0. 자료구조

데이터를 효율적으로 엑세스하고 조작할 수 있도록 데이터의 구조를 만들어 데이터를 저장하고 관리하는 것

추상적 자료형

특별한 구현방법을 지정하지 않은 자료형. 일반적으로 알고리즘이 문제를 해결하는데 필요한 자료의 형태와 그 자료를 사용한 연산들을 수학적으로 정의한 모델.

자료구조 vs 추상적 자료형

추상적 자료형은 무엇이 구현되어져야 하는지 정의한 것으로 자료형의 논리적 형태인 반면, 자료구조는 어떻게 구현할 지를 파악해서 물리적 형태로 구현하는 것.

자료구조 ex. 배열, 연결 리스트

추상적 자료형 ex. 스택

자료구조의 종류

1. 단순구조: 기본적인 데이터 타입

ex. 정수, 실수, 문자, 불린

2. 선형구조: 선형적으로 연결되어 있는 구조. 앞 자료와 뒤 자료가 1 대 1

ex. 배열, 연결 리스트, 스택, 큐

3. 비선형구조: 자료 간 관계가 1 대 다, 다 대 다 구조로 계층구조 혹은 네트워크 망 구조

ex. 트리, 그래프

4. 파일구조: 레코드의 집합인 파일에 대한 구조

ex. 순차파일, 색인파일, 직접파일

1. 배열

연속적인 메모리 상에 동일한 데이터 타입의 요소들을 순차적으로 일렬로 저장하는 자료구조

# C/C++에서의 배열은 연속적인 메모리 공간에 대한 주소, C#에서의 배열은 배열 객체

배열의 차원

한 배열 요소를 선택하기 위해 사용하는 인덱스 수

가변배열

배열의 배열 요소가 배열 타입인 경우 각 배열 요소는 서로 다른 차원과 크기를 가질 수 있음

ex.

A[0] = new Int[2]

A[1] = new Int[3]

동적배열

배열을 확장하거나 축소하는 기능

새로운 요소가 추가될 때마다 배열의 크기를 늘려가는 방식이 있지만 O(n)의 시간이 소요되어 비효율적. 이를 보완하기 위한 것이 배열 크기의 확대(1.5~2배). 아무리 데이터가 많아져도 O(1)이라는 상수시간 안에 완료됨.

#점근표기법

알고리즘의 수행시간을 대략적으로 나타내는 방법

O: 최악 Ω: 최선 Θ: 최악과 최선을 동시에

예를 들어 데이터가 17번째 요소 추가 시, 배열 크기를 32인 새 배열로 확장한 후 16개의 기존 요소들을 복사하게 되는데, 이러한 16개의 복사 수행 시간을 기존 첫 요소부터 16번 요소까지의 수행시간에 분할하여 분산한다면 결국 수행시간은 O(1)이 됨.

* 분할상환분석

일부 수행에서 일어나는 비싼 비용을 분할시켜 다른 일반 수행들로 분할 상환하여 비용 계산

원형배열(원형버퍼, 링버퍼)

고정된 크기의 배열을 양 끝이 연결된 것처럼 사용할 수 있게 한 자료구조. 배열의 마지막 요소에 도착하면 다음 배열 요소는 첫번째 요소로 FIFO 방식의 큐나 데이터 스트림 버퍼 구현 시 사용

.Net의 배열 클래스

1차원에서 32차원까지 가능. 2GB까지 가능하지만 app.config에서 gcAllowVeryLargeObjects를 enable하면 더 크게도 가능.

동적배열으로는

ArrayList:: Objecvt

List<T>: Generic

1. 연결 리스트

각 노드가 데이터와 포인터를 가지고 있으면서 노드들이 한 줄로 연결되어 있는 방식. 노드들이 한 방향으로 다음 노드를 가리키는 리스트를 단일 연결 리스트이고, 각 노드가 이전 노드 다음 노두를 모두 가리키는 양방향 노드로 되어 있는 것은 이중 연결리스트.

단일 연결 리스트

단방향으로 노드들을 연결. 노드의 데이터피륻와 다음 노드를 가리키는 포인터로 구성. 첫 노드를 가리키는 헤드 필드를 사용하여 전체 리스트를 순차적으로 엑세스.