2021 학년도 1 학기

컴퓨터 정보과

자료구조(Data Structures)

담당교수 : 김주현

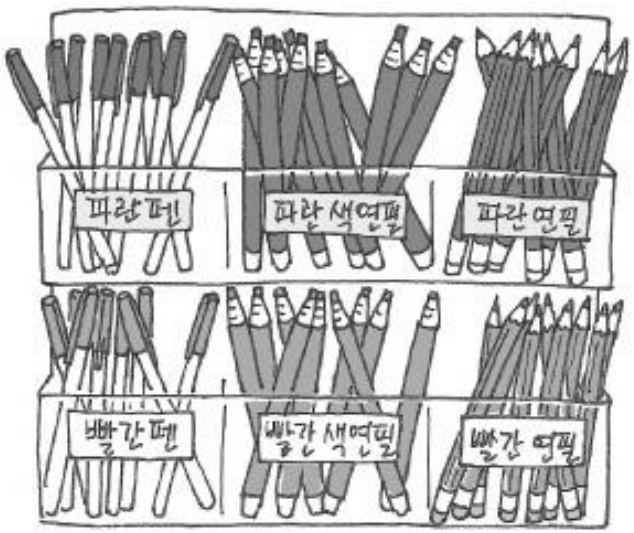
제 2 주차

DIT 동의과학대학교
DONG-EUI INSTITUTE OF TECHNOLOGY

❖ 자료구조란?

- 자료를 효율적으로 사용하기 위해서 자료의 특성에 따라서 분류하여 구성하고 저장 및 처리하는 모든 작업

[나쁜 자료구조]

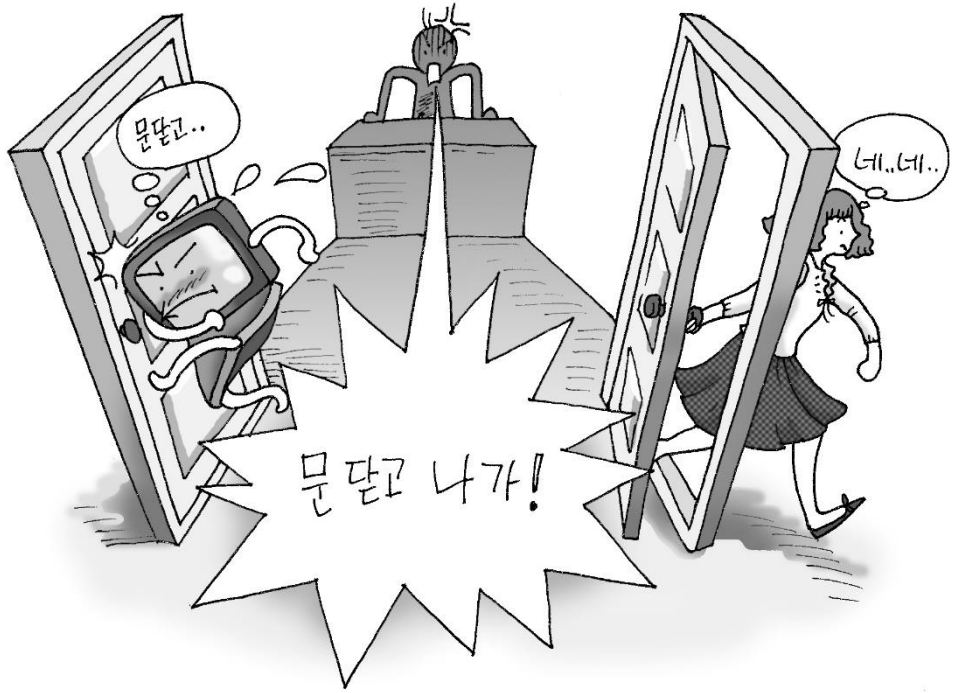


[좋은 자료구조]

