

5

배열

- 학습목표**
- 배열의 개념과 구조를 이해하고 배열의 선언과 초기화 방법을 설명할 수 있다.
 - 1차원, 2차원 배열을 사용한 프로그램을 작성할 수 있다.

생각 펼치기

사물함, 공동주택의 우편함, 도서관의 책 등과 같이 동일한 성격의 자료들을 저장하기 위해 프로그램에서는 배열과 같은 구조를 사용할 수 있다. 배열을 사용하여 내용 요소를 분류 및 정리하고 다양한 형태로 구조화하면, 자료를 보다 체계적이고 효율적으로 관리할 수 있다.



핵심 질문

▶ C 언어로 배열을 선언하고 초기화하는 방법은 무엇일까?

미션

이 단원을 학습하면서 해결해 보자.



태양에서 수성, 금성, 지구, 화성까지의 거리(km)는 표와 같다.

행성	거리(km)
수성	약 58,000,000
금성	약 108,000,000
지구	약 150,000,000
화성	약 228,000,000


빛의 속도가 300,000km/s일 때, 태양으로부터 각 행성까지 빛이 도달하는 시간(분)을 구하는 프로그램을 작성해 보자.

1 배열에 대해 알아보자

문제 해결을 위해 프로그램에서는 데이터를 저장할 수 있는 다량의 변수를 사용한다. 변수 하나에는 하나의 데이터만 저장할 수 있으므로 한꺼번에 여러 자료형의 데이터를 저장하고 처리하려면 그 수만큼의 변수를 지정해서 사용해야 한다. 이때, 배열을 사용하면 보다 효율적으로 프로그램을 작성할 수 있다.

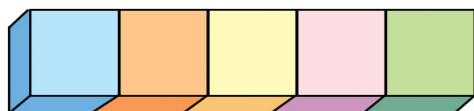
```
int Yang;
int Kang;
int Kim;
int Park;
int Lee;
```

Yang Kang Kim Park Lee



(a) 개별 변수의 사용

```
int height[5];
```



height[0] height[1] height[2] height[3] height[4]

(b) 배열 사용

[그림 II-4] 정수형 변수 5개 선언과 정수형 배열 선언

배열은 동일한 타입의 데이터가 여러 개 저장되어 있는 데이터 저장 장소이다. 예를 들어, 5명의 자료를 저장하려고 할 때 개별 변수를 사용할 수도 있지만, 만약 필요한 학생의 정보가 많아지면 일일이 그 수만큼 변수를 선언해야 하기 때문에 효율적이지 못하다.

그러나 배열을 사용하였을 경우에는 반복문을 통해서 각 배열의 요소에 접근하여 데이터를 처리할 수 있기 때문에 쉽고 편리하게 해결할 수 있다.

배열은 동일한 자료형을 가진 변수들을 한 번에 정의하고, 배열의 주소에 해당하는 첨자를 이용하여 배열 안의 자료를 구분해요.



300명의 이름과 키를 기억하려면 변수가 너무 많이 필요해. 좋은 방법이 없을까?



배열을 이용하면 300개도 한 번에 선언할 수 있어. 변수명은 하나이지만 저장 공간은 원하는 만큼 지정할 수 있는 것이 배열이거든.



2 1차원 배열에 대해 알아보자

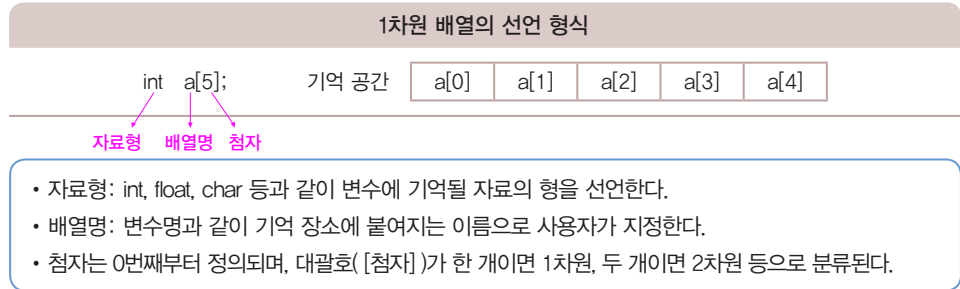
하나의 행 또는 하나의 열로만 이루어진 배열을 **1차원 배열**이라고 한다. 같은 자료형과 이름으로 여러 개의 기억 장소를 지정할 때 1차원 배열을 사용한다.

