C++ 문자열

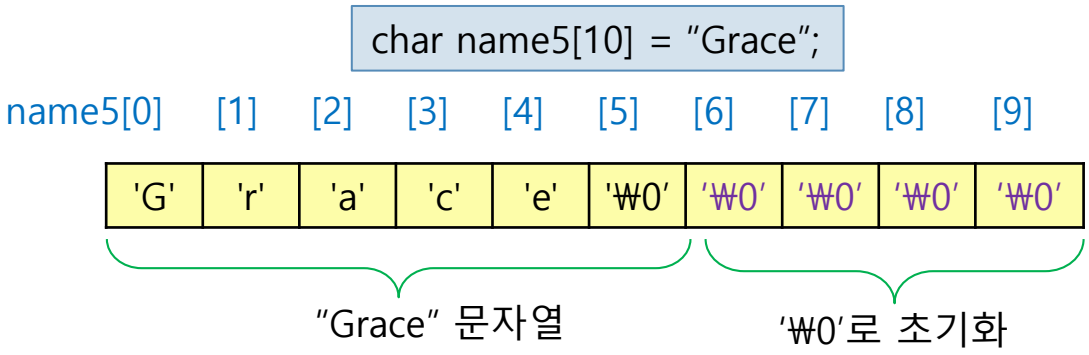
- C++의 문자열 표현 방식 : 2가지
  - **C-스트링 방식** – '₩0