자료구조의 개요

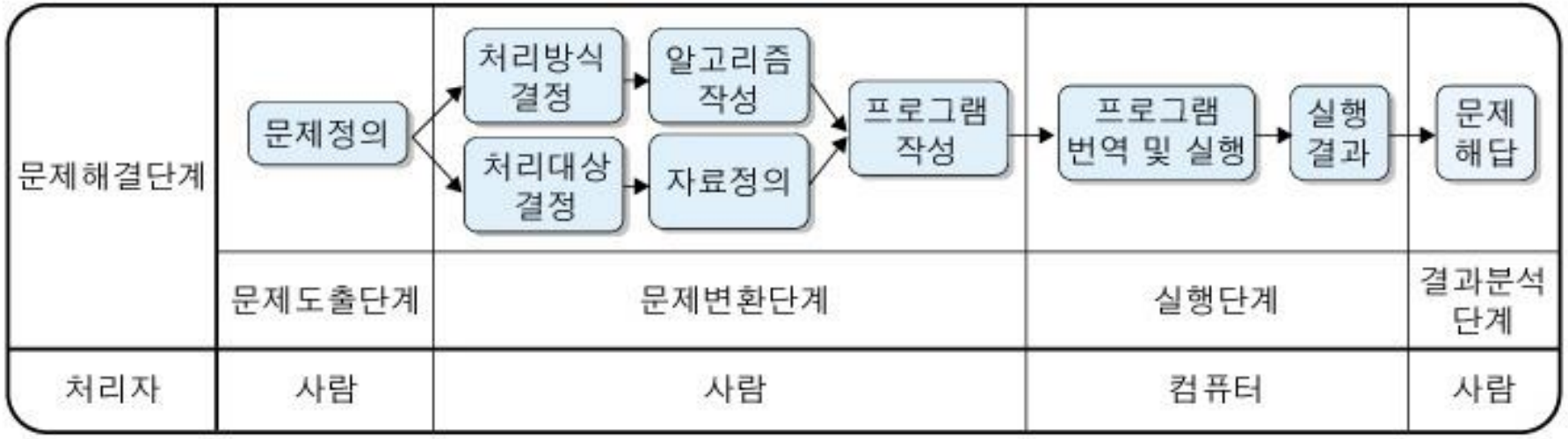
❖ 컴퓨터 분야에서 자료구조를 배우는 이유

- **컴퓨터는 사람이 원하는 것을 알아서 처리할 수 없음**
  - 예: 컴퓨터와 사람에게 “문닫고 나가” 라는 명령을 준 경우



자료구조의 개요

- **컴퓨터에 의한 문제 해결 과정**



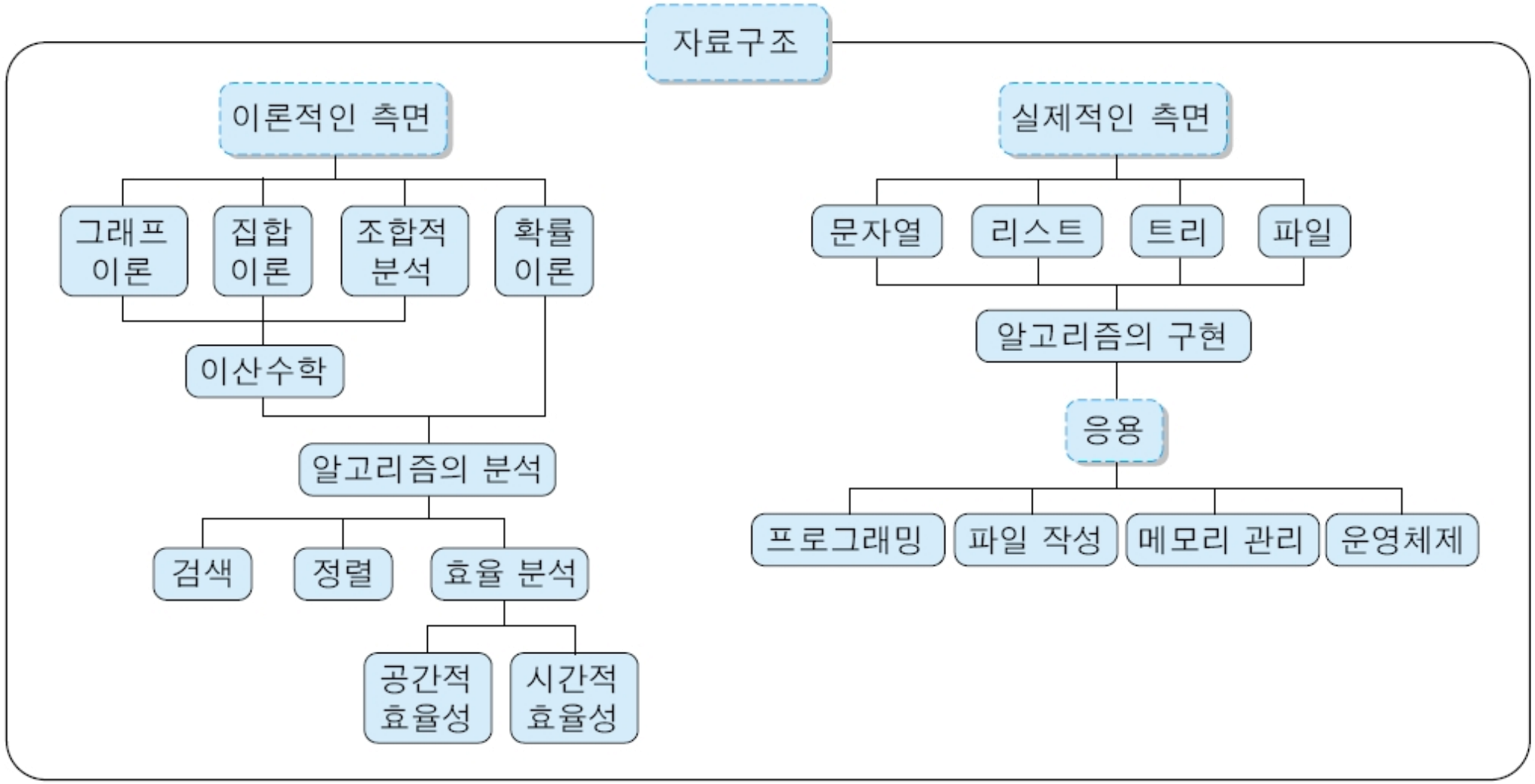
- **컴퓨터가 효율적으로 문제를 처리하기 위해서는 문제를 정의하고 분석하여 그에 대한 최적의 프로그램을 작성해야 한다.**
  - ☞ 자료구조에 대한 개념과 활용 능력 필요!