1 배열의 선언

다양한 배열의 선언

- `int e[35], m[35];`: 정수형으로 배열명 `e`와 `m`은 35개의 기억 장소를 지정한다.
- `double avg[10];`: 실수형으로 배열명 `avg`는 10개의 기억 장소를 지정한다.
- `char code[20];`: 문자형으로 배열된 `code`는 20개의 기억 장소를 지정한다.
- `char` 배열에서 문자열을 저장할 경우 마지막에 `'\0'` 문자가 저장되어야 하므로, 저장할 문자의 개수보다 하나 더 크게 크기를 선언해야 한다.

1차원 배열을 선언하는 형식은 다음과 같다.



2 배열의 초기화

배열을 선언하면서 각 배열 요소의 값을 초기화하기 위해서는 배열의 자료형에 맞는 값을 할당하여야 한다.

예 1 일반적으로 배열을 선언하고 초기화하는 방법

`int a[5] = {10, 20, 30, 40, 50};`

기억 공간	10	20	30	40	50
	<code>a[0]</code>	<code>a[1]</code>	<code>a[2]</code>	<code>a[3]</code>	<code>a[4]</code>

예 2 배열 크기보다 요소의 초기화 값이 적게 지정된 경우: 나머지는 '0'으로 초기화된다.

`int b[5] = {3, 5, 7};`

기억 공간	3	5	7	0	0
	<code>b[0]</code>	<code>b[1]</code>	<code>b[2]</code>	<code>b[3]</code>	<code>b[4]</code>

예 3 배열의 크기를 지정하지 않은 경우: 배열의 크기를 지정하지 않았을 때에는 컴파일러가 초기화 값을 가지고 배열의 크기를 지정한다.

`int d[] = {1, 2, 3, 4, 5};`

기억 공간	1	2	3	4	5
	<code>d[0]</code>	<code>d[1]</code>	<code>d[2]</code>	<code>d[3]</code>	<code>d[4]</code>

배열의 모든 요소를 0으로 초기화

`int a[5] = {0};` 과 같이 설정하면, 배열의 모든 요소값을 0으로 초기화할 수 있다.



예제

배열에 기억된 5개의 데이터를 제어문을 이용하여 출력하는 프로그램을 작성해 보자.

프로그램

```

01  #include <stdio.h>
02  int main( )
03  {
04      int height[5] = {172,181,163,175,143};
05      int i;
06      for (i = 0; i < 5; i++)
07      {
08          printf("%d ", height[i]);
09      }
10      return 0;
11  }

```

배열 변수의 첨자 부분을 반복문의 변수와 연결하여 배열의 값을 차례대로 출력할 수 있음.

실행 결과

172 181 163 175 143



예제

1차원 배열에 기억된 5개의 데이터의 합계를 구하는 프로그램을 작성해 보자.

프로그램

```

01  #include <stdio.h>
02  int main( )
03  {
04      int score[5] = {72,98,85,66,78};
05      int i, sum=0;
06      for (i = 0; i < 5; i++)
07      {
08          sum = sum + score[i];
09      }
10      printf("총점은 %d입니다.", sum);
11      return 0;
12  }

```

실행 결과

총점은 399입니다.



예제

문자열을 입력받아 문자열에 포함된 각 문자의 출현 횟수를 출력하는 프로그램을 작성해 보자(키보드 입력은 소문자만 입력받는다고 가정한다.).


