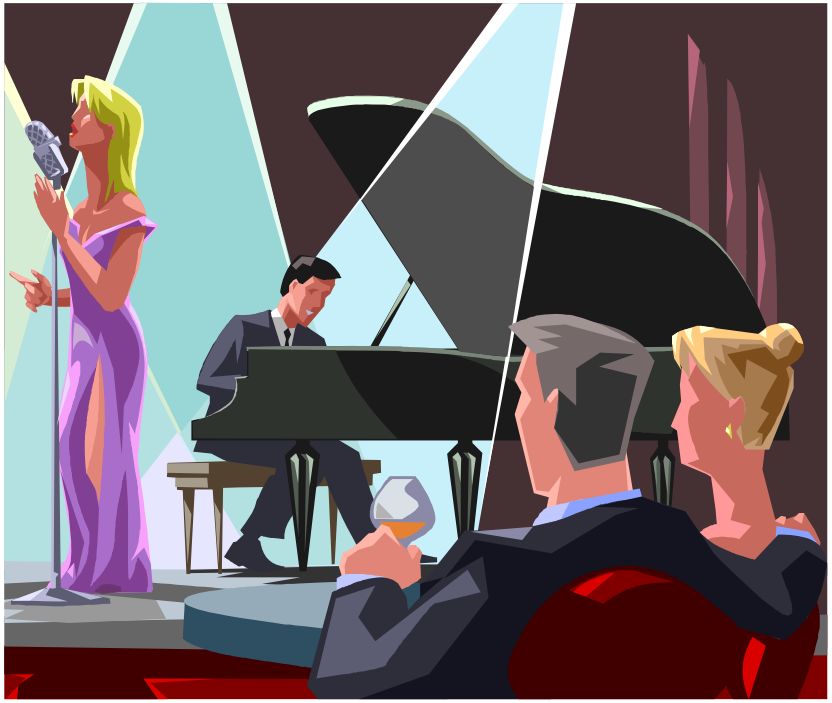
누구나 즐기는 C언어 콘서트

제8장 배열

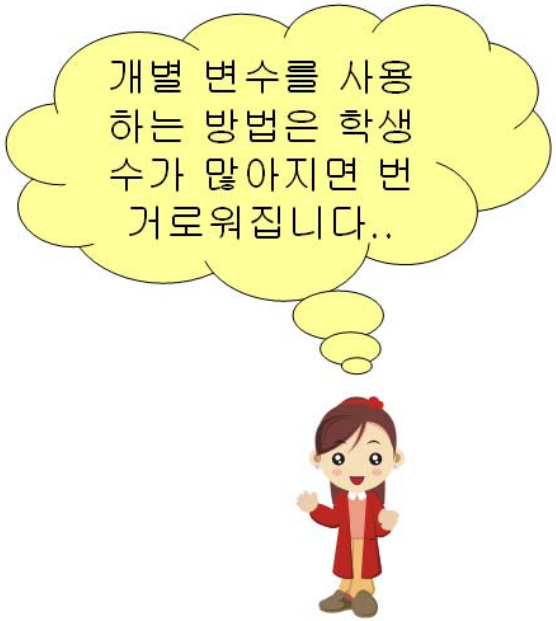


이번 장에서 학습할 내용

- 반복의 개념 이해
- 배열의 개념
- 배열의 선언과 초기화
- 일차원 배열
- 다차원 배열

배열의 필요성

- 학생이 10명이 있고 이들의 평균 성적을 계산한다고 가정하자.



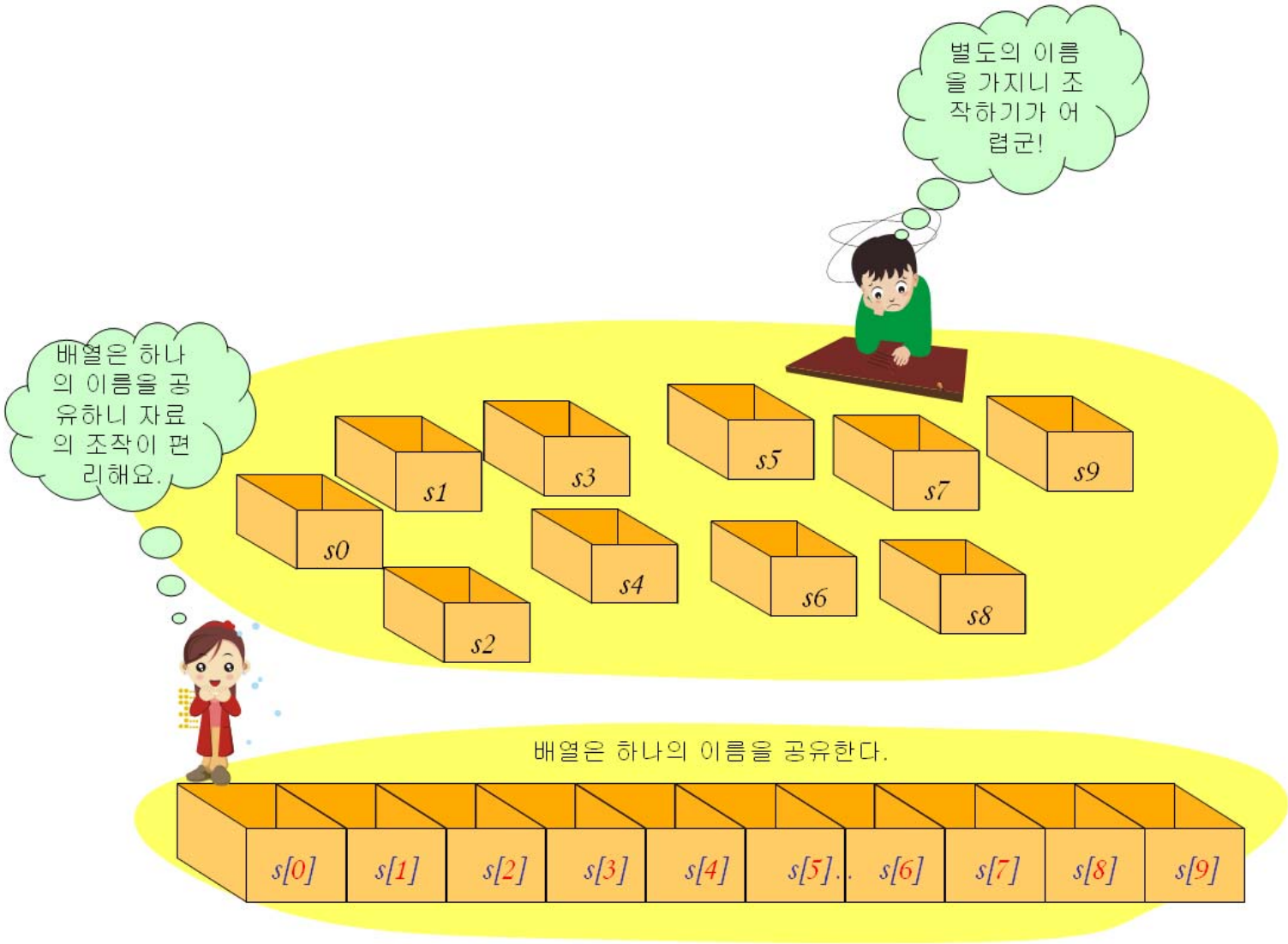
방법 #1: 개별 변수 사용

```
int s0;
int s1;
  ...
int s9;
```