자료구조의 분류

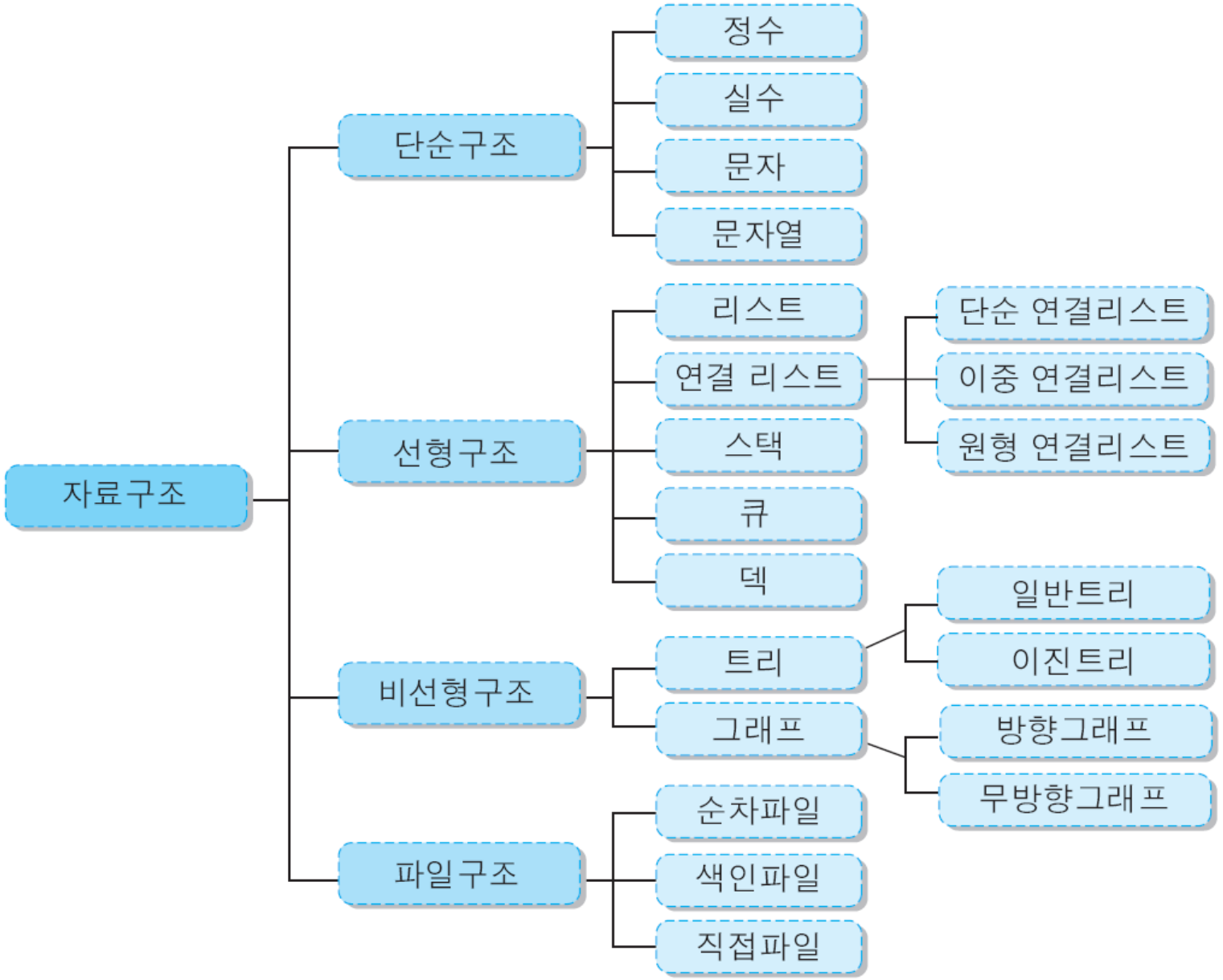
❖ 자료의 형태에 따른 분류

- **단순 구조**
  - 정수, 실수, 문자, 문자열 등의 기본 자료형
- **선형구조**
  - 자료들 간의 앞뒤 관계가 1:1의 선형 관계
  - 리스트, 연결리스트, 스택, 큐, 덱 등
- **비선형구조**
  - 자료들 간의 앞뒤 관계가 1:多, 또는 多:多의 관계
  - 트리, 그래프 등
- **파일구조**
  - 레코드의 집합인 파일에 대한 구조
  - 순차파일, 색인파일, 직접파일 등

자료구조의 분류


자료구조
이론적인 측면
그래프 이론
집합 이론
조합적 분석
확률 이론
이산수학
알고리즘의 분석
검색
정렬
효율 분석
공간적 효율성
시간적 효율성
실제적인 측면
문자열
리스트
트리
파일
알고리즘의 구현
응용
프로그래밍
파일 작성
메모리 관리
운영체제

자료구조의 분류


자료구조
단순구조
정수
실수
문자
문자열
선형구조
리스트
연결 리스트
단순 연결리스트
이중 연결리스트
원형 연결리스트
스택
큐
덱
비선형구조
트리
일반트리
이진트리
그래프
방향그래프
무방향그래프
파일구조
순차파일
색인파일
직접파일

추상자료형

❖ 뇌의 추상화 기능

- 기억할 대상의 구별되는 특징만을 단순화하여 기억하는 기능
  - 예: 뇌의 추상화 기능

추상자료형

❖ 컴퓨터를 이용한 문제해결에서의 추상화

- **크고 복잡한 문제를 단순화시켜 쉽게 해결하기 위한 방법**
- **자료 추상화(Data Abstraction)**
  - 처리할 자료, 연산, 자료형에 대한 추상화 표현
  - 자료 : 프로그램의 처리 대상이 되는 모든 것을 의미
  - 연산: 어떤 일을 처리하는 과정. 연산자에 의해 수행
    - ➢ 예) 더하기 연산은 +연산자에 의해 수행
  - 자료형 : 처리할 자료의 집합과 자료에 대해 수행할 연산자의 집합
    - ➢ 정수 자료형의 자료 : 정수의 집합. {..., -1, 0, 1, ...}
    - ➢ 연산자 : 정수에 대한 연산자 집합. {+, -, x, ÷, mod}

