Selection Sort

Selection Sort

Selection Sort is a simple comparison-based sorting algorithm. It works by repeatedly finding the minimum element from the unsorted part of the list and moving it to the beginning. Here's a step-by-step explanation with an example:

Steps

1. **Initialize**: Start with the first element of the array.
2. **Find Minimum**: Find the smallest element in the unsorted portion of the array.
3. **Swap**: Swap the smallest element with the first unsorted element.
4. **Repeat**: Move the boundary of the unsorted part one element to the right.

Example

Let's sort the array `[64, 25, 12, 22, 11]` using Selection Sort.

1. **Initial Array**: `[64, 25, 12, 22, 11]`
2. **Find Minimum in [64, 25, 12, 22, 11]**: Minimum is 11
3. **Swap 64 and 11**: `[11, 25, 12, 22, 64]`
4. **Find Minimum in [25, 12, 22, 64]**: Minimum is 12
5. **Swap 25 and 12**: `[11, 12, 25, 22, 64]`
6. **Find Minimum in [25, 22, 64]**: Minimum is 22
7. **Swap 25 and 22**: `[11, 12, 22, 25, 64]`
8. **Find Minimum in [25, 64]**: Minimum is 25 (no swap needed)
9. **Final Sorted Array**: `[11, 12, 22, 25, 64]`

Code Implementation

C#

```
using System;
```

```
class SelectionSort
{
    public static void Sort(int[] arr)
    {
        int n = arr.Length;

        for (int i = 0; i < n - 1; i++)
        {
            int minIndex = i;
            for (int j = i + 1; j < n; j++)
            {
                if (arr[j] < arr[minIndex])
                {
                    minIndex = j;
                }
            }
            int temp = arr[minIndex];
            arr[minIndex] = arr[i];
            arr[i] = temp;
        }
    }

    public static void Main()
    {
        int[] arr = {64, 25, 12, 22, 11};
        Sort(arr);
        Console.WriteLine("Sorted array: " + string.Join(",
", arr));
    }
}
```

Java

```
public class SelectionSort {

    public static void sort(int[] arr) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            int minIndex = i;
            for (int j = i + 1; j < n; j++) {
                if (arr[j] < arr[minIndex]) {
                    minIndex = j;
                }
            }
            int temp = arr[minIndex];
            arr[minIndex] = arr[i];
            arr[i] = temp;
        }
    }

    public static void main(String[] args) {
        int[] arr = {64, 25, 12, 22, 11};
        sort(arr);
        System.out.println("Sorted array: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

Python

```
def selection_sort(arr):
    n = len(arr)

    for i in range(n - 1):
        min_index = i
        for j in range(i + 1, n):
            if arr[j] < arr[min_index]:
                min_index = j
        arr[i, min_index] = arr[min_index], arr[i]  # Swap
the found minimum element with the first element

if __name__ == "__main__":
    arr = [64, 25, 12, 22, 11]
    selection_sort(arr)
    print("Sorted array:", arr)
```

Explanation

- **C# and Java**: The code defines a `Sort` method to implement the selection sort algorithm, iterating through the array, finding the minimum element, and swapping it with the first unsorted element.
- **Python**: The `selection_sort` function does the same, with a more concise swapping mechanism using tuple assignment.

These implementations show how Selection Sort can be effectively coded in different programming languages.