

## 4주차 2차시 선형리스트의 구현

### 【학습목표】

1. 선형리스트의 C 프로그램 구현에 대하여 설명할 수 있다.
2. 1,2,3차원 배열을 이용한 구현을 구분할 수 있다.

### 학습내용1 : 1차원 배열을 이용한 구현

#### 1. 개요

- 순차 구조로 구현하기 위하여 배열을 사용
  - 배열 : <인덱스, 원소>의 쌍으로 구성
  - 배열의 인덱스 : 배열 원소의 순서 표현

#### 2. 1차원 배열을 이용한 선형리스트 구현

\* [표4-3] 분기별 판매량 리스트

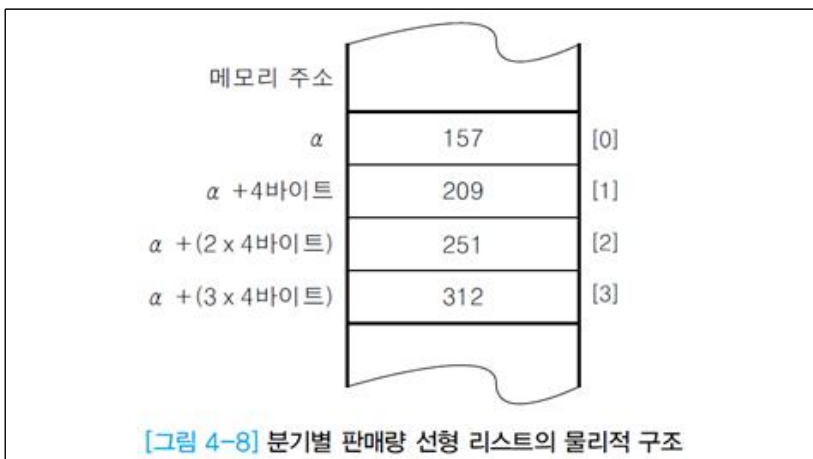
분기	1/4 분기	2/4 분기	3/4 분기	4/4 분기
판매량	157	209	251	312

\* 배열을 이용한 구현

- 선형리스트의 논리적 구조

	[0]	[1]	[2]	[3]
sale	157	209	251	312

- 선형리스트의 물리적 구조

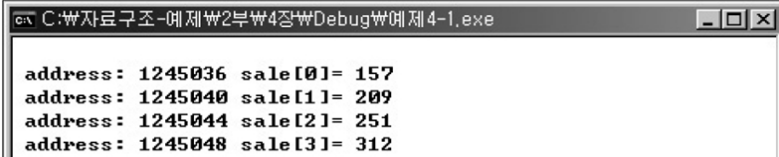


- 1차원 배열을 이용한 선형리스트 C 프로그램

```

01 #include <stdio.h>
02
03 void main()
04 {
05     int i, sale[4]={157, 209, 251, 312};
06
07     for(i=0; i<4; i++){
08         printf("\n address:%u sale[%d]=%d", &sale[i], i, sale[i]);
09     }
10
11     getchar();
12 }
    
```

\* 실행결과



\* 실행결과 확인

- 배열 sale 시작주소 : 1245036
- $\text{sale}[2] = \text{시작주소} + (\text{인덱스} \times 4 \text{ 바이트})$   
 $= 1245036 + (2 \times 4 \text{ 바이트})$   
 $= 1245044$
- 논리적인 순서대로 메모리에 연속 저장되었음을 확인

## 학습내용2 : 2차원 배열을 이용한 구현

### 1. 2차원 배열을 이용한 선형리스트 구현

\* [표4-4] 2009~2010년 분기별 노트북 판매량

년 \ 분기	1/4 분기	2/4 분기	3/4 분기	4/4 분기
2009년	63	84	140	130
2010년	157	209	251	312

\* 2차원 배열을 이용한 구현

- 논리적 구조

```
int sale[2][4] = {{63, 84, 140, 130},
                  {157, 209, 251, 312}};
```

	[0]	[1]	[2]	[3]
sale [0]	63	84	140	130
[1]	157	209	251	312

[그림 4-9] 2009~2010년 분기별 판매량 선형 리스트의 논리적 구조

\* 2차원 배열의 물리적 저장 방법

- 행 우선 순서 방법(row major order)

- 2차원 배열의 첫 번째 인덱스인 행을 기준으로 하여 같은 행 안에 있는 열을 저장

- 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 ni이고 열의 개수가 nj인 2차원 배열 A[ni][nj]의 시작주소가  $\alpha$ 이고  
원소의 길이가  $\ell$  일 때, i행 j열 원소 즉, A[i][j]의 위치