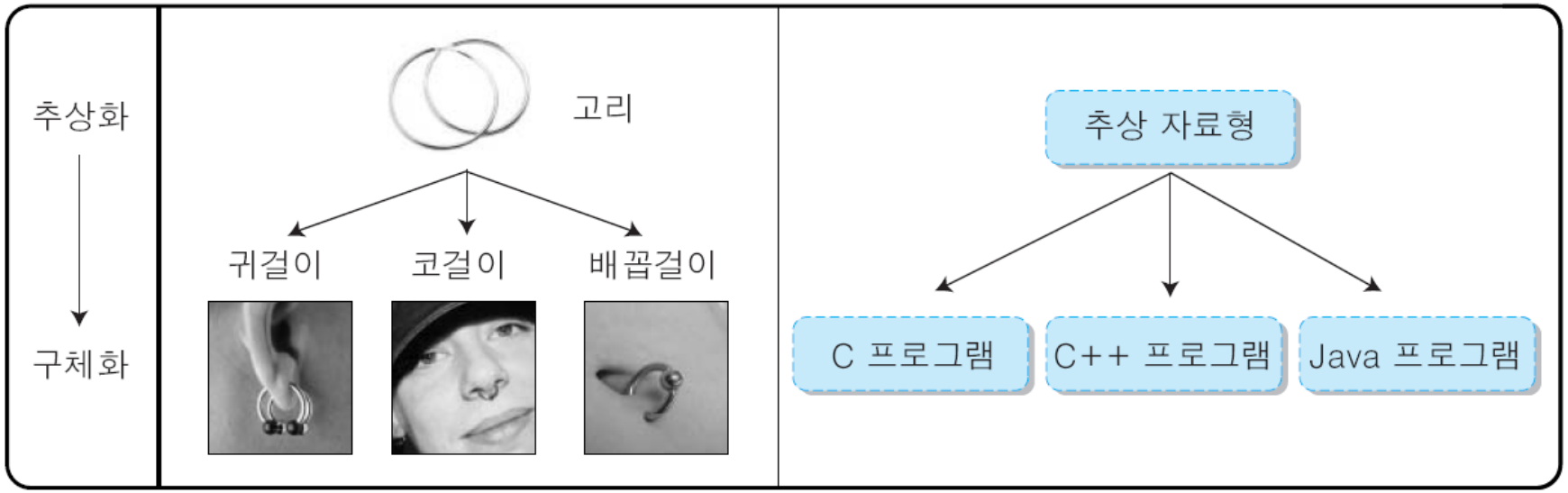
추상자료형

❖ 추상 자료형(ADT, Abstract Data Type)

- 자료와 연산자의 특성을 논리적으로 추상화하여 정의한 자료형

❖ 추상화와 구체화

- 추상화 – "무엇(what)인가?"를 논리적으로 정의
- 구체화 – "어떻게(how) 할 것인가?"를 실제적으로 표현



리스트

❖ 리스트(list)

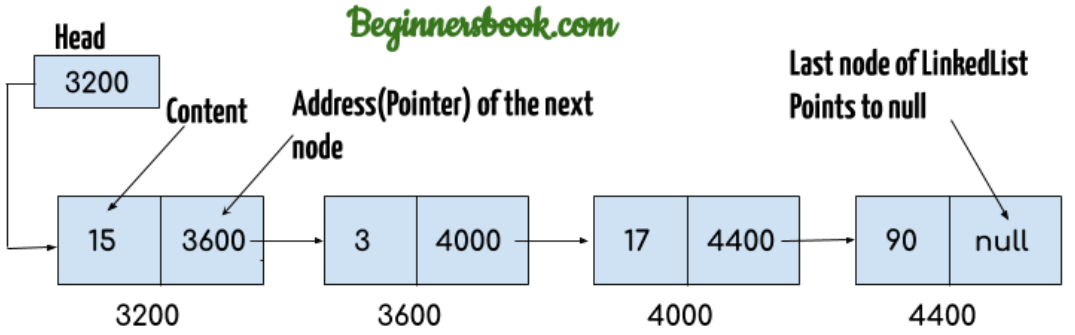
- 일련의 동일한 타입의 항목(item)들
- 자료를 나열한 목록
- 리스트의 예

동창 리스트	좋아하는 음식 리스트	오늘의 할일 리스트
김좌진	김치찌개	운동
신채호	닭볶음탕	자료구조 수업
안중근	된장찌개	동아리 공연 연습
이봉창	잡채	과제 제출
한용운	북어국	방청소

리스트의 구현

❖ 리스트의 구현

- 배열
- 연결 리스트



배열

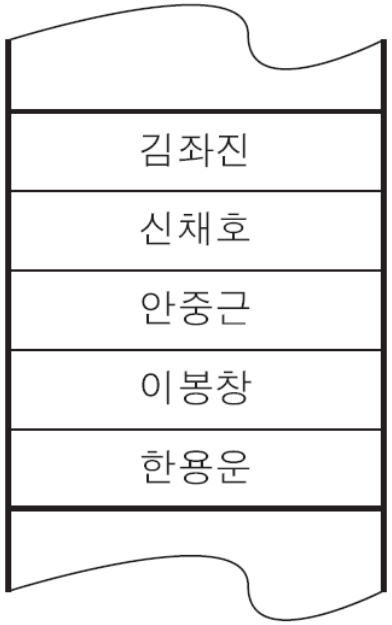
❖ 배열

- 배열(Array)은 동일한 타입의 원소들이 연속적인 메모리 공간에 할당되어 각 항목이 하나의 원소에 저장되는 기본적인 자료구조이다.
- 특정 원소에 접근할 때에는 배열의 인덱스를 이용하여 O(1) 시간에 접근할 수 있다.
- 새 항목이 배열 중간에 삽입되거나 중간에 있는 항목을 삭제하면, 뒤 따르는 항목들을 한 칸씩 뒤로 또는 앞으로 이동시켜야 하므로 삽입이나 삭제 연산은 항상 O(1) 시간에 수행할 수 없다