프로그램

```
01 #include <stdio.h>
02 int main( )
03 {
04     char str[100];
05     int alphabet[26] = { 0 };
06     int i = 0;
07     scanf("%s", str);
08     while (str[i] != '\0')
09     {
10         ++alphabet[str[i] - 97];
11         i++;
12     }
13     for (i = 0; i < 26; i++)
14     {
15         if (alphabet[i] != 0)
16             printf("%c : %d\n", i + 97, alphabet[i]);
17     }
18     return 0;
19 }
```

키보드로 입력받은 문자열을 저장할 배열을 선언함.

소문자 'a'의 코드값이 97이므로 'a'의 횟수는 배열 요소의 첫 번째 칸에 저장됨.

실행 결과

```
hello 
e : 1
h : 1
l : 2
o : 1
```

프로그램 설명

- 08 while 반복문을 사용한 이유는 사용자가 입력할 문자열의 길이를 예측할 수 없기 때문이다. 조건에서 '\0' 문자가 아닐 때까지 반복하는 의미는 문자열의 마지막이 '\0' 문자로 끝나기 때문이다.
- 10 문자열 요소 값에서 97을 빼면(97)은 아스키 코드값으로 소문자 a를 나타냄.) alphabet 배열의 요소 위치를 나타낸다. 이 값을 1증가시킨다.
- 15 alphabet 배열의 카운팅 된 숫자가 0이 아니라면 알파벳 문자와 개수를 출력한다. 13번 라인에서 for 반복문을 사용한 이유는 고정된 크기의 배열의 요소를 검사하는 것으로 반복의 횟수를 예측할 수 있기 때문이다.

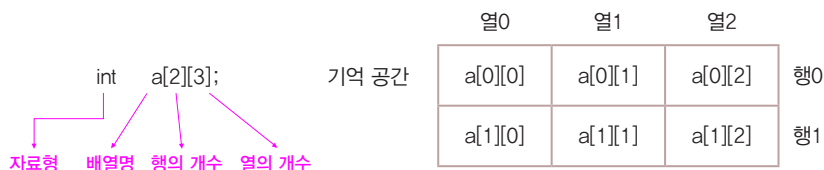
3 2차원 배열에 대해 알아보자

현실 세계의 문제를 해결하기 위해서는 1차원 배열로는 문제를 해결하기가 어려운 경우가 있다. 2차원 배열은 1차원 배열이 여러 개 필요할 때 사용하며, 메모리 공간에 연속적으로 저장된 데이터를 인간이 이해하기 쉽게 행과 열의 논리적인 2차원 공간으로 해석하여 사용할 수 있기 때문에 유용하게 사용된다.

1 배열의 선언

2차원 배열을 선언하는 형식은 다음과 같다.

2차원 배열 선언 형식



- 자료형: int, float, char 등과 같이 변수에 기억될 자료의 형을 선언한다.
- 배열명: 변수명과 같이 기억 장소에 붙여지는 이름으로 임의로 지정한다.
- 1차원 배열과 다르게 2차원 배열은 대괄호 안에 행의 개수와 열의 개수를 지정하여 2차원 형태의 배열을 선언한다.

2 배열의 초기화

2차원 배열을 선언하면서 각 배열 요소의 값을 행과 열에 알맞게 초기화하기 위한 방법은 다음과 같다.

예 1 배열의 요소를 쉼표를 사용하여 초기화하는 방식

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

	a[0][0]	a[0][1]	a[0][2]
기억 공간	1	2	3
	4	5	6
	a[1][0]	a[1][1]	a[1][2]

1차원 배열과 2차원 배열의 예

- 1차원 배열: 한 줄의 주차 라인에서 주차된 차, 명절표 등
- 2차원 배열: 영화관, 강당, 아파트, 비행기 좌석 등

예 2 총 배열 요소의 개수보다 초기화 값이 적은 경우

int b[2][3] = {1, 2, 3};

	b[0][0]	b[0][1]	b[0][2]
기억 공간	1	2	3
	0	0	0
	b[1][0]	b[1][1]	b[1][2]