- 열 우선 순서 방법(column major order)

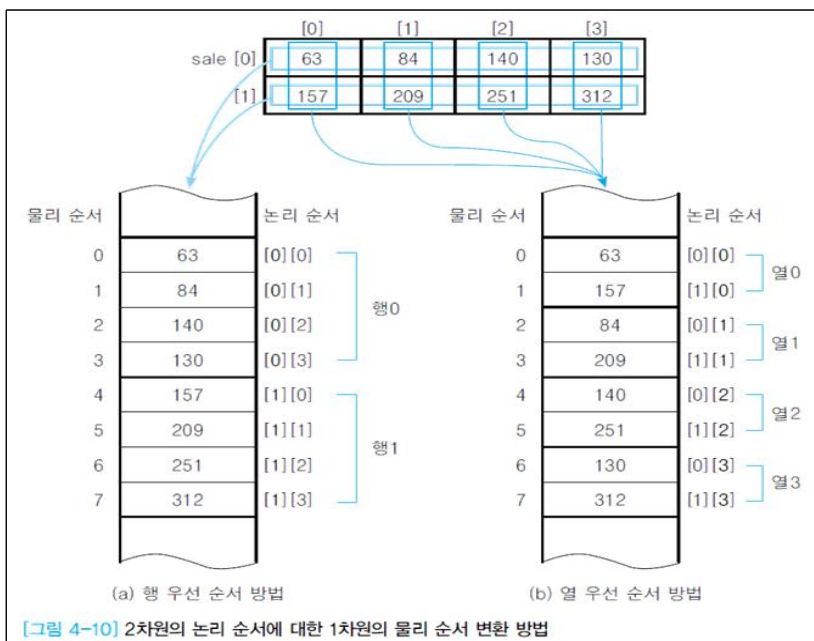
- 2차원 배열의 마지막 인덱스인 열을 기준으로 하여 같은 열 안에 있는 행을 저장

- 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 :  $\alpha + (j \times n_i + i) \times \ell$

\* 물리적 구조



[그림 4-10] 2차원의 논리 순서에 대한 1차원의 물리 순서 변환 방법

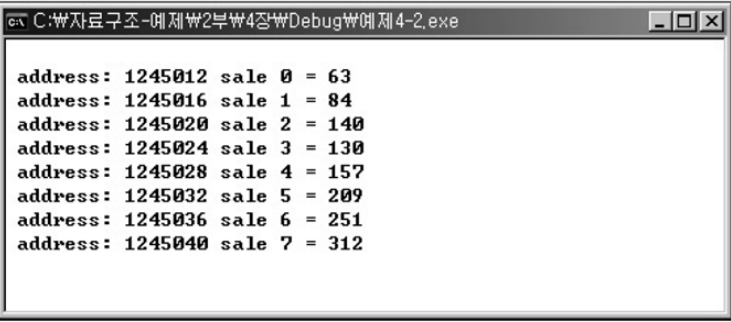
## \* 2차원 배열을 이용한 선형리스트 C 프로그램

```

01  #include <stdio.h>
02
03  void main()
04  {
05      int i, n=0, *ptr;
06      int sale[2][4] = {{63, 84, 140, 130},
07                      {157, 209, 251, 312}}; //2차원배열 초기화
08
09      ptr=&sale[0][0];
10      for(i=0; i<8; i++){
11          printf("\n address: %u sale %d = %d", ptr, i, *ptr);
12          ptr++;
13      }
14      getchar();
15  }

```

## \* 실행결과



```

address: 1245012 sale 0 = 63
address: 1245016 sale 1 = 84
address: 1245020 sale 2 = 140
address: 1245024 sale 3 = 130
address: 1245028 sale 4 = 157
address: 1245032 sale 5 = 209
address: 1245036 sale 6 = 251
address: 1245040 sale 7 = 312

```

## \* 실행결과 확인

- 시작주소  $\alpha=1245012$ ,  $ni=2$ ,  $nj=4$ ,  $i=1$ ,  $j=1$ ,  $\ell=4$
- $sale[1][1]=209$ 의 위치  $= \alpha + (i \times nj + j) \times \ell$ 

$$= 1245012 + (1 \times 4 + 1) \times 4$$

$$= 1245012 + 20$$

$$= 1245032$$
- C 컴파일러가 행 우선 순서 방법을 사용한다는 것을 확인!