배열

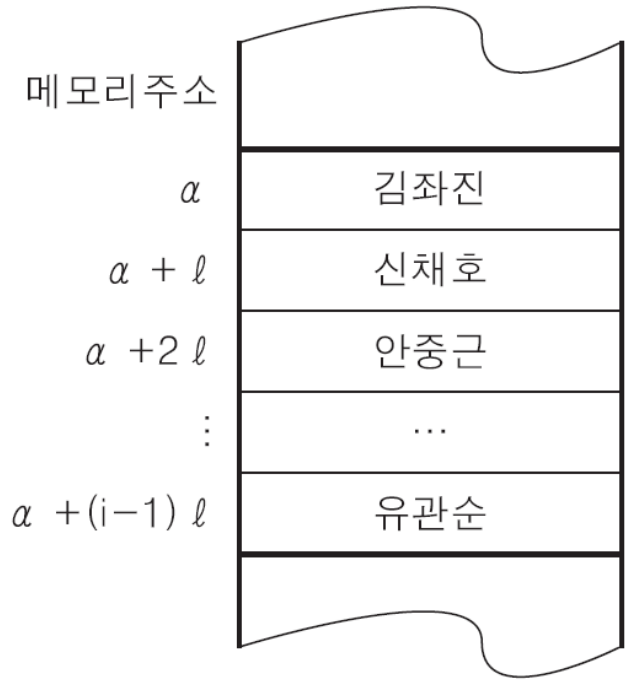
❖ 선형 리스트(배열)의 저장

- **원소들의 논리적 순서와 같은 순서로 메모리에 저장**
- **순차 자료구조**
  - 원소들의 논리적 순서 = 원소들이 저장된 물리적 순서
  - 동창 선형 리스트가 메모리에 저장된 물리적 구조



배열

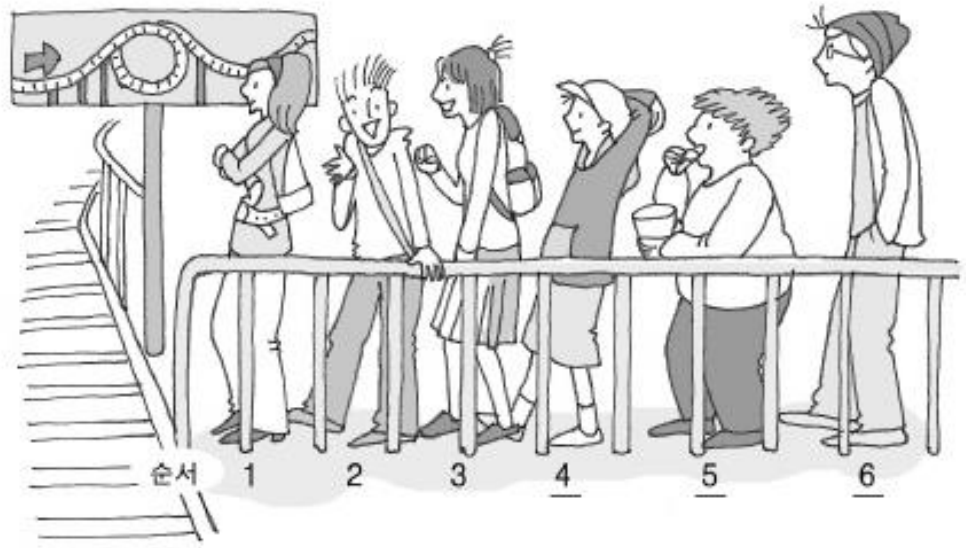
- **순차 자료구조의 원소 위치 계산**
  - 선형 리스트가 저장된 시작 위치 : α
  - 원소의 길이 : ℓ
  - i번째 원소의 위치 = $\alpha + (i-1) \times \ell$



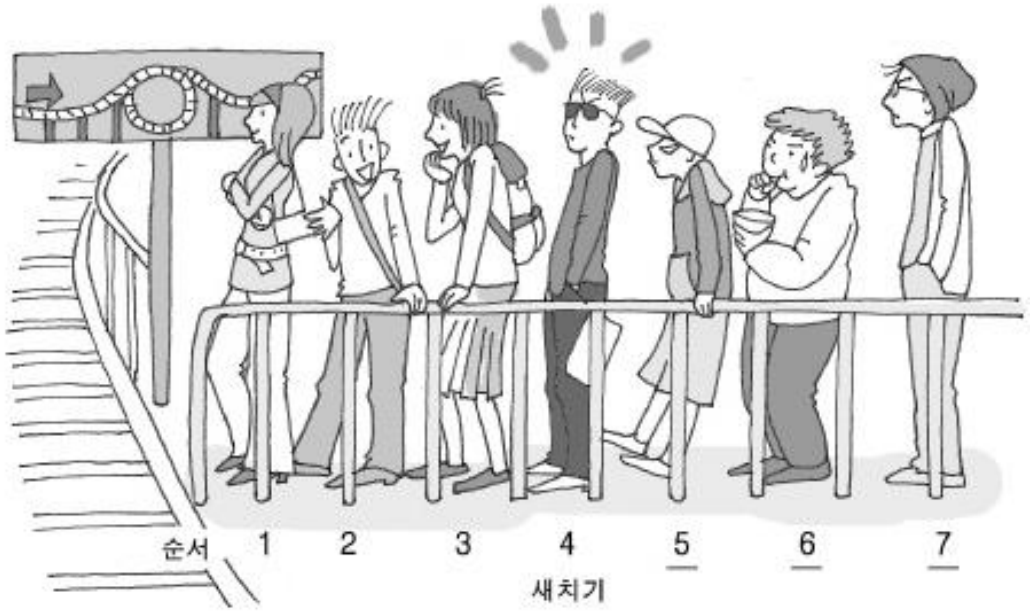
배열

❖ 선형 리스트에서의 원소 삽입

- 선형리스트 중간에 원소가 삽입되면, 그 이후의 원소들은 한자리씩 자리를 뒤로 이동하여 물리적 순서를 논리적 순서와 일치시킨다.



