제3장 배열과 문자열

이번 장에서 학습할 내용

- 배열의 개념
- 배열의 선언과 초기화
- 포인터의 개념
- 문자열의 개념

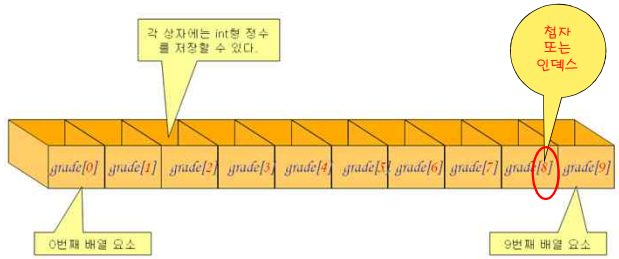
<활용>
- 랭킹 관리
  - 배열, 문자열 사용
  - 클래스 사용
  - 파일 입출력
  - 전역변수,지역변수 ,멤버변수

배열이란?

- 배열(array): 같은 종류의 데이터들이 순차적으로 저장되어 있는 자료 구조

배열 원소와 인덱스

- *인덱스(index):* 배열 원소의 번호

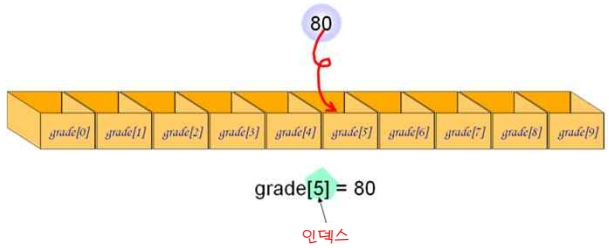


배열의 선언

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

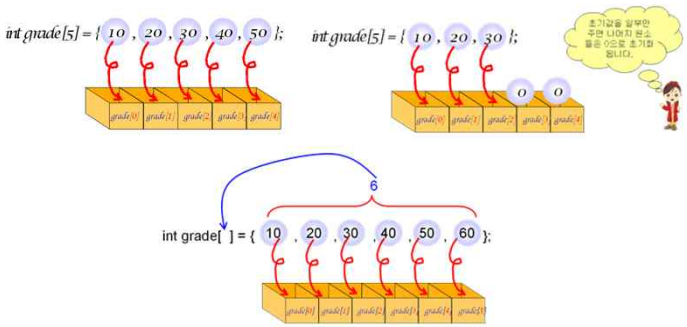
```
int  score[60];          // 60개의 int형 값을 가지는 배열 grade
float  cost[12];         // 12개의 float형 값을 가지는 배열 cost
char  name[50];          // 50개의 char형 값을 가지는 배열 name
char  src[10], dst[10];  // 2개의 문자형 배열을 동시에 선언
int   index, days[7];    // 일반 변수와 배열을 동시에 선언
```


배열 원소 접근




```
grade[5] = 80;
grade[1] =  grade[0];
grade[i] = 100;          // i는 정수 변수
grade[i+2] = 100;        // 수식이 인덱스가 된다.
grade[index[3]] = 100;   // index[]는 정수 배열
```


배열의 초기화



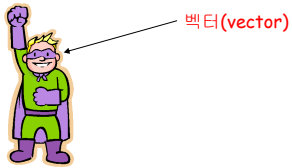
예제

```
#include <iostream>
using namespace std;
int main(void)
{
    const int STUDENTS=5;
    int grade[STUDENTS];
    int sum = 0;
    int i, average;
    for(i = 0; i < STUDENTS; i++)
    {
        cout << "학생들의 성적을 입력하시오: ";
        cin >> grade[i];
    }
    for(i = 0; i < STUDENTS; i++)
        sum += grade[i];
    average = sum / STUDENTS;
    cout << "성적 평균= " << average << endl;
    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
```

동적 배열

- C++에는 더 좋은 배열이 존재한다.
- STL 라이브러리로 제공

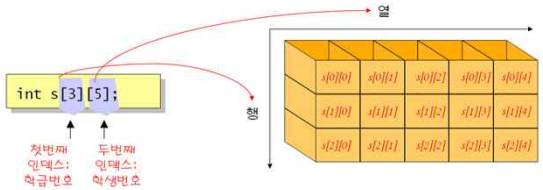


중간 점검 문제

1. n개의 원소를 가지는 배열의 경우, 첫 번째 원소의 번호는 무엇인가?
2. n개의 원소를 가지는 배열의 경우, 마지막 원소의 번호는 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?
4. 배열 a[6]의 원소를 1, 2, 3, 4, 5, 6으로 초기화하는 문장을 작성하라.
5. 배열의 초기화에서 초기값이 개수가 배열 원소의 개수보다 적은 경우에는 어떻게 되는가? 또 반대로 많은 경우에는 어떻게 되는가?
6. 배열의 크기를 주지 않고 초기값의 개수로 배열의 크기를 결정할 수 있는가?