방법 #2: 배열 사용

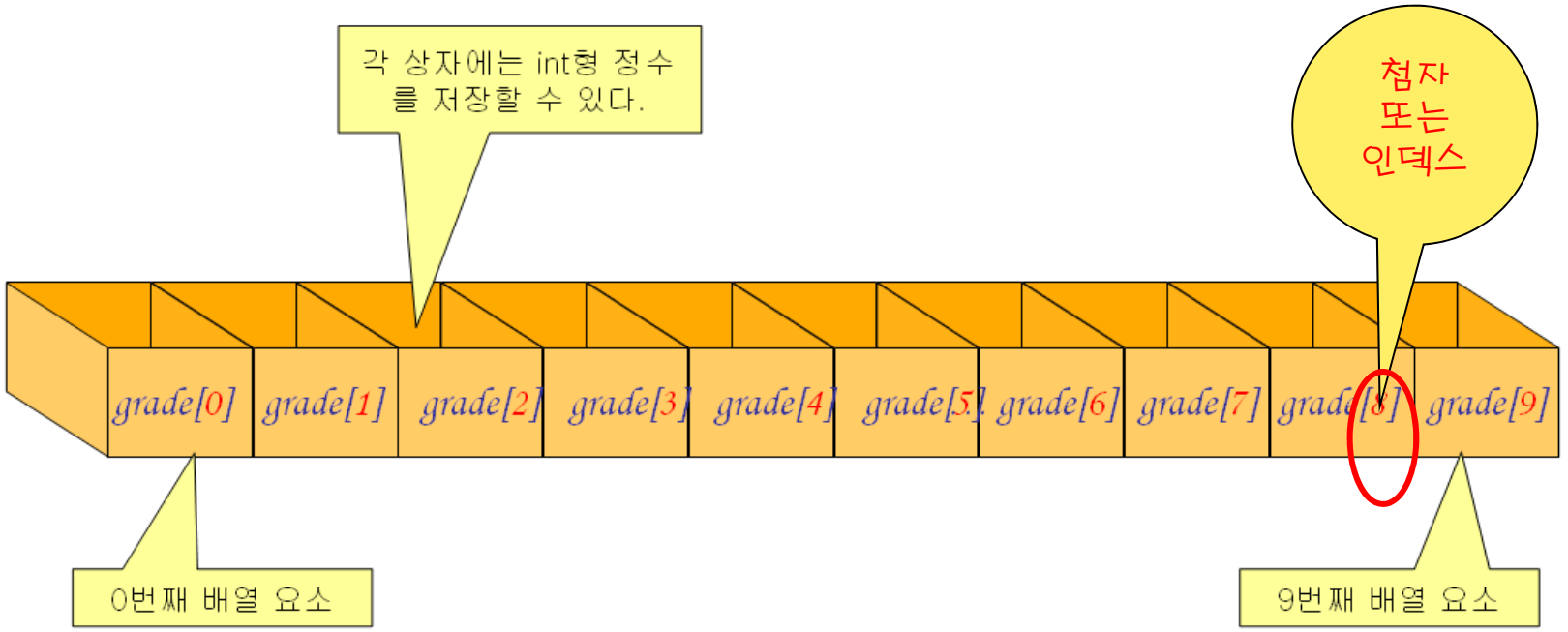
```
int s[10];
```

배열의 필요성



배열이란?

- 배열(array): 동일한 타입의 데이터가 여러 개 저장되어 있는 데이터 저장 장소
- 배열 안에 들어있는 각각의 데이터들은 정수로 되어 있는 번호(인덱스)에 의하여 접근
- 배열을 이용하면 여러 개의 값을 하나의 이름으로 처리할 수 있다.

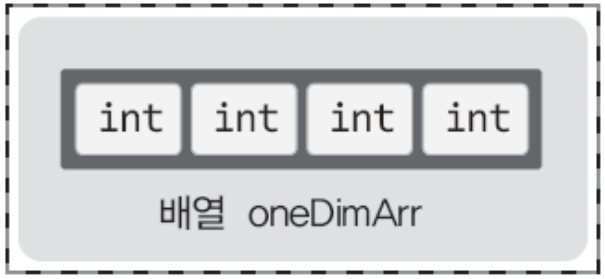


배열의 선언

```
int oneDimArr[4];
```

1차원 배열 선언의 예

int	배열을 이루는 요소(변수)의 자료형
oneDimArr	배열의 이름
[4]	배열의 길이



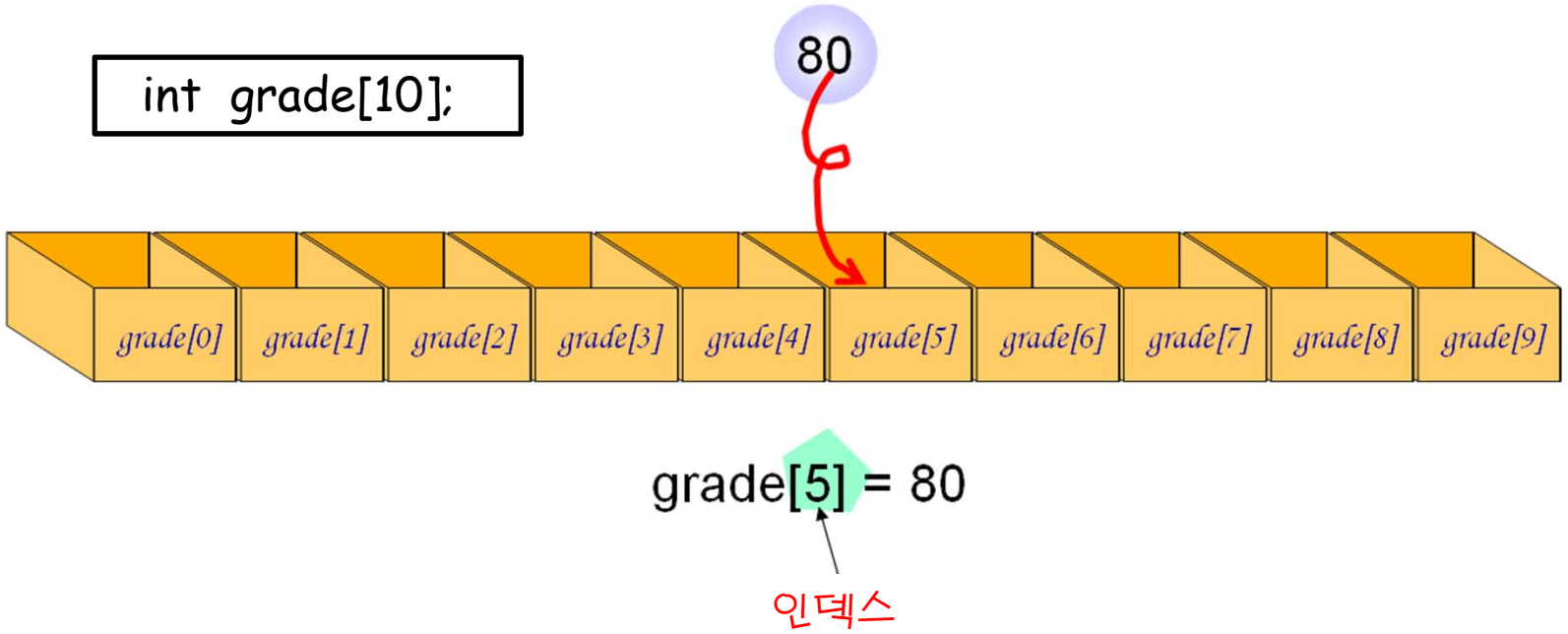
생성되는 배열의 형태

- 자료형: 배열 원소들이 int형라는 것을 의미
- 배열 이름: 배열을 사용할 때 사용하는 이름이 grade
- 배열 크기: 배열 원소의 개수가 10개
- 인덱스(첨자)는 항상 0부터 시작한다.

```
int  score[60];          // 60개의 int형 값을 가지는 배열 grade
float  cost[12];         // 12개의 float형 값을 가지는 배열 cost
char  name[50];          // 50개의 char형 값을 가지는 배열 name
char  src[10], dst[10];  // 2개의 문자형 배열을 동시에 선언
int  index, days[7];     // 일반 변수와 배열을 동시에 선언
```