(a) 새치기 전



(b) 새치기 후

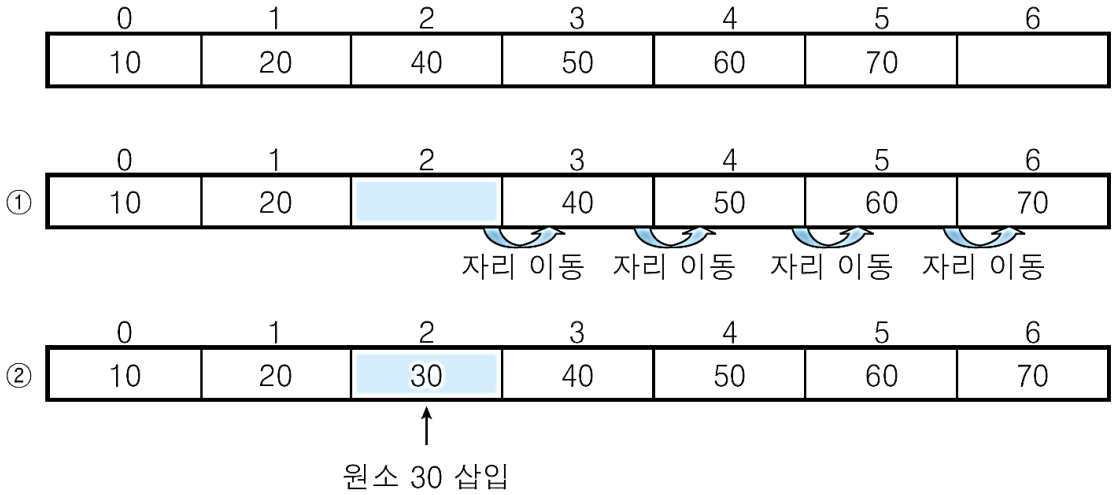
배열

원소 삽입 방법

① 원소를 삽입할 빈 자리 만들기

☞ 삽입할 자리 이후의 원소들을 한자리씩 뒤로 자리 이동 시키기

② 준비한 빈 자리에 원소 삽입하기



삽입할 자리를 만들기 위한 자리이동 횟수

- (n+1)개의 원소로 이루어진 선형 리스트에서 k번 자리에 원소를 삽입하는 경우 : k번 원소 부터 마지막 n번 원소까지 (n−k+1)개의 원소를 이동

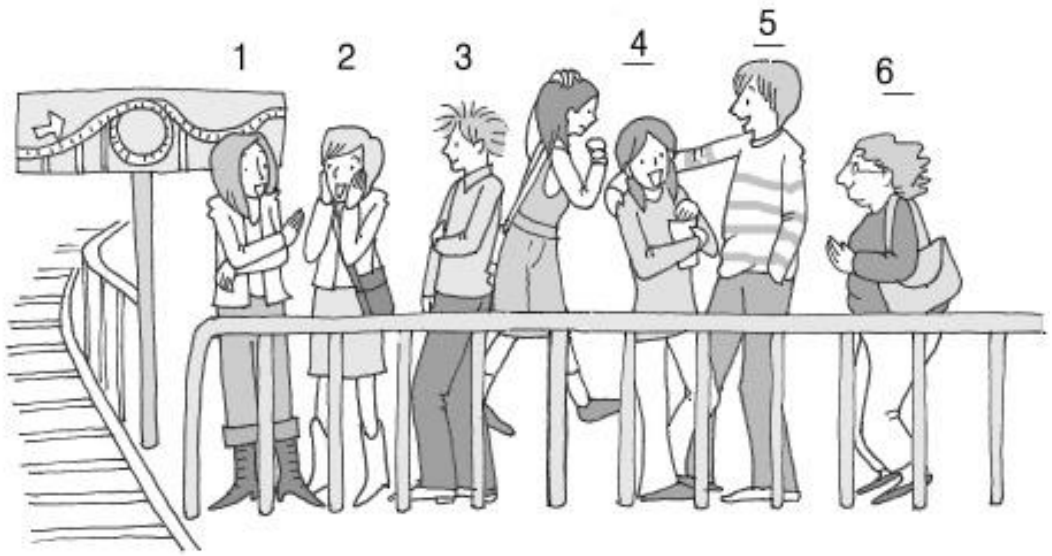
  ➢ **이동횟수 = n−k+1 = 마지막 원소의 인덱스 − 삽입할 자리의 인덱스 +1**