예 3 중괄호를 이용한 배열 요소의 초기화

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

	c[0][0]	c[0][1]	c[0][2]
기억 공간	1	2	3
	4	5	6
	c[1][0]	c[1][1]	c[1][2]

예 4 배열의 열의 개수만을 이용한 초기화

int d[2][3] = {{1}, {2, 3}};

	d[0][0]	d[0][1]	d[0][2]
기억 공간	1	0	0
	2	3	0
	d[1][0]	d[1][1]	d[1][2]

스스로 해결하기

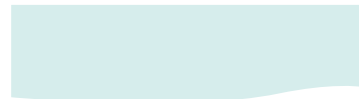
제어문을 이용하여 2차원 배열에 저장된 값을 출력하는 프로그램을 실행했을 때 예측되는 결과값을 작성해 보자.

프로그램

```

01  #include <stdio.h>
02  int main( )
03  {
04      int i,j;
05      int a[2][3] = {1,2,3,4,5,6 };
06      for (i = 0; i < 2; i++) {
07          for (j = 0; j < 3; j++) {
08              printf("%d ", a[i][j]);
09          }
10          puts("");
11      }
12      return 0;
13  }
```

실행 결과



다음은 아파트 각 층별 호수의 관리비 내역을 논리적인 표로 나타낸 것이다. 2차원 배열을 이용해서 각 층의 관리비 평균과 아파트 전체 층의 관리비 평균을 구하는 프로그램을 작성해 보자.

	1호	2호	3호	4호
3층	127,000	133,000	155,000	143,000
2층	136,000	142,000	112,000	135,000
1층	111,000	125,000	136,000	98,000

프로그램

```

01 #include <stdio.h>
02 int main( )
03 {
04     int floor,ho;
05     int apt[3][4] = {{111000,125000,136000,98000},
06                     {136000,142000,112000,135000},
07                     {127000,133000,155000,143000} };
08     int sumFloor[3] = {0,0,0};
09     for (floor = 0; floor < 3; floor++)
10     {
11         for (ho = 0; ho < 4; ho++) {
12             sumFloor[floor] = sumFloor[floor] + apt[floor][ho];
13         }
14         printf("%d층의 관리비 평균은 %1.f원 입니다.\n",
15             floor+1, sumFloor[floor]/4.0);
16     }
17     printf("아파트 전체층의 관리비 평균은 %0.f원 입니다.",
18         (sumFloor[0]+sumFloor[1]+sumFloor[2])/12.0);
19     return 0;
20 }
```

각 층의 호수의 관리비 합계를 저장하는 연산

층을 나타냄.

1, 2, 3층의 관리비를 모두 더해 12가구로 나눔.

실행 결과

1층의 관리비 평균은 117500원 입니다.
 2층의 관리비 평균은 131250원 입니다.
 3층의 관리비 평균은 139500원 입니다.
 아파트 전체층의 관리비 평균은 129417원 입니다.

예제

2차원 배열을 이용하여 흰색과 검은색 돌을 놓을 수 있는 간단한 바둑 프로그램
을 작성해 보자.

입력 형식

0 3 7
1 8 15
2

첫 번째 줄의 0은 검은색 돌(B), 두 번째 줄의 1은 흰색 돌(W), 세 번째 줄의 2는 프로그램의 종료를 의미한다. 0 또는 1을 입력한 이후에 입력하는 두 수는 행과 열을 의미한다. 즉, 0 3 7 입력은 검은색 돌을 3행 7열에, 1 8 15 입력은 흰색 돌을 8행 15열에 입력하는 것을 의미한다.

출력 형식

A 20x20 grid with a light orange background and dark gray lines. The letter 'B' is located at row 3, column 5. The letter 'W' is located at row 8, column 15.