2차원 배열

```
int s[10];      // 1차원 배열
int s[3][10];   // 2차원 배열
int s[5][3][10]; // 3차원 배열
```



2차원 배열의 초기화

```
int s[ ][5] = {
  { 0,  1,  2,  3,  4 }, // 첫 번째 행의 원소들의 초기값
  { 10, 11, 12, 13, 14 }, // 두 번째 행의 원소들의 초기값
  { 20, 21, 22, 23, 24 }, // 세 번째 행의 원소들의 초기값
};
```

3차원 배열

```
int s [6][3][5];
```

첫번째 인덱스: 학년번호 / 두번째 인덱스: 학급번호 / 세번째 인덱스: 학생번호

```
#include <iostream>
using namespace std;

int main()
{
	int s[3][3][3];		// 3차원 배열 선언
	int x, y, z;			// 3개의 인덱스 변수
	int i = 1;			// 배열 원소에 저장되는 값

	for(z=0;z<3;z++)
		for(y=0;y<3;y++)
			for(x=0;x<3;x++)
				s[z][y][x] = i++;
	return 0;
}
```


다차원 배열을 이용한 행렬의 표현

```
#include <iostream>
using namespace std;
const int ROWS=3;
const int COLS=3;

int main()
{
	int A[ROWS][COLS] = { { 2,3,0 }, { 8,9,1 }, { 7,0,5 } };
	int B[ROWS][COLS] = { { 1,0,0 }, { 1,0,0 }, { 1,0,0 } };
	int C[ROWS][COLS];
	int r,c;

	for(r = 0;r < ROWS; r++)
		for(c = 0;c < COLS; c++)
			C[r][c] = A[r][c] + B[r][c];
	for(r = 0;r < ROWS; r++) {
		for(c = 0;c < COLS; c++)
			cout << C[r][c] << " ";
		cout << endl;
	}
	return 0;
}
```

```
330
991
805
```

$$A=\begin{bmatrix}2&3&0\\8&9&1\\7&0&5\end{bmatrix} \quad B=\begin{bmatrix}000700\\900008\\000000\\650000\\000001\\002000\end{bmatrix}$$

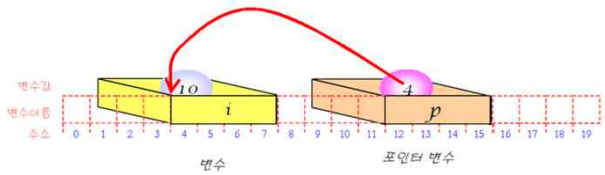
포인터란?

- *포인터(pointer):* 주소를 가지고 있는 변수

포인터의 선언

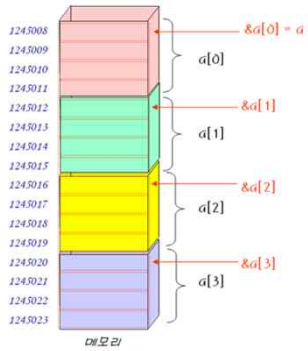
- 포인터: 변수의 주소를 가지고 있는 변수

```
int i = 10;		// 정수형 변수 i 선언
int *p = &i;		// 변수 i의 주소가 포인터 p로 대입
```



포인터와 배열

- 배열 이름은 첫 번째 배열 원소의 주소
- 포인터를 사용하여서 배열 원소를 처리 가능



예제

```
#include <iostream>
using namespace std;

int main(void)
{
        const int STUDENTS = 5;
        int grade[STUDENTS] = { 10, 20, 30, 40, 50 };

        for(int *p=grade, *pend=grade+STUDENTS; p != pend; p++)
                cout << *p << " ";
        return 0;
}
```

```
10 20 30 40 50
```


배열과 함수

- 배열을 함수의 인수로 전달하는 경우에는 배열 이름이 포인터이므로 포인터가 전달된다.