배열 원소 접근

```
int grade[10];
```




```
(예)
grade[0] = 80;              // 0번째 원소에 80을 대입
grade[1] = grade[0];        // 0번째 원소를 1번째 원소로 복사
grade[i] = 100;             // i는 정수 변수
grade[i+2] = 100;           // 수식이 인덱스가 된다.
```

배열 선언 예제

```
#include <stdio.h>

int main(void)
{
	int i;
	int grade[5];

	grade[0] = 10;
	grade[1] = 20;
	grade[2] = 30;
	grade[3] = 40;
	grade[4] = 50;

	for(i=0;i < 5; i++)
		printf("grade[%d]=%d\n",i, grade[i]);
	return 0;
}
```

```
grade[0]=10
grade[1]=20
grade[2]=30
grade[3]=40
grade[4]=50
```

배열 선언 예제

```
#include <stdio.h>

int main(void)
{
      int i, j;
      int grade[5];

      for(i=0, j=10 ; i<5 ; i++, j=+10)
          grade[i] = j;

      for(i=0;i < 5; i++)
              printf("grade[%d]=%d\n",i, grade[i]);
      return 0;
}
```

```
grade[0]=10
grade[1]=20
grade[2]=30
grade[3]=40
grade[4]=50
```

배열 원소 참조 예제

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 10

int main(void)
{
        int grade[SIZE];
        int i;

        for(i = 0; i < SIZE; i++)
                grade[i] = rand() % 100;

        printf("====================\n");
        printf("인덱스      값\n");
        printf("====================\n");

        for(i = 0; i < SIZE; i++)
                printf("%5d    %5d\n", i, grade[i]);
        return 0;
}
```

```
====================
인덱스      값
====================
    0     41
    1     67
    2     34
    3      0
    4     69
    5     24
    6     78
    7     58
    8     62
    9     64
```

배열 선언 예제

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 5

int main(void)
{
        int i;
        int grade[SIZE];
        printf("5명의 점수를 입력하시오\n");

        for(i = 0; i < SIZE; i++)
                scanf("%d", &grade[i]);

        for(i = 0; i < SIZE; i++)
                printf("grade[%d]=%d\n", i, grade[i]);
        return 0;
}
```

```
grade[0]=41
grade[1]=67
grade[2]=34
grade[3]=0
grade[4]=69
```

배열 선언 예제

```
#include <stdio.h>
#define STUDENTS 5
int main(void)
{
        int grade[STUDENTS];
        int sum = 0;
        int i, average;
        for(i = 0; i < STUDENTS; i++)
        {
                printf("학생들의 성적을 입력하시오: ");
                scanf("%d", &grade[i]);
        }

        for(i = 0; i < STUDENTS; i++)
                sum += grade[i];
        average = sum / STUDENTS;
        printf("성적 평균= %d\n", average);
        return 0;
}
```

학생들의 성적을 입력하시오: 10
학생들의 성적을 입력하시오: 20
학생들의 성적을 입력하시오: 30
학생들의 성적을 입력하시오: 40
학생들의 성적을 입력하시오: 50
성적 평균 = 30

만약 이 프로그램에서 다음과 같이 입력한다면?

학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 6
성적 평균 = ?????

배열 선언 예제(Fix It!)

```
#include <stdio.h>
#define STUDENTS 5
int main(void)
{
        int grade[STUDENTS];
        double sum = 0, average= 0;
        int i;
        for(i = 0; i < STUDENTS; i++)
        {
                printf("학생들의 성적을 입력하시오: ");
                scanf("%d", &grade[i]);
        }

        for(i = 0; i < STUDENTS; i++)
                sum += grade[i];
        average = sum / STUDENTS;
        printf("성적 평균= %f\n", average);

        return 0;
}
```

```
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 6
성적 평균 = 5.2
```

만약 평균을 구하는 부분을 함수로 만든다면?

배열 선언 예제(Function It!)

```
#include <stdio.h>
#define STUDENTS 5
double getAverage(int grade[], int i )

int main(void)
{
    int grade[STUDENTS];
    int i;
    for(i = 0; i < STUDENTS; i++)
    {
        printf("학생들의 성적을 입력하시오: ");
        scanf("%d", &grade[i]);
    }
    printf("성적 평균= %f\n", getAverage( grade, STUDENTS) );
   return 0;
}

double getAverage(int grade[], int i )

{

    double average = 0, sum = 0;
       for (i = 0; i < STUDENTS; i++)
           sum += grade[i];
        average = sum / STUDENTS;
       return average;
}
```

```
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 5
학생들의 성적을 입력하시오: 6
성적 평균 = 5.2
```

잘못된 인덱스 문제

- 인덱스가 배열의 크기를 벗어나게 되면 프로그램에 치명적인 오류를 발생시킨다.
- C에서는 프로그래머가 인덱스가 범위를 벗어나지 않았는지를 확인하고 책임을 져야 한다.

```
int   grade[5];
...
grade[5] = 60;       // 치명적인 오류!
```