0 3 7과 1 8 15를 입력하면 위와 같이 화면에 출력된다. 표 모양은 배열의 가상의 선으로 실제로는 출력되지 않는다.

```

01 #include <stdio.h>
02 int main( )
03 {
04     int i, j;
05     int color, row, col;
06     char alphago[19][19] = { 0 };
07     while (1)
08     {
09         putchar('>');
10         scanf("%d %d %d", &color, &row, &col);
11         if (color == 2) break;
12         if (color == 0) alphago[row-1][col-1] = 'B';
13         else if (color == 1) alphago[row-1][col-1] = 'W';
14         for (i = 0; i < 19; i++)
15         {
16             for (j = 0; j < 19; j++)
17             {
18                 printf("%c ", alphago[i][j]);
19             }
20             puts("");
21         }
22     }
23     return 0;
24 }

```

2차원 배열을 이용하여 선언

돌의 색에 '2'를 입력하면 무한 반복문을 빠져 나옴.

B 입력

W 입력



미션

해결하기

빛의 속도가 300,000km/s일 때 태양으로부터 각 행성까지 빛이 도달하는 시간(분)을 구하는 프로그램을 작성해 보자.

프로그램

```
01 #include <stdio.h>
02 int main( )
03 {
04     char planet[4][10] = {"mercury", "vinus", "earth", "mars"};
05     int distance[4] = {580000000, 1080000000, 1500000000, 2280000000};
06     double result[4] = { 0.0 };
07     int light = 300000;
08     int i;
09     for (i = 0; i < 4; i++)
10     {
11         result[i] = distance[i] / light;
12         printf("%s : %.0f\n" , planet[i], result[i]/60);
13     }
14     return 0;
15 }
```

실행 결과

```
mercury : 3
vinus : 6
earth : 8
mars : 13
```

더

알아보기

동적 메모리 할당 malloc

동적 메모리 할당이란 컴퓨터 프로그래밍에서 실행 시간 동안 사용할 메모리 공간을 할당하는 것을 말한다. C 언어에서는 동적으로 메모리 할당을 하기 위해 malloc 함수를 사용한다. 다음의 예제를 통해 동적 메모리 할당이 어떻게 쓰여지는지 확인해 보자.

프로그램

```
01 #include <stdio.h>
02 #include <stdlib.h>
03 int main( )
04 {
05     int sizeOfArray=0, *myArray, i, count = 0;
06     printf("배열의 원소를 몇 개 만들까요?");
07     scanf("%d", &sizeOfArray);
08     myArray = (int *)malloc(sizeof(int)*sizeOfArray);
09     for (i = 0; i < sizeOfArray; i++) myArray[i] = ++count;
10     for (i = 0; i < sizeOfArray; i++) printf("%d ", myArray[i]);
11     return 0;
12 }
```

C 언어에서 동적 메모리 등을 관리하는 함수들을 포함하고 있음.

실행 결과 예

```
배열의 원소를 몇 개 만들까요? 7 Enter
1 2 3 4 5 6 7
```

우리나라의 화폐 단위는 다음과 같다. 월급을 입력하면 화폐(또는 동전)의 개수를 최소화하여 지급하려고 한다. 월급이 1,268,439원이라고 할 때, 각 화폐(또는 동전)의 개수를 출력하는 프로그램을 작성해 보자.

입력값

50000원, 10000원, 5000원, 1000원, 500원, 100원, 50원, 10원, 5원, 1원

프로그램

```
01  #include <stdio.h>
02  int main( )
03  {
04      int i, salary;
05      int money[10] = {50000,10000,5000,1000,500,100,50,10,5,1};
06      int count[10];
07      scanf("월급: %d", &salary);
08      for (i = 0; i < 10; i++)
09      {
10          count[i] = salary / money[i];
11          salary = salary - money[i] * count[i];
12      }
13      for (i = 0; i < 10; i++)
14      {
15          printf("%5d원 : %d\n", money[i], count[i]);
16      }
17      return 0;
18  }
```

실행 결과

```
1268439 Enter
50000원: 25
10000원: 1
5000원: 1
1000원: 3
500원: 0
100원: 4
50원: 0
10원: 3
5원: 1
1원: 4
```



수행 평가

반복 구조와 배열을 이용한 채점 프로그램

활동 목표 반복 구조를 이용하여 배열의 요솟값을 초기화하고 제어할 수 있다.