예제

```
#include <iostream>

int get_average(const int score[], int n);
void increment(int score[], int n);

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

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

        return 0;
}
void increment(int score[], int n)              // 함수정의
{
        int i;

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

예제

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

        for(i = 0; i < n; i++)
                sum += score[i];

        return sum / n;
}
```

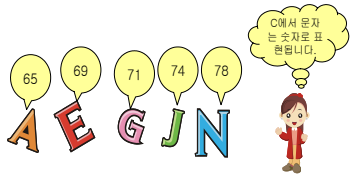
```
평균은 4입니다.
계속하려면 아무 키나 누르십시오 . . .
```


중간 점검 문제

1. 배열의 첫 번째 원소의 주소를 계산하는 2가지 방법을 설명하라.
2. 배열 a[]에서 *a의 의미는 무엇인가?
3. 배열의 이름에 다른 변수의 주소를 대입할 수 있는가?
4. 포인터를 이용하여 배열의 원소들을 참조할 수 있는가?
5. 포인터를 배열의 이름처럼 사용할 수 있는가?
6. 함수의 매개 변수로 전달된 배열을 변경하면 원본 배열이 변경되는가?
7. 배열을 전달받은 함수가 배열을 변경하지 못하게 하려면 어떻게 하여야 하는가?

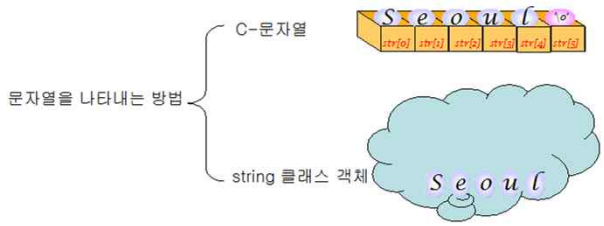
문자표현방법

- 컴퓨터에서는 각각의 문자에 숫자코드를 붙여서 표시한다.
- 아스키코드(ASCII code): 표준적인 8비트 문자코드
  - 0에서 127까지의 숫자를 이용하여 문자표현
- 유니코드(unicode): 표준적인 16비트 문자코드
  - 전세계의 모든 문자를 일관되게 표현하고 다룰 수 있도록 설계

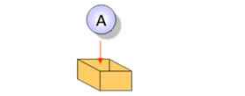


문자열 표현 방법

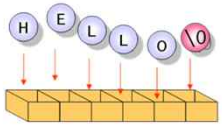
- *문자열(string)*: 문자들이 여러 개 모인것
  - "A"
  - "Hello World!"



C-문자열



하나의 문자는 char형 변수로 저장



문자열은 char형 배열로 저장

문자 배열의 초기화

1. 문자 배열 원소들을 중괄호 안에 넣어주는 방법
   - char str[6] = { 'H', 'e', 'l', 'l', 'o', '\0' };
2. 문자열 상수를 사용하여 초기화하는 방법
   - char str[6] = "Hello";·
3. 만약 배열을 크기를 지정하지 않으면 컴파일러가 자동으로 배열의 크기를 초기화값에 맞추어 설정
   - char str[] = "C Bible"; // 배열의 크기는 7이 된다.
4. strcpy()를 사용하여 문자열을 문자 배열에 복사
   - strcpy(str, "World");

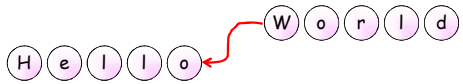
문자열 길이 계산 예제

```
// 문자열의 길이를 구하는 프로그램
#include <iostream>
using namespace std;

int main()
{
	char str[30] = "C++ language is easy";
	int i = 0;

	while(str[i] != 0)
		i++;
	cout << "문자열 "<< str << "의 길이는 " << i << "입니다." << endl;

	return 0;
}
```

```
문자열 C++ language is easy의 길이는 20입니다.
계속하려면 아무 키나 누르십시오 . . .
```


문자열 처리 라이브러리

함수	설명
strlen(s)	문자열 s의 길이를 구한다.
strcpy(s1, s2)	s2를 s1에 복사한다.
strcat(s1, s2)	s2를 s1의 끝에 붙여넣는다.
strcmp(s1, s2)	s1과 s2를 비교한다.
strncpy(s1, s2, n)	s2의 최대 n개의 문자를 s1에 복사한다.
strncat(s1, s2, n)	s2의 최대 n개의 문자를 s1의 끝에 붙여넣는다.
strncmp(s1, s2, n)	최대 n개의 문자까지 s1과 s2를 비교한다.
strchr(s, c)	문자열 s안에서 문자 c를 찾는다.
strstr(s1, s2)	문자열 s1에서 문자열 s2를 찾는다.