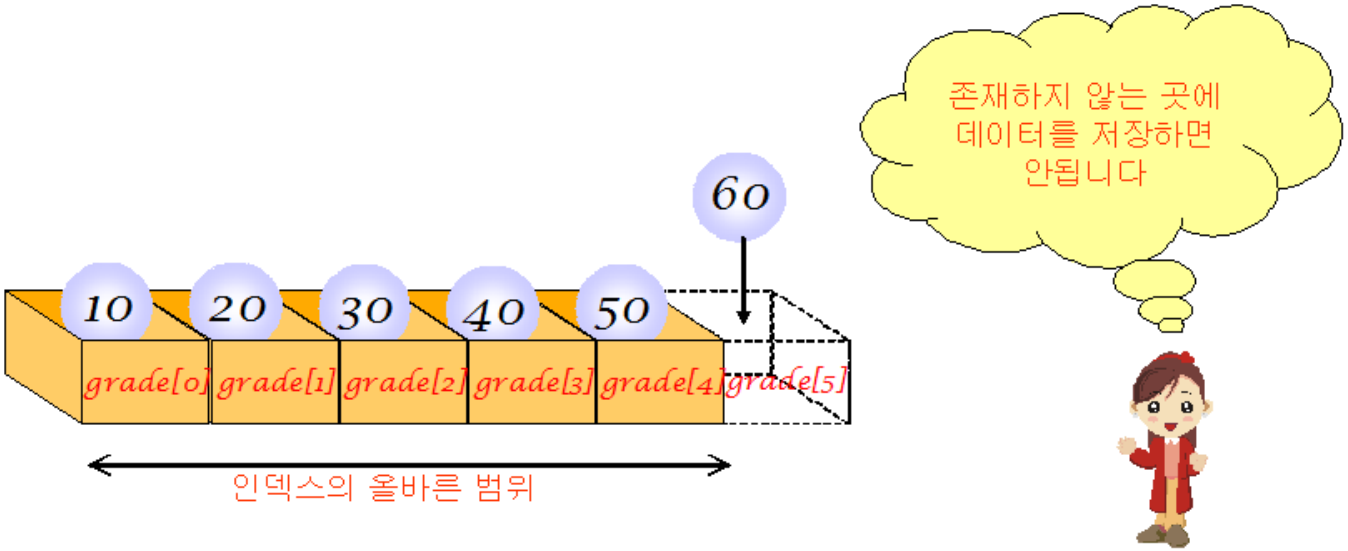


그림 8.4 잘못된 인덱스

잘못된 인덱스 예제

```
#include <stdio.h>
int main(void)
{
        int grade[5];
        int i;

        grade[0]=10;
        grade[1]=20;
        grade[2]=30;
        grade[3]=40;
        grade[4]=50;
        grade[5]=60;


        for(i = 0; i <= 5; i++)
                printf("grade[%d]=%d\n", i, grade[i]);
        return 0;
}
```

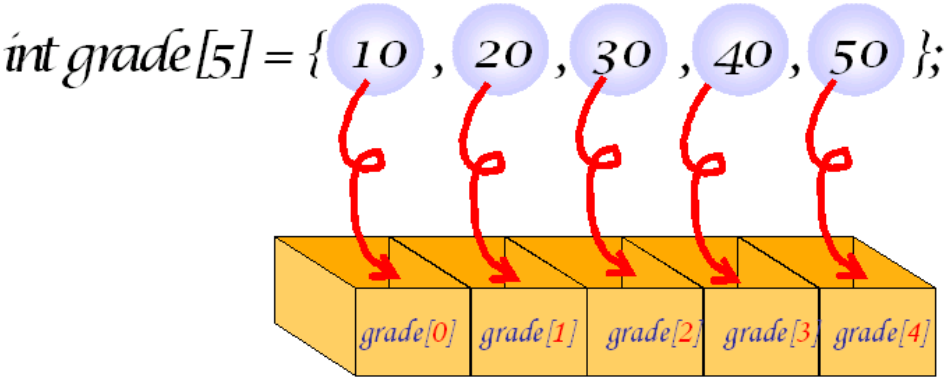
시스템에 심각한 오류가 발생할 수도 있다.

중간 점검

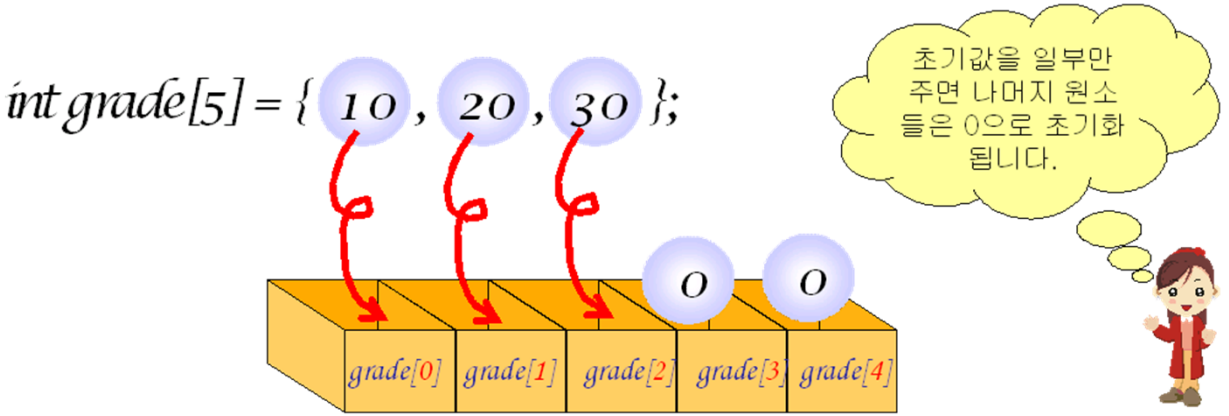
1. n개의 원소를 가지는 배열의 경우, 첫 번째 원소의 인덱스는 무엇인가?
2. n개의 원소를 가지는 배열의 경우, 마지막 원소의 인덱스는 무엇인가?
3. 범위를 벗어나는 인덱스를 사용하면 어떻게 되는가? 즉 int a[10];과 같이 선언된 배열이 있는 경우, a[10]에 6을 대입하면 어떻게 되는가?

배열의 초기화

- int grade[5] = { 10,20,30,40,50 };

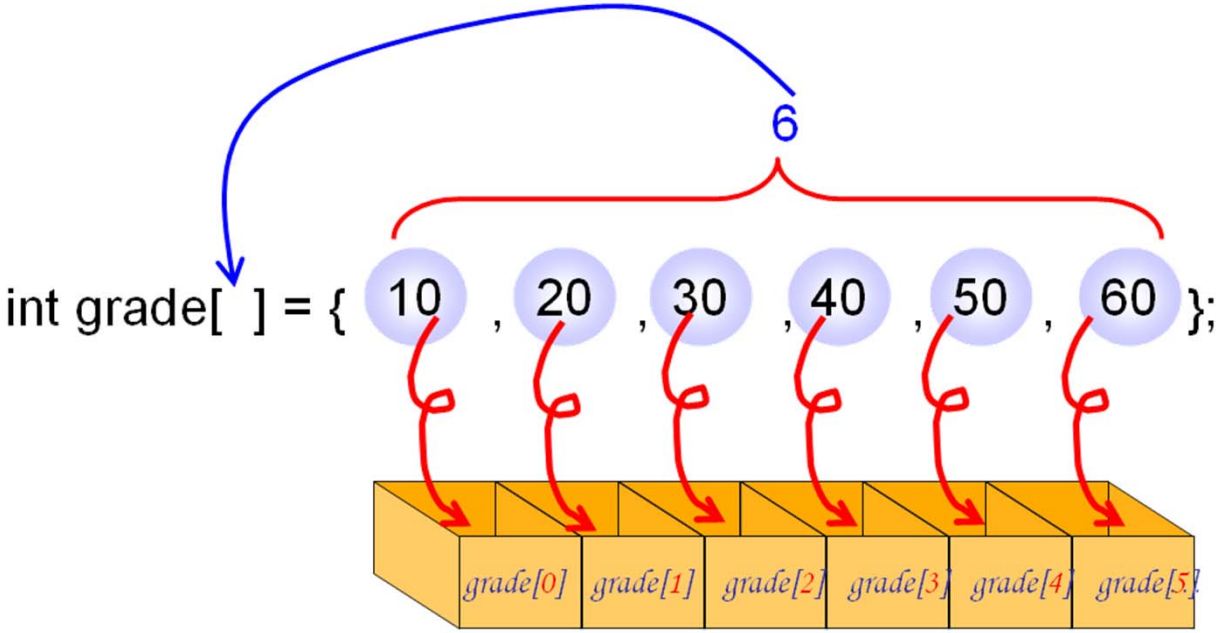


- int grade[5] = { 10,20,30 };



배열의 초기화

- 배열의 크기가 주어지지 않으면 자동적으로 초기값의 개수만큼이 배열의 크기로 잡힌다.