## 학습내용3 : 3차원 배열을 이용한 구현

### 1. 3차원 배열을 이용한 선형리스트 구현

\* [표4-5] 1팀과 2팀에 대한 2009~2010년 분기별 노트북 판매량 리스트

1팀				
년 \ 분기	1/4 분기	2/4 분기	3/4 분기	4/4 분기
2009년	63	84	140	130
2010년	157	209	251	312

2팀				
년 \ 분기	1/4 분기	2/4 분기	3/4 분기	4/4 분기
2009년	59	80	130	135
2010년	149	187	239	310

\* 3차원 배열을 이용한 구현

```
int sale[2][2][4] = {{{63, 84, 140, 130},
                      {157, 209, 251, 312}},
                     {{59, 80, 130, 135},
                      {149, 187, 239, 310}}}
;
```



[그림 4-11] 1팀과 2팀의 2009~2010년 분기별 판매량 선형 리스트의 논리적 구조

\* 3차원 배열의 물리적 저장 방법

- 3차원 논리적 순서를 1차원 물리적 순서로 변환하는 방법을 사용

- 면 우선 순서 방법

- 3차원 배열의 첫 번째 인덱스인 면을 1차 기준으로 면 안에 있는 행을 먼저 저장 다시 행을 2차 기준으로 하여 같은 행 안에 있는 열을 저장하는 방법

- 원소의 위치 계산 방법 :  $\alpha + \{(i \times n_j \times n_k) + (j \times n_k) + k\} \times \ell$

면의 개수가  $n_i$ 이고 행의 개수가  $n_j$ 이고, 열의 개수가  $n_k$  인 3차원 배열  $A[n_i][n_j][n_k]$ , 시작주소가  $\alpha$ 이고 원소의 길이가  $\ell$  일 때,  $i$ 면  $j$ 행  $k$ 열 원소 즉,  $A[i][j][k]$ 의 위치

- 열 우선 순서 방법

- 열을 1차 기준으로 열 안에 있는 행을 먼저 저장하고 다시 2차 기준으로 하여 같은 행에 대한 면을 저장

- 원소의 위치 계산 방법 :  $\alpha + \{(k \times n_j \times n_i) + (j \times n_i) + i\} \times \ell$

- 물리적 구조

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

(a) 면 우선 순서 방법 (b) 열 우선 순서 방법

[그림 4-12] 3차원의 논리 순서에 대한 1차원의 물리 순서

\* 3차원 배열을 이용한 선형리스트 C 프로그램

```

01 #include <stdio.h>
02
03 void main()
04 {
05     int i, n=0, *ptr;
06     int sale[2][2][4] = {{(63, 84, 140, 130)}, //면0
07                          {(157, 209, 251, 312)},
08                          {(59, 80, 130, 135)}, //면1
09                          {(149, 187, 239, 310)}};
10
11     ptr=&sale[0][0][0]; //3차원 배열을 포인터에 설정
12     for(i=0; i<16; i++){
13         printf("\n address: %u sale %2d = %3d", ptr, i, *ptr);
14         ptr++;
15     }
16     getchar();
17 }

```

\* 실행결과

```

C:\자료구조-예제\2부\4장\Debug\예제4-3.exe

address: 1244980 sale 0 = 63
address: 1244984 sale 1 = 84
address: 1244988 sale 2 = 140
address: 1244992 sale 3 = 130
address: 1244996 sale 4 = 157
address: 1245000 sale 5 = 209
address: 1245004 sale 6 = 251
address: 1245008 sale 7 = 312
address: 1245012 sale 8 = 59
address: 1245016 sale 9 = 80
address: 1245020 sale 10 = 130
address: 1245024 sale 11 = 135
address: 1245028 sale 12 = 149
address: 1245032 sale 13 = 187
address: 1245036 sale 14 = 239
address: 1245040 sale 15 = 310

```

\* 실행결과 확인

- 시작주소 $\alpha=1244980$ ,  $n_i=2$ ,  $n_j=2$ ,  $n_k=4$ ,  $i=1$ ,  $j=1$ ,  $k=2$ ,  $\ell=4$
- $\text{sale}[1][1][2]=239$ 의 위치  $= \alpha + \{(i \times n_j \times n_k) + (j \times n_k) + k\} \times \ell$   
 $= 1244980 + \{(1 \times 2 \times 4) + (1 \times 4) + 2\} \times 4$   
 $= 1244980 + 56$   
 $= 1245036$
- C 컴파일러가 먼 우선 순서 방법으로 3차원 배열을 저장함을 확인!

【학습정리】

1. 배열은 <인덱스, 원소>의 쌍으로 구성되어 메모리에 연속적으로 할당되는데, 인덱스는 배열 원소의 순서를 표시하고 배열 원소들이 순서대로 메모리에 연속하여 순차 저장된다.
2. C 프로그래밍의 배열을 사용하여 선형리스트를 구현한다.