예제

```
// strcpy와 strcat
#include <iostream>
#include <cstring>
using namespace std;

int main()
{
      char string[80];

      strcpy( string, "Hello World from " );
      strcat( string, "strcpy() " );
      strcat( string, "and " );
      strcat( string, "strcat()!" );
      cout << string << endl;
      return 0;
}
```

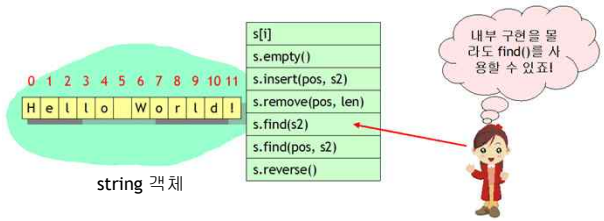
```
Hello World from strcpy() and strcat()!
```


중간 점검 문제

1. C-문자열 src를 C-문자열 dst로 복사하는 문장을 써라.
2. "String"을 저장하려면 최소한 어떤 크기 이상의 문자 배열이 필요한가?
3. 문자열을 서로 비교하는 함수는?

string 클래스

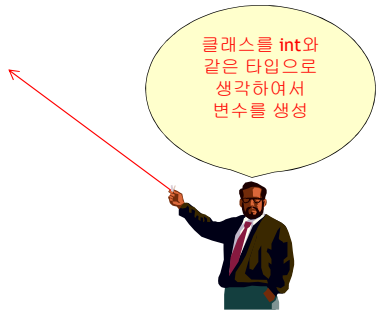
- C++에서는 문자열을 나타내는 클래스 string을 제공한다.



string 객체

클래스에서 객체를 생성하는 방법

- string s1;
- string s2 = "Hello World";



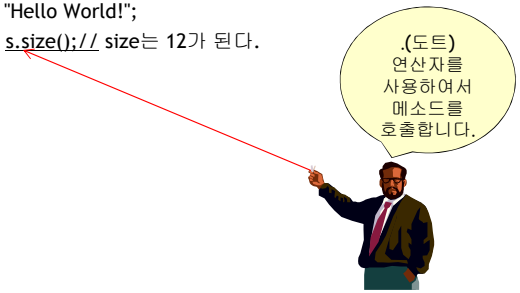
객체 생성의 예

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
      string s1 = "This is a test.";   // string 객체를 생성하고 초기화한다.
      string s2;                       // 비어있는 string 객체를 생성한다.

      cout << s1 << endl;
      cout << "문자열을 입력하시오: " << endl;
      cin >> s2;
      cout << s2 << endl;
      return0;
}
```

```
This is a test.
문자열을 입력하시오: This
This
```


멤버 함수 호출

- string s = "Hello World!";
- **int** size = s.size();// size는 12가 된다.



string 클래스의 멤버 함수

멤버 함수	설명
s[i]	i번째 원소
s.empty()	s가 비어있으면 true 반환
s.insert(pos, s2)	s의 pos 위치에 s2를 삽입
s.remove(pos, len)	s의 pos 위치에 len만큼을 삭제
s.find(s2)	s에서 문자열 s2가 발견되는 첫번째 인덱스를 반환
s.find(pos, s2)	s의 pos 위치부터 문자열 s2가 발견되는 첫번째 인덱스를 반환

멤버 함수 호출의 예

```
#include <string>
…
int main()
{
   // string 객체를 생성 및 초기화
    string s1 = "This is a test.";

    s1.insert(4, "Hello");
    cout << s1 << endl;
    int index = s1.find("test");
    cout << index << endl;
    s1.append("World");
    cout << s1 << endl;
    return 0;
}
```

```
ThisHello is a test.
15
ThisHello is a test.World
```

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s1("Slow"), s2("steady");
    string s3 = "the race.";
    string s4;

    s4 = s1 + " and " + s2 + " wins " + s3;
    cout << s4 << endl;
    return 0;
}
```