배열 초기화 예제

```
#include <stdio.h>
int main(void)
{
    int grade[5] = { 31, 63, 62, 87, 14 };
    int i;

    for(i = 0; i < 5; i++)
        printf("grade[%d] = %d\n", i, grade[i]);

    return 0;
}
```

```
grade[0] = 31
grade[1] = 63
grade[2] = 62
grade[3] = 87
grade[4] = 14
```

배열 초기화 예제

```
#include <stdio.h>
int main(void)
{
    int grade[5] = { 31, 63 };
    int i;

    for(i = 0; i < 5; i++)
        printf("grade[%d] = %d\n", i, grade[i]);

    return 0;
}
```

```
grade[0] = 31
grade[1] = 63
grade[2] = 0
grade[3] = 0
grade[4] = 0
```

배열 초기화 예제

```
#include <stdio.h>
int main(void)
{
    int grade[5] ;
    int i;

    for(i = 0; i < 5; i++)
        printf("grade[%d] = %d\n", i, grade[i]);

    return 0;
}
```

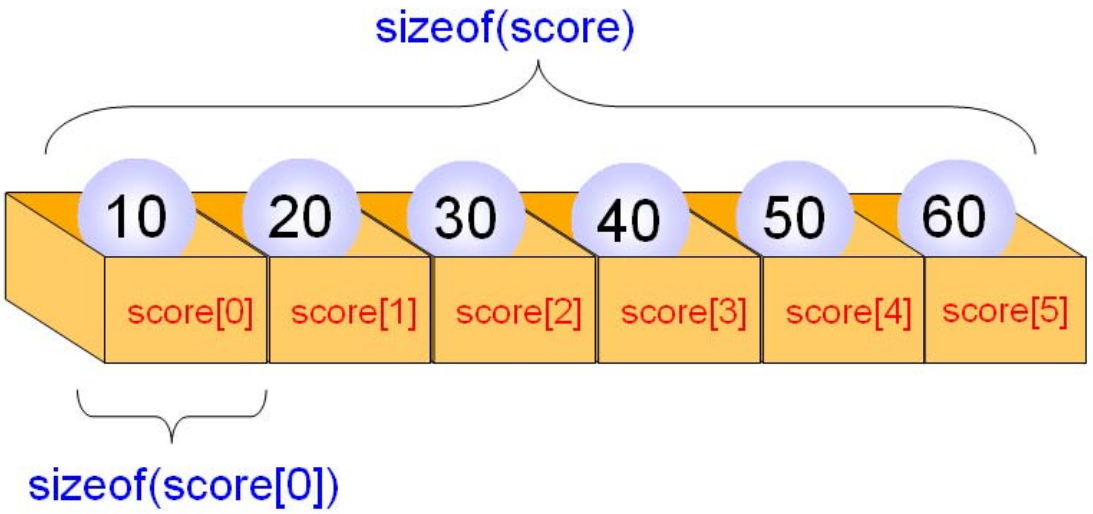
```
grade[0]=4206620
grade[1]=0
grade[2]=4206636
grade[3]=2018779649
grade[4]=1
```

잘못된 인덱스로 접근하는 경우

```
#include <stdio.h>
#define SIZE 5

int main(void)
{
	int array[SIZE] = {1, 2, 3, 4, 5};
	int i;

	for(i = 0; i <= SIZE; i++)
		printf("array[%d]   %d\n", i, array[i]);

	return 0;
}
```

```
array[0]      1
array[1]      2
array[2]      3
array[3]      4
array[4]      5
array[5]      1245120
```

배열 원소의 개수 계산




```
int grade[] = { 1, 2, 3, 4, 5, 6 };
int i, size;

size = sizeof(grade) / sizeof(grade[0]);

for(i = 0; i < size ; i++)
        printf("%d ", grade[i]);
```

배열 원소 개수 자동 계산

1차원 배열의 선언, 초기화 및 접근 관련 예제


```c
int main(void)
{
    int arr1[5]={1, 2, 3, 4, 5};
    int arr2[ ]={1, 2, 3, 4, 5, 6, 7};
    int arr3[5]={1, 2};
    int ar1Len, ar2Len, ar3Len, i;

    printf("배열 arr1의 크기: %d \n", sizeof(arr1));
    printf("배열 arr2의 크기: %d \n", sizeof(arr2));
    printf("배열 arr3의 크기: %d \n", sizeof(arr3));

    ar1Len = sizeof(arr1) / sizeof(int);   // 배열 arr1의 길이 계산
    ar2Len = sizeof(arr2) / sizeof(int);   // 배열 arr2의 길이 계산
    ar3Len = sizeof(arr3) / sizeof(int);   // 배열 arr3의 길이 계산

    for(i=0; i<ar1Len; i++)
        printf("%d ", arr1[i]);
    printf("\n");

    for(i=0; i<ar2Len; i++)
        printf("%d ", arr2[i]);
    printf("\n");

    for(i=0; i<ar3Len; i++)
        printf("%d ", arr3[i]);
    printf("\n");
    return 0;
}
```

sizeof 연산의 결과로
배열의 바이트 크기정보 반환

배열의 길이를 계산하는
방식에 주목!

배열이기에 for문을 통한
순차적 접근이 가능하다.

다수의 변수라면 반복문을 통한
순차적 접근 불가능!

실행결과

```
배열 arr1의 크기: 20
배열 arr2의 크기: 28
배열 arr3의 크기: 20
1 2 3 4 5
1 2 3 4 5 6 7
1 2 0 0 0
```

배열의 활용

배열의 복사

```
int grade[SIZE];
int score[SIZE];

score = grade;              // 컴파일 오류!
```

잘못된 방법

```
#include <stdio.h>
#define SIZE 5

int main(void)
{
        int i;
        int a[SIZE] = {1, 2, 3, 4, 5};
        int b[SIZE];

        for(i = 0; i < SIZE; i++)
                b[i] = a[i];

        return 0;
}
```

올바른 방법

배열의 비교

```c
#include <stdio.h>
#define SIZE 5

int main(void)
{
	int i;
	int a[SIZE] = { 1, 2, 3, 4, 5 };
	int b[SIZE] = { 1, 2, 3, 4, 5 };

	if( a == b )		// ① 올바르지 않은 배열 비교
		printf("잘못된 결과입니다.\n");
	else
		printf("잘못된 결과입니다.\n");

	for(i = 0; i < SIZE ; i++) // ② 올바른 배열 비교
	{
		if ( a[i] != b[i] )
		{
			printf("a[]와 b[]는 같지 않습니다.\n");
			return 0;
		}
	}
	printf("a[]와 b[]는 같습니다.\n");
	return 0;
}
```

