1. 정렬

ttps://namu.wiki/w/%EC%A0%95%EB%A0%AC%20%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98#s-2.1.1.1 꺼뮤위키 죄송합니다..

**컴퓨터 분야에서 중요시되는 문제 가운데 하나**

**어떤 데이터들이 주어졌을 때 이를 정해진 순서대로 나열하는 문제**

**실제 컴퓨터 분야에서 사용하는 데이터의 경우 숫자의 순서나 어휘의 순서대로 정렬한 다음 사용해야 되는 경우가 거의 항상 발생하는데 이걸 얼마나 효과적으로 해결할 수 있느냐가 정렬 문제의 핵심**

2. 종류

가. 시간복잡도에 따른 분류

**O(n²) 버블정렬(bubble) 선택정렬(selection) 삽입정렬(insertion)**

**O(nlogn) 병합정렬(merge) 힙정렬(heap) 퀵정렬(quick) 트리(tree) 팀정렬(tim) 인트로정렬(intro)**

**O(n) 기수정렬(radix) 등등….**

나. 세부 설명

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **구 분** | **그 림** | **특 징** | **시간/공간복잡도** | **비 고** |
| **버블정렬** |  |  | **n² / 1** | 실제 개발에서도 전혀 쓰이지 않는다.  어지간한 경우가 아닌 이상 피하라 |
| **칵테일정렬**  **(셰이커)** | external/upload.... | 홀수번째 돌 때는 앞부터,  짝수번째는 뒤부터 훑는 정렬  마지막과 처음이 번갈아가며 정렬 | **n² / 1** |  |
| **선택정렬** | external/upload.... | 1번째부터 끝까지 훑어서  가장 작은걸 앞 자리와 **교체**  용하는 정렬 방식을 가장 많이 닮음 | **n² /** | 버블정렬보다  두 배 빠름 |

나. 세부 설명 (계속)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **구 분** | **그 림** | **특 징** | **시간/공간복잡도** | **비 고** |
| **삽입정렬** | external/upload.... | 1번째부터 끝까지 훑어서  가장 작은 숫자를 해당위치에 **삽입**  인간에게 뭔가를 정렬하라고 하면 무의식적으로 사용하는 대표적인 알고리즘 | **n² / 1** | 평균적으로 n²중  빠른 편이나  작은게 뒤에 몰려있으면 오래걸림  다만, 자료구조에 하나씩 삽입/삭제 해야 하는 경우 현실적으로 최고의 정렬 알고리즘 |
| **병합정렬** | external/upload.... | 천재 폰노이만이 개발  Divide and conquer  (분할 정복) 방식  데이터 상태에 영향을  받지 않는다  데이터 적으면 Overhead발생 | **nlogn / n**  5-17-2-1 | https://www.ddanzi.com/ddanziNews/135996436 |
| **힙정렬** |  | 선택 정렬과 거의 같은  알고리즘. 단지 가장 큰 원소를 뒤로 보내는 데에 단순히 매번 쭉 돌면서 알아내느냐 힙을 사용하여 알아내느냐가 유일한 차이  추가 메모리 필요 X | **nlogn /**  **최악의 경우 n²** |  |
| **퀵정렬** | external/upload.... | 기준(pivot)을 정하고  작은 것을 앞으로 큰 것을 뒤로 (partition step) | **nlogn / logn**  **최악의 경우 n²**  **(pivot이 최소/최대)** | 5-17-3 |
| **팀정렬** |  | 병합 + 삽입정렬  Tim peter가 파이썬을 위해 C로 구현  실제 데이터는 대부분 이미 정렬되어 있을 것이다라고 가정하고 실제 데이터에서 고성능을 낼 수 있도록 설계한 알고리즘이다. 데이터가 엉망으로 뒤죽박죽 섞여 있는 일은 실제로는 거의 일어나지 않을 것이라 가정 | **nlogn /**  의 추가 메모리공간 필요 | 파이썬의 sort() |

4. 어디에 쓸까?

테이블이(가) 표시된 사진

자동 생성된 설명