배열

❖ 선형 리스트에서의 원소 삭제

- 선형리스트 중간에서 원소가 삭제되면, 그 이후의 원소들은 한자리씩 자리를 앞으로 이동하여 물리적 순서를 논리적 순서와 일치시킴

(a) 나가기 전



(b) 나간 후

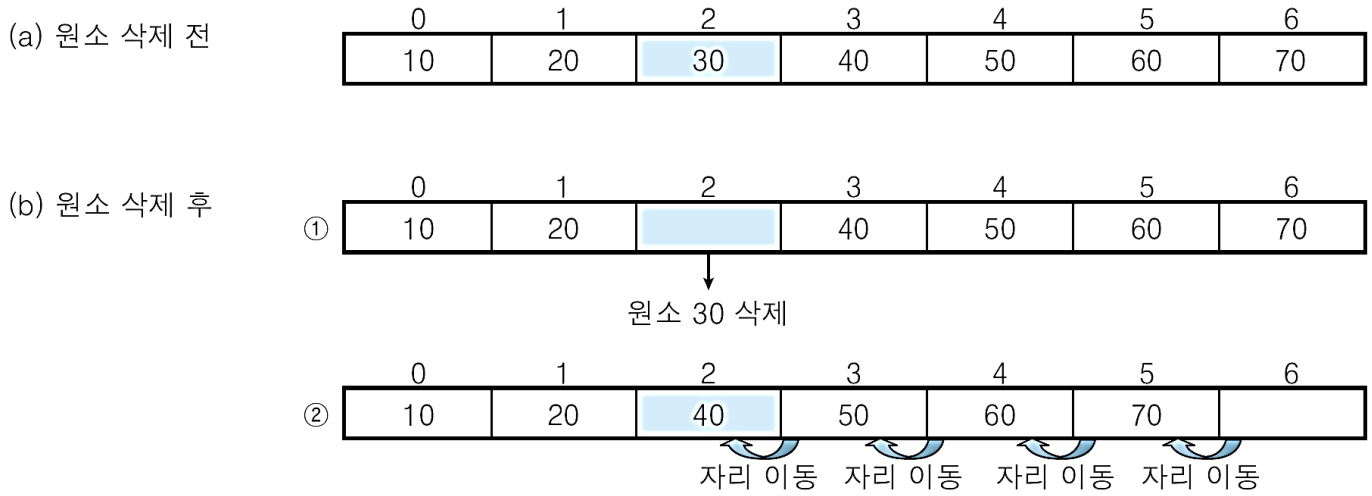
배열

원소 삭제 방법

① 원소 삭제하기

② 삭제한 빈 자리 채우기

☞ 삭제한 자리 이후의 원소들을 한자리씩 앞으로 자리 이동시키기



삭제 후, 빈 자리를 채우기 위한 자리이동 횟수

- (n+1)개 원소로 이루어진 선형 리스트에서 k번 자리의 원소를 삭제한 경우

  ➢ (k+1)번 원소부터 마지막 n번 원소까지 (n-(k+1)+1)개의 원소를 이동

  ➢ 이동횟수 = n-(k+1)+1 = n-k
  = 마지막 원소의 인덱스-삭제한 자리의 인덱스

1차원 배열

```
class OneDim{
public static void main(String srgs[]){
          int sale[] = new int[]{157, 209, 251, 312};

        for(int i=0; i<4; i++)
            System.out.printf("%d/4분기 : sale[%d]= %d %n", i+1, i, sale[i]);
          }
 }
```