배열 원소 역순 출력

```c
#include <stdio.h>
#define SIZE 5

int main(void)
{
	int data[SIZE];
	int i;

	for(i = 0; i < SIZE; i++)		// 정수를 입력받는 루프
	{
		printf("정수를 입력하시오:");
		scanf("%d", &data[i]);
	}

	for(i = SIZE - 1;i >= 0; i--)	// 역순으로 출력하는 루프
	{
		printf("%d\n", data[i]);
	}
	return 0;
}
```

```
정수를 입력하시오:10
정수를 입력하시오:20
정수를 입력하시오:30
정수를 입력하시오:40
정수를 입력하시오:50
50
40
30
20
10
```

예제

```
#include <stdio.h>
#define STUDENTS 5

int main(void)
{
	int grade[STUDENTS] = { 30, 20, 10, 40, 50 };
	int i, s;

	for(i = 0; i < STUDENTS; i++)
	{
		printf("번호 %d: ", i);
		for(s = 0; s < grade[i]; s++)
			printf("*");
		printf("\n");
	}

	return 0;
}
```

```
번호 0: ******************************
번호 1: ********************
번호 2: **********
번호 3: ****************************************
번호 4: **************************************************
```

최소값 탐색

```c
#include <stdio.h>
#define SIZE 5

int main(void)
{
	int grade[SIZE];
	int i, min;

	for(i = 0; i < SIZE; i++)
	{
		printf("성적을 입력하시오: ");
		scanf("%d", &grade[i]);
	}

	min = grade[0];

	for(i = 1; i < SIZE; i++)
	{
		if( grade[i] < min )
			min = grade[i];
	}

	printf("최소값은 %d입니다.\n", min);
	return 0;
}
```

```
숫자를 입력하시오: 50
숫자를 입력하시오: 40
숫자를 입력하시오: 30
숫자를 입력하시오: 20
숫자를 입력하시오: 10
숫자를 입력하시오: 20
숫자를 입력하시오: 30
숫자를 입력하시오: 40
숫자를 입력하시오: 60
숫자를 입력하시오: 70
최소값은 10입니다.
```

빈도 계산

```c
#include <stdio.h>
#define SIZE 101

int main(void)
{
	int freq[SIZE];
	int i, score;

	for(i = 0; i < SIZE; i++)
		freq[i] = 0;

	while(1)
	{
		printf("숫자를 입력하시오(종료-1): ");
		scanf("%d", &score);
		if (score < 0) break;
		freq[score]++;
	}

	printf("값   빈도\n");

	for(i = 0; i < SIZE; i++)
		printf("%3d    %3d \n", i, freq[i]);

	return 0;
}
```

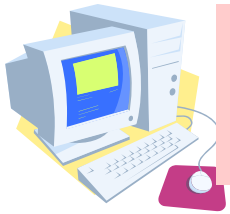
```
숫자를 입력하시오(종료 -1): 0
숫자를 입력하시오(종료 -1): 1
숫자를 입력하시오(종료 -1): 99
숫자를 입력하시오(종료 -1): 100
숫자를 입력하시오(종료 -1): 100
숫자를 입력하시오(종료 -1): -1
값    빈도
  0     1
  1     1
  2     0
 ...
 98     0
 99     1
100     2
```

주사위면 빈도 계산

```c
#include <stdio.h>
#include <stdlib.h>

#define SIZE 6

int main(void)
{
        int freq[SIZE] = { 0 };              // 주사위의 면의 빈도를 0으로 한다.
        int i;

        for(i = 0; i < 10000; i++)          // 주사위를 10000번 던진다.
                ++freq[ rand() % 6 ];        // 해당면의 빈도를 하나 증가한다.

        printf("====================\n");
        printf("면      빈도\n");
        printf("====================\n");

        for(i = 0; i < SIZE; i++)
                printf("%3d     %3d \n", i, freq[i]);

        return 0;
}
```

```
====================
면    빈도
====================
 0    1657
 1    1679
 2    1656
 3    1694
 4    1652
 5    1662
```

함수에서의 배열 전달



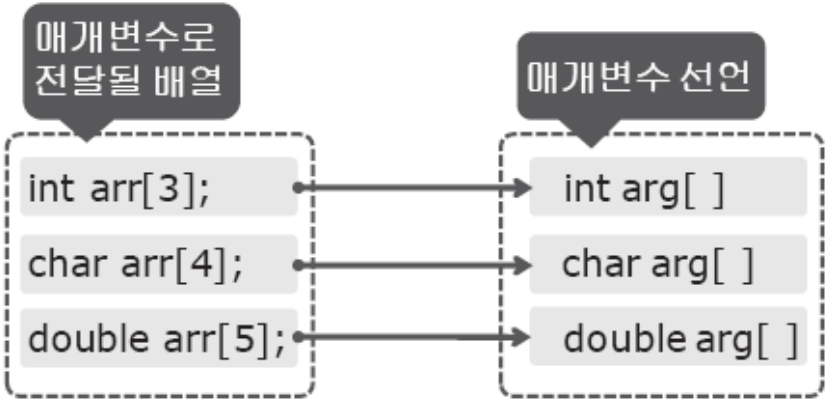
■ 배열 전달의 기본 원리

"배열을 매개변수로 선언하는 방법은 존재하지 않는다."

"대신 배열의 주소 값을 전달한다!"

■ 예제 20-1.c

```
1.  #include <stdio.h>
2.  void ArrPrintf(int arg[]);
3.
4.  int main(void)
5.  {
6.      int arr[3]={1, 2, 3};
7.      ArrPrintf(arr);
8.      return 0;
9.  }
10.
11. void ArrPrintf(int arg[])
12. {
13.     int i;
14.     for(i=0; i<3; i++)
15.         printf("%4d", arg[i]);
16.     printf("₩n");
17. }
```




```
   1   2   3
```

■ 배열의 내용이 바뀔 수도 있으니 주의하세요

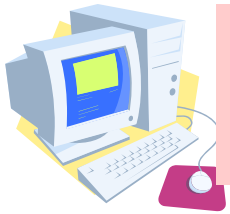
"매개변수에 배열의 주소 값이 저장되므로 매개변수를 통해서 배열에 직접 접근이 가능하다."

잠시 배열 접근을 위한 주소 값 계산과정을 생각해 보면 이해에 도움이 됩니다.

■ 예제 20-2.c

```
1.  #include <stdio.h>
2.  void IncreArrElm(int arg[]);
3.  void ArrPrintf(int arg[]);
4.
5.  int main(void)
6.  {
7.      int arr[3]={1, 2, 3};
8.      printf("*** 증가 이전 배열정보 출력 *** ₩n");
9.      ArrPrintf(arr);
10.
11.     IncreArrElm(arr);
12.     printf("*** 증가 이후 배열정보 출력 *** ₩n");
13.     ArrPrintf(arr);
14.     return 0;
15. }
16.
17. void IncreArrElm(int arg[])
18. {
19.     int i;
20.     for(i=0; i<3; i++)
21.         arg[i]=arg[i]+1;
22. }
23.
24. void ArrPrintf(int arg[])
25. {
26.     int i;
27.     for(i=0; i<3; i++)
28.         printf("%4d", arg[i]);
29.     printf("₩n");
30. }
```

```
*** 증가 이전 배열정보 출력 ***
   1   2   3
*** 증가 이후 배열정보 출력 ***
   2   3   4
```



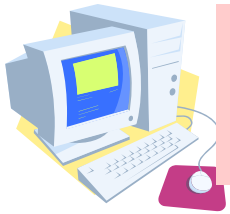
■ 배열의 매개변수 선언에 const를 붙이면 변경 불가능

```
int AddFunction(const int n1, const int n2)
{
    // n1++;  // 컴파일 오류 발생 위치
    // n2++;  // 컴파일 오류 발생 위치
    return n1+n2;
}
```

n1과 n2에 저장된 값의 변경 허용 않겠다!

```
void ArrPrintf(const int arg[])
{
    int i;
    for(i=0; i<3; i++)
        printf("%4d", arg[i]);

    // arg[i]=arg[i]+1; // 실수로 삽입된 코드라고 가정합시다!
    printf("₩n");
}
```

매개변수 arg가 참조하는 배열의 변경을 허용 않겠다!