다음은 어떤 올림픽 종목의 7명의 심사위원의 점수(0~100점)를 입력받아 심사위원의 최고 점수와 최저 점수를 제외한 5명의 평균 점수를 출력하는 채점 프로그램이다.

```
01  #include <stdio.h>
02  int main( )
03  {
04      int score[7];
05      int i, max=0, min=100, sum=0;
06      for (i = 0; i < 7; i++)
07      {
08          printf("%d번 심사 위원 점수 :", i + 1);
09          scanf("%d", &score[i]);
10      }
11      for (i = 0; i < 7; i++)
12      {
13          if (score[i] > max) max = score[i];
14          if (score[i] < min) min = score[i];
15      }
16      for (i = 0; i < 7; i++)
17      {
18          sum = sum + score[i];
19      }
20      printf("점수는 %f 입니다.", );
21      return 0;
22  }
```

초급 빈칸에 들어 갈 프로그램 코드를 적어 보자.

중급 만약 이 올림픽 종목의 심사위원을 10명으로 늘리고, 심사위원들은 점수를 0점부터 10점으로 조정하고, 심사위원의 최고 점수와 최저 점수를 제외한 8명의 총점으로 채점 시스템을 변경한다고 할 때, 변경 사항을 만족하는 채점 프로그램으로 수정해 보자.

스스로 평가하기

평가 항목	구분		
	그렇다	보통이다	그렇지 않다
• 배열의 구조를 이해하여 값을 저장할 수 있다.	<input type="text"/>	<input type="text"/>	<input type="text"/>
• 배열과 제어문을 이용하여 프로그램을 작성할 수 있다.	<input type="text"/>	<input type="text"/>	<input type="text"/>



내 실력 확인하기

내용을 이해했나요?

- **배열**: 메모리의 연속적인 공간에 같은 구조의 데이터를 여러 개 저장할 수 있는 데이터 저장 장소를 말한다.
- **1차원 배열**: 하나의 행 또는 열로만 이루어진 배열로, 하나의 기준으로 나열되는 자료를 관리할 때 사용한다.
- **2차원 배열**: 1차원 배열이 여러 개 필요할 때 사용하며, 메모리 공간에 연속적으로 저장된 데이터를 인간이 이해하기 쉽게 행과 열의 논리적인 2차원 공간으로 해석하여 사용할 수 있기 때문에 유용하게 사용된다.

문제로 확인할까요?

1. 프로그램이 실행될 때 필요한 기억 장소로서, 프로그램이 실행되는 동안 그 값이 변할 수 있는 것은?

- ① 상수 ② 배열 ③ 변수 ④ 데이터 ⑤ 수식

2. 동일한 데이터 형을 가진 변수들의 집합을 무엇이라고 하는가?

- ① 품 ② 상수 ③ 배열 ④ 모듈 ⑤ 프로시저

3. 다음과 같은 프로그램을 실행했을 때 실행 결과를 작성해 보자.

프로그램

```
01  #include <stdio.h>
02  int main( ) {
03      int i, s=0, a[ ]={ 2, 5, 7, 9, 15 };
04      for(i=3; i<5; i++)
05          s += a[i];
06      printf( "s = %d\n ", s );
07      return 0;
08  }
```

실행 결과

평가해 볼까요?

★ 다음 평가 항목에 따라 자신의 성취 척도를 스스로 점검해 보자.

영역	평가 항목	척도				
		1	2	3	4	5
이해	배열의 개념과 구조를 이해하고, 배열의 선언과 초기화 방법을 설명할 수 있는가?					
적용	1차원, 2차원 배열을 사용한 프로그램을 작성할 수 있는가?					