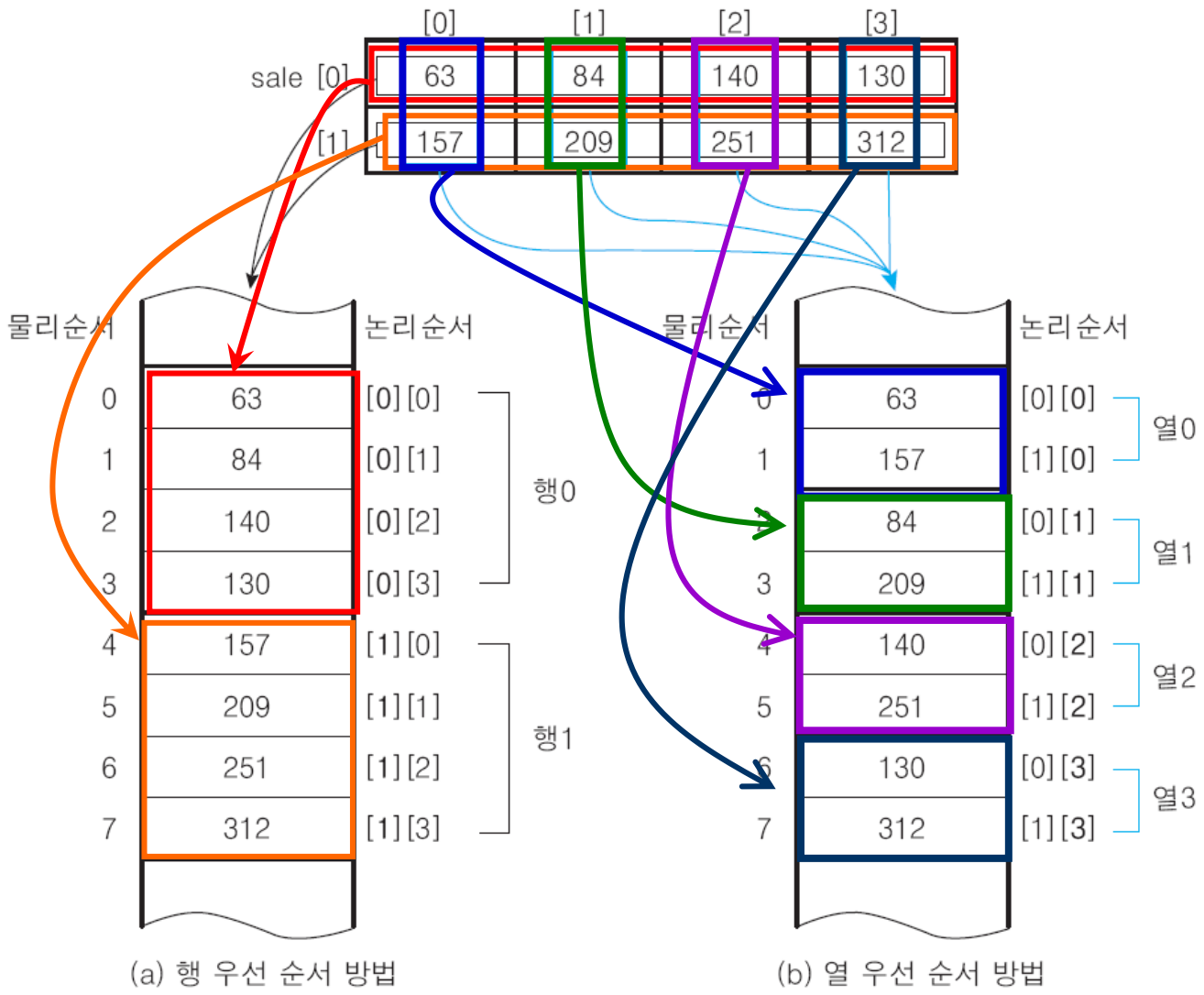
2차원 배열

2차원 배열의 물리적 저장 방법

- 2차원의 논리적 순서를 1차원의 물리적 순서로 변환하는 방법 사용
- 행 우선 순서 방법(row major order)
  - 2차원 배열의 첫 번째 인덱스인 행 번호를 기준으로 사용하는 방법
  - sale[0][0]=63, sale[0][1]=84, sale[0][2]=140, sale[0][3]=130,
    sale[1][0]=157, sale[1][1]=209, sale[1][2]=251, sale[1][3]=312
  - 원소의 위치 계산 방법 : $\alpha + (i \times n_j + j) \times \ell$
    - » 행의 개수가 n_i이고 열의 개수가 n_j인 2차원 배열 A[n_i][n_j]의 시작주소가 α이고 원소의 길이가 ℓ 일 때, i행 j열 원소 즉, A[i][j]의 위치
- 열 우선 순서 방법(column major order)
  - 2차원 배열의 마지막 인덱스인 열 번호를 기준으로 사용하는 방법
  - sale[0][0]=63, sale[1][0]=157, sale[0][1]=84, sale[1][1]=209,
    sale[0][2]=140, sale[1][2]=251, sale[0][3]=130, sale[1][3]=312
  - 원소의 위치 계산 방법 : $\alpha + (j \times n_i + i) \times \ell$

2차원 배열

➲ 물리적 구조



(a) 행 우선 순서 방법

(b) 열 우선 순서 방법

2차원 배열

```
class TwoDim{
public static void main(String srgs[]){
        int sale[][] = new int[][]{{63, 84, 140, 130},
                                   {157, 209, 251, 312}};

        for(int i=0; i<2; i++){
                for(int j=0; j<4; j++)
                        System.out.printf("%d/4분기 : sale[%d][%d]= %d %n", j+1, i, j, sale[i][j]);
                        System.out.println();
                }
        }
}
```

3차원 배열

3차원 배열의 물리적 저장 방법

- 3차원의 논리적 순서를 1차원의 물리적 순서로 변환하는 방법 사용
- 면 우선 순서 방법
  - 3차원 배열의 첫 번째 인덱스인 면 번호를 기준으로 사용하는 방법
  - 원소의 위치 계산 방법 : $\alpha + \{(i \times n_j \times n_k) + (j \times n_k) + k\} \times \ell$
    - 면의 개수가 n_i이고 행의 개수가 n_j이고, 열의 개수가 n_k 인 3차원 배열 $A[n_i][n_j][n_k]$
    - 시작주소가 α이고 원소의 길이가 ℓ 일 때, i면 j행 k열 원소 즉, A[i][j][k]의 위치
- 열 우선 순서 방법
  - 3차원 배열의 마지막 인덱스인 열 번호를 기준으로 사용하는 방법
  - 원소의 위치 계산 방법 : $\alpha + \{(k \times n_j \times n_i) + (j \times n_i) + i\} \times \ell$

3차원 배열

➲ 물리적 구조

물리 순서		논리 순서	
0	63	[0][0][0]	면0
1	84	[0][0][1]	
2	140	[0][0][2]	
3	130	[0][0][3]	
4	157	[0][1][0]	
5	209	[0][1][1]	
6	251	[0][1][2]	
7	312	[0][1][3]	
8	59	[1][0][0]	면1
9	80	[1][0][1]	
10	130	[1][0][2]	
11	135	[1][0][3]	
12	149	[1][1][0]	
13	187	[1][1][1]	
14	239	[1][1][2]	
15	310	[1][1][3]	

(a) 면 우선 순서 방법

물리 순서		논리 순서	
0	63	[0][0][0]	열0
1	59	[1][0][0]	
2	157	[0][1][0]	
3	149	[1][1][0]	
4	84	[0][0][1]	열1
5	80	[1][0][1]	
6	209	[0][1][1]	
7	187	[1][1][1]	
8	140	[0][0][2]	열2
9	130	[1][0][2]	
10	251	[0][1][2]	
11	239	[1][1][2]	
12	130	[0][0][3]	열3
13	135	[1][0][3]	
14	312	[0][1][3]	
15	310	[1][1][3]	

(b) 열 우선 순서 방법

3차원 배열

```
class Ex5_3{
public static void main(String srgs[]){
   int sale[][][] = new int [][][]{{{63, 84, 140, 130},
                                    {157, 209, 251, 312}},
                                   {{59, 80, 130, 135},
                                    {149, 187, 239, 310}}
                                   };

   for(int i=0; i<2; i++){
          System.out.printf("<< %d 팀 >> %n", i+1);
          for(int j=0; j<2; j++){
             for(int k=0; k<4; k++)
                    System.out.printf("%d/4분기 : sale[%d][%d][%d] = %d %n", k+1, i, j, k, sale[i][j][k]);
             System.out.println("------------------------");
           }
          System.out.println();
  }
}
}
```

Reference

- 자바로 배우는 쉬운 자료구조, 이지영, 한빛아카데미
- 자바와 함께하는 자료구조의 이해, 양성봉, 생능출판

감사합니다

DIT 동의과학대학교
DONG-EUI INSTITUTE OF TECHNOLOGY

DIT 동의과학대학교
DONG-EUI INSTITUTE OF TECHNOLOGY