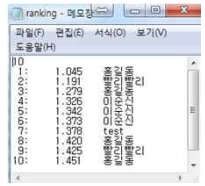
```
Slow and steady wins the race.
계속하려면 아무 키나 누르십시오 . .
```

구구단 게임의 랭킹 관리

- 랭킹에 필요한 정보
  - 경기자 이름, 성적, 게임 시각 등
  - ➔ 경기자 클래스를 만들자.
- 랭킹 저장에 사용할 자료구조 ➔ 일단 전역 변수로 ➔ 멤버 변수로
  - 경기자 리스트: 배열로 구현 (경기자 배열)
  - 현재 랭킹에 포함된 인원
- 랭킹 관리에 필요한 기능 ➔ 일단 일반 함수로 ➔ 나중에 멤버 함수로
  - 현재 랭킹을 화면에 출력한다. printRanking(FILE* fp);
  - 경기가 끝나면 랭킹을 갱신한다. addRanking(name, score);
  - 랭킹을 파일에서 읽어온다. loadRanking(filename);
  - 랭킹을 파일에 저장한다. storeRanking(filename);

경기자 클래스

```
#pragma once
#include <cstdio>
#include <cstring>

class Player
{
    double  score;
    char    name[80];
public:
    Player(void){ }
    ~Player(void){ }

    double getScore() { return score; }
    void set(char *na, double scr) {
        score = scr;
        strcpy(name, na);
    }

    void print(FILE *fp=stdout) {
        fprintf(fp, "%5.3f\t %s\n", score, name);
    }
    void read(FILE *fp) {
        char str[20];   // ranking: 읽어서 그냥 버림.
        fscanf(fp, "%s%lf%s", str, &score, name);
    }
};
```



- *클래스에 조금 더 친해지자.*
- *"문자열" 사용에 자신을 갖자.*
- *"파일"도 별 것 아니다.*

전역변수, 입출력함수

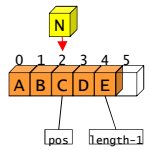
```
#include "Player.h"
const int NumTopPlayer = 10;

void printRanking( FILE *fp=stdout );

static Player   list[NumTopPlayer];
static int      nTopPlayer;

void loadRanking( char *filename )
{
    nTopPlayer = 0;
    FILE *fp = fopen(filename , "r");
    if( fp == NULL ) return;
    fscanf(fp, "%d", &nTopPlayer);
    if( nTopPlayer > NumTopPlayer )
        nTopPlayer = NumTopPlayer;

    for( int i=0 ; i<nTopPlayer ; i++ )
        list[i].read(fp);

    fclose(fp);
}
```

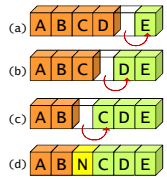
```
void storeRanking( char *filename  )
{
    FILE *fp = fopen(filename , "w");
    if( fp == NULL ) return;
    fprintf(fp, "%d\n", nTopPlayer);
    printRanking(fp);
    fclose(fp);
}
void printRanking( FILE *fp )
{
    for( int i=0 ; i<nTopPlayer ; i++ ) {
        fprintf(fp, "%2d:\t", i+1);
        list[i].print(fp);
    }
}
```

- *전역 변수로 경기자 배열과 현재 랭킹에 들어 있는 인원 사용.*
  - *외부에서 안 보이도록 static처리 (static, extern등 이해)*
- *파일 입력 출력을 너무 무서워하지 말자. (printf, scanf 충실히 공부)*
- *파일에 랭킹을 저장하고 읽으려면 파일 형식을 결정해야 한다.*

랭킹 갱신?

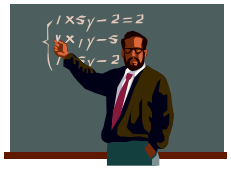
- 미리 결정할 사항: 랭킹? 게임의 수와 난이도는???
- 랭킹 갱신 함수: addRanking()
  - 자신의 순위 찾기
  - 찾은 위치에 넣기
  - List 사용 (자료구조)





- 배열을 사용한 리스트의 삽입 문제
- *리스트는 임의의 위치에 삽입, 삭제가 가능하다.*
- *삽입시 요소들을 뒤로 미는 과정이 필요하다.(배열로 구현시)*
- *리스트 알고리즘은 자료구조에서 더 자세히 공부한다.*

Q & A



41/40