



2) 정렬 알고리즘



매 강의 강의자료 시작에 PDF파일을 올려두었어요!

▼ PDF 파일

[수업 목표]

- 이 강의에서는 다양한 종류의 정렬 알고리즘을 배우게 됩니다.
- 각 알고리즘의 작동 방식과 시간 복잡도, 공간 복잡도를 분석하며, 실제 C# 코드를 통해 이해합니다.

[목차]

01. 정렬 알고리즘

02. C# Sort



모든 토글을 열고 닫는 단축키

Windows : **Ctrl** + **alt** + **t**

Mac : **⌘** + **⌥** + **t**

01. 정렬 알고리즘

▼ 1) 정렬 알고리즘이란?

- 정렬 알고리즘은 컴퓨터 과학에서 중요한 주제 중 하나입니다.
- 주어진 데이터 세트를 특정 순서(대개는 숫자의 오름차순 또는 내림차순, 문자열의 사전식 순서)로 배열하는 방법을 제공합니다.

▼ 2) 선택 정렬 (Selection Sort)

- 선택 정렬은 배열에서 최소값(또는 최대값)을 찾아 맨 앞(또는 맨 뒤)와 교환하는 방법입니다.
- 시간 복잡도: 최악의 경우와 평균적인 경우 모두 $O(n^2)$
- 공간 복잡도: $O(1)$ (상수 크기의 추가 공간이 필요하지 않음)

▼ 구현 예제

- 가장 작은 원소를 찾아서 맨 앞에 위치하는 것을 반복하여 정렬하는 알고리즘

```
int[] arr = new int[] { 5, 2, 4, 6, 1, 3 };

for (int i = 0; i < arr.Length - 1; i++)
{
    int minIndex = i;

    for (int j = i + 1; j < arr.Length; j++)
    {
        if (arr[j] < arr[minIndex])
        {
            minIndex = j;
        }
    }

    int temp = arr[i];
    arr[i] = arr[minIndex];
    arr[minIndex] = temp;
}

foreach (int num in arr)
{
    Console.WriteLine(num);
}
```

▼ 3) 삽입 정렬 (Insertion Sort)

- 삽입 정렬은 정렬되지 않은 부분에서 요소를 가져와 정렬된 부분에 적절한 위치에 삽입하는 방법입니다.
- 시간 복잡도: 최악의 경우 $O(n^2)$, 하지만 정렬되어 있는 경우에는 $O(n)$
- 공간 복잡도: $O(1)$ (상수 크기의 추가 공간이 필요하지 않음)

▼ 구현 예제

- 현재 위치에서 그 이하의 배열을 정렬된 배열 부분에 삽입하는 방법으로 정렬하는 알고리즘

```

int[] arr = new int[] { 5, 2, 4, 6, 1, 3 };

for (int i = 1; i < arr.Length; i++)
{
    int j = i - 1;
    int key = arr[i];

    while (j >= 0 && arr[j] > key)
    {
        arr[j + 1] = arr[j];
        j--;
    }

    arr[j + 1] = key;
}

foreach (int num in arr)
{
    Console.WriteLine(num);
}

```

▼ 4) 퀵 정렬 (Quick Sort)

- 퀵 정렬은 피벗을 기준으로 작은 요소들은 왼쪽, 큰 요소들은 오른쪽으로 분할하고 이를 재귀적으로 정렬하는 방법입니다.
- 시간 복잡도: 최악의 경우 $O(n^2)$, 하지만 평균적으로 $O(n \log n)$
- 공간 복잡도: 평균적으로 $O(\log n)$, 최악의 경우 $O(n)$ (재귀 호출에 필요한 스택 공간)

▼ 구현 예제

- 분할 정복 방법을 이용하여 정렬하는 알고리즘

```

void QuickSort(int[] arr, int left, int right)
{
    if (left < right)
    {
        int pivot = Partition(arr, left, right);

        QuickSort(arr, left, pivot - 1);
        QuickSort(arr, pivot + 1, right);
    }
}

int Partition(int[] arr, int left, int right)
{
    int pivot = arr[right];
    int i = left - 1;

```

```

        for (int j = left; j < right; j++)
        {
            if (arr[j] < pivot)
            {
                i++;
                Swap(arr, i, j);
            }
        }

        Swap(arr, i + 1, right);

        return i + 1;
    }

    void Swap(int[] arr, int i, int j)
    {
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }

    int[] arr = new int[] { 5, 2, 4, 6, 1, 3 };

    QuickSort(arr, 0, arr.Length - 1);

    foreach (int num in arr)
    {
        Console.WriteLine(num);
    }

```

▼ 5) 병합 정렬 (Merge Sort)

- 병합 정렬은 배열을 반으로 나누고, 각 부분을 재귀적으로 정렬한 후, 병합하는 방법입니다.
- 시간 복잡도: 모든 경우에 대해 $O(n \log n)$
- 공간 복잡도: $O(n)$ (정렬을 위한 임시 배열이 필요함)

▼ 구현 예제

- 분할 정복 방법을 이용하여 정렬하는 알고리즘

```

void MergeSort(int[] arr, int left, int right)
{
    if (left < right)
    {
        int mid = (left + right) / 2;

        MergeSort(arr, left, mid);
        MergeSort(arr, mid + 1, right);
        Merge(arr, left, mid, right);
    }
}

```

```

}

void Merge(int[] arr, int left, int mid, int right)
{
    int[] temp = new int[arr.Length];

    int i = left;
    int j = mid + 1;
    int k = left;

    while (i <= mid && j <= right)
    {
        if (arr[i] <= arr[j])
        {
            temp[k++] = arr[i++];
        }
        else
        {
            temp[k++] = arr[j++];
        }
    }

    while (i <= mid)
    {
        temp[k++] = arr[i++];
    }

    while (j <= right)
    {
        temp[k++] = arr[j++];
    }

    for (int l = left; l <= right; l++)
    {
        arr[l] = temp[l];
    }
}

int[] arr = new int[] { 5, 2, 4, 6, 1, 3 };

MergeSort(arr, 0, arr.Length - 1);

foreach (int num in arr)
{
    Console.WriteLine(num);
}

```

02. C# Sort

▼ 1) Sort 메서드

- **Sort** 메서드는 배열이나 리스트의 요소들을 정렬하는 메서드입니다.
- 정렬은 오름차순으로 수행되며, 요소들의 자료형에 따라 다양한 정렬 기준을 사용할 수 있습니다.
- **Sort** 메서드는 원래의 배열이나 리스트를 직접 수정하므로 반환값이 없습니다.


▼ 2) 사용 예제

```
// 정수 배열 정렬 예제
int[] numbers = { 5, 2, 8, 3, 1, 9, 4, 6, 7 };
Array.Sort(numbers);
Console.WriteLine(string.Join(", ", numbers));

// 문자열 리스트 정렬 예제
List<string> names = new List<string> { "John", "Alice", "Bob", "Eve", "David" };
names.Sort();
Console.WriteLine(string.Join(", ", names));
```

이전 강의

다음 강의

 1) 알고리즘 기초

17강 탐색 알고리즘