next largest, swap it with the appropriate element, i.e. with 10.

Selection Sort Example



Selection Sort Implementation

```
void selection_sort(vector<int>& arr)
{
    for (int i = 0; i < arr.size(); ++i)
    {
        int min_index = i;
        for (int j = i + 1; j < arr.size(); ++j)
        {
            if (arr[min_index] > arr[j])
            {
                min_index = j;
            }
        }
        swap(arr[min_index], arr[i]);
    }
}
```

Selection Sort Analysis

- Selection sort is performing $n - i - 1$ operations for each i to find maximum/minimum.

- So time complexity of this sort is $(n - 1) + (n - 2) + \ldots + 1 = O(n^2)$.