■ 함수 내에서 배열의 길이를 계산할 수 있는가?

```
void IncreArrElm(int arg[])
{
    int i;
    int len=sizeof(arg)/sizeof(int);

    for(i=0; i<len; i++)
        arg[i]=arg[i]+1;
}
```

매개변수는 매개변수일 뿐 배열의 이름이 아니다!

■ 예제 20-5.c

```
1.  #include <stdio.h>
2.  void IncreArrElm(int arg[], const int len);
3.  void ArrPrintf(const int arg[], const int len);
4.
5.  int main(void)
6.  {
7.      int arr1[3]={1, 2, 3};
8.      int arr2[5]={1, 2, 3, 4, 5};
9.
10.     printf("*** 증가 이전 배열정보 출력 *** ₩n");
11.     ArrPrintf(arr1, sizeof(arr1)/sizeof(int));
12.     ArrPrintf(arr2, sizeof(arr2)/sizeof(int));
13.
14.     IncreArrElm(arr1, sizeof(arr1)/sizeof(int));
15.     IncreArrElm(arr2, sizeof(arr2)/sizeof(int));
16.
17.     printf("*** 증가 이후 배열정보 출력 *** ₩n");
18.     ArrPrintf(arr1, sizeof(arr1)/sizeof(int));
19.     ArrPrintf(arr2, sizeof(arr2)/sizeof(int));
20.     return 0;
21. }
22.
23. void IncreArrElm(int arg[], const int len)
24. {
25.     int i;
26.     for(i=0; i<len; i++)
27.         arg[i]=arg[i]+1;
28. }
```

배열과 함수

```
#include <stdio.h>
#define STUDENTS 5
int get_average(int score[], int n);     // ①

int main(void)
{
        int grade[STUDENTS] = { 1, 2, 3, 4, 5 };
        int avg;

        avg = get_average(grade, STUDENTS);
        printf("평균은 %d입니다.\n", avg);
        return 0;
}

int get_average(int score[], int n)      // ②
{
        int i;
        int sum = 0;

        for(i = 0; i < n; i++)
                sum += score[i];
        return sum / n;
}
```

배열이 인수인 경우,
참조에 의한 호출

배열의 원본이
score[]로 전달

배열이 함수의 인수인 경우 1/2

```
#include <stdio.h>
#define SIZE 7

void square_array(int a[], int size);
void print_array(int a[], int size);
void square_element(int e);

int main(void)
{
	int list[SIZE] = { 1, 2, 3, 4, 5, 6, 7 } ;

	print_array(list, SIZE);
	square_array(list, SIZE);  // 배열은 원본이 전달된다.
	print_array(list, SIZE);

	printf("%3d\n", list[6]);
	square_element(list[6]);   // 배열 요소는 복사본이 전달된다.
	printf("%3d\n", list[6]);

	return 0;
}
```

배열이 함수의 인수인 경우 2/2

```
void square_array(int a[], int size)
{
        int i;

        for(i = 0; i < size; i++)
                a[i] = a[i] * a[i];
}
void square_element(int e)
{
        e = e * e;
}
void print_array(int a[], int size)
{
        int i;

        for(i = 0; i < size; i++)
                printf("%3d ", a[i]);
        printf("\n");
}
```

배열 원소의 사본이 e로 전달

배열의 원본이 a[]로 전달

```
  1  2  3  4  5  6  7
  1  4  9 16 25 36 49
  7
  7
```

원본 배열의 변경을 금지하는 방법

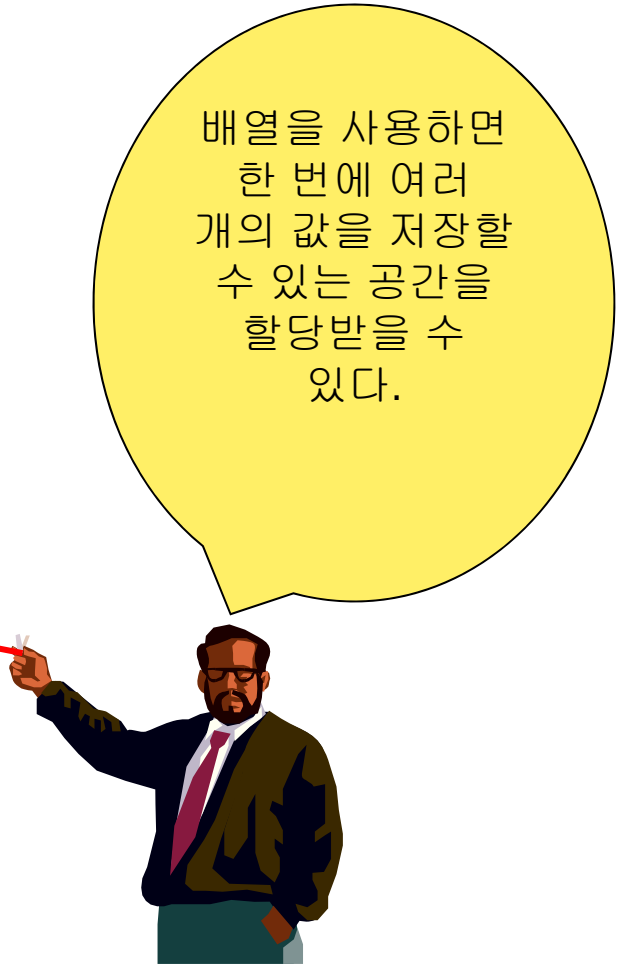
```
#include <stdio.h>
#define SIZE 20

void copy_array(char dest[], const char src[], int count);

int main(void)
{
	char s[SIZE] = { 'H', 'E', 'L', 'L', 'O', '\0' };
	char d[SIZE];

	copy_array(d, s, SIZE);
	printf("%s\n", d);
	return 0;
}
void copy_array(char dest[], const char src[], int size)
{
	int i;
	for(i = 0; i < size; i++)
	{
		dest[i] = src[i];
	}
}
```

배열 원본의 변경 금지

```
HELLO
```

이번 장에서 학습할 내용

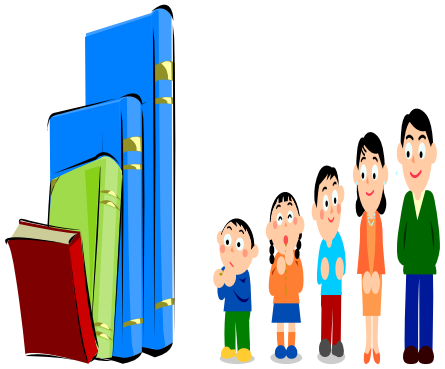
- 반복의 개념 이해
- 배열의 개념
- 배열의 선언과 초기화
- 일차원 배열
- 배열의 응용
- 정렬과 탐색
- 다차원 배열



정렬이란?

- 정렬은 물건을 크기순으로 오름차순이나 내림차순으로 나열하는 것
- 정렬은 컴퓨터 공학분야에서 가장 기본적이고 중요한 알고리즘중의 하나
- 정렬은 자료 탐색에 있어서 필수적이다.

(예) 만약 사전에서 단어들이 정렬이 안되어 있다면?



비교	제조사	모델명	요약설명	최저가↓	업체수	출시
☐	ROLLEI	D-41com	410만화소(0.56")/1.8"LCD/3배줌/연사/CF카드	320,000	4	02년
☐	카시오	QV-R40	413만화소(0.56")/1.6"LCD/3배줌/동영상/히스토그램/앨범기능/SD,MMC카드	344,000	73	03년
☐	파나소닉	DMC-LC43	423만화소(0.4")/1.5"LCD/3배줌/동영상+녹음/연사/SD,MMC카드	348,000	36	03년
☐	현대	DC-4311	400만화소(0.56")/1.6"LCD/3배줌/동영상/SD,MMC카드	350,000	7	03년
☐	삼성테크윈	Digimax420	410만화소(0.56")/1.5"LCD/3배줌/동영상+녹음/음성메모/한글/SD카드	353,000	47	03년
☐	니콘	Coolpix4300	413만화소(0.56")/1.5"LCD/3배줌/동영상/연사/CF카드HOT4	356,800	79	02년
☐	올림푸스	뮤-20 Digital	423만화소(0.4")/1.5"LCD/3배줌/동영상/연사/생활방수/xD카드	359,000	63	03년
☐	코닥	LS-443(Dock포함)	420만화소/1.8"LCD/3배줌/동영상+녹음/SD,MMC카드/Dock시스템	365,000	39	02년
☐	올림푸스	C-450Z	423만화소(0.4")/1.8"LCD/3배줌/동영상/연사/xD카드	366,000	98	03년
☐	올림푸스	X-1	430만화소/1.5"LCD/3배줌/동영상/연사/xD카드	367,000	19	03년
☐	미놀타	DiMAGE-F100	413만화소(0.56")/1.5"LCD/3배줌/동영상+녹음/음성메모/동체추적AF/연사/SD,MMC카드	373,000	18	02년
☐	삼성테크윈	Digimax410	410만화소(0.56")/1.6"LCD/3배줌/동영상+녹음/음성메모/한글/CF카드	374,000	4	02년

선택정렬(selection sort)

- 선택정렬(selection sort): 정렬이 안된 숫자들중에서 최소값을 선택하여 배열의 첫번째 요소와 교환

선택 정렬 1/2

```
#include <stdio.h>
#define SIZE 10

int main(void)
{
	int list[SIZE] = { 3, 2, 9, 7, 1, 4, 8, 0, 6, 5 };

	int i, j, temp, least;
```

선택 정렬 2/2

```
    for(i = 0; i < SIZE-1; i++)
    {
        least = i;   // i번째 값을 최소값으로 가정

        for(j = i + 1; j < SIZE; j++)    // 최소값 탐색
            if(list[j] < list[least])
                least = j;
        // i번째 원소와 least 위치의 원소를 교환
        temp = list[i];
        list[i] = list[least];
        list[least] = temp;
    }
    for(i = 0;i < SIZE; i++)
        printf("%d ", list[i]);
    printf("\n");
    return 0;
}
```

```
0 1 2 3 4 5 6 7 8 9
계속하려면 아무 키나 누르십시오 . . .
```

탐색

- 탐색(search)은 컴퓨터가 가장 많이 하는 작업
- (예) 인터넷 탐색

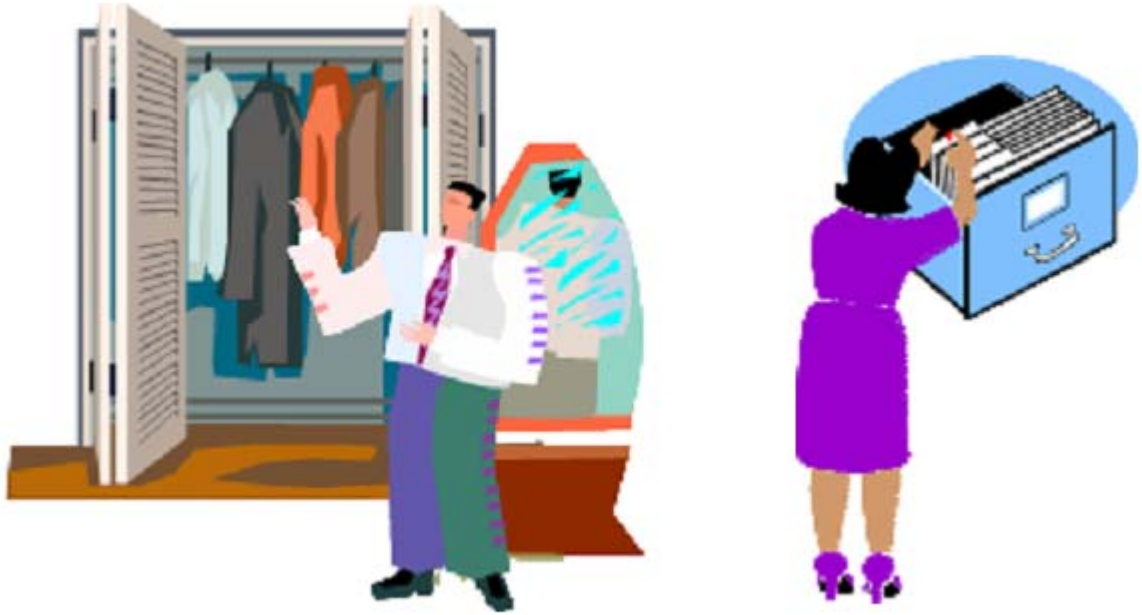
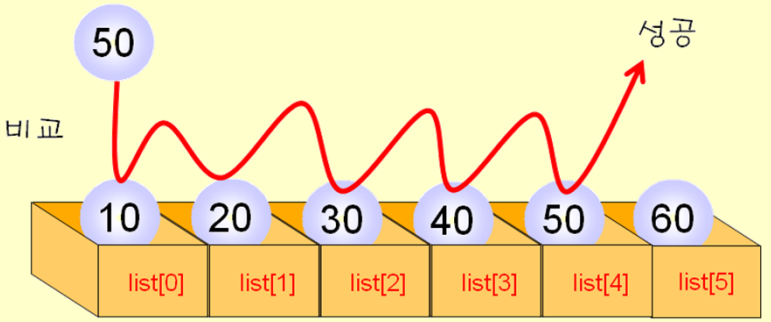


그림 8.12 탐색의 예

순차 탐색

```c
#include <stdio.h>
#define SIZE 10

int main(void)
{
	int key, i;
	int list[SIZE] = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

	printf("탐색할 값을 입력하시오:");
	scanf("%d", &key);

	for(i = 0; i < SIZE; i++)
		if(list[i] == key)
			printf("탐색 성공 인덱스= %d\n", i);

	printf("탐색 종료\n");
	return 0;
}
```



```
탐색할 값을 입력하시오:7
탐색 성공 인덱스 = 6
탐색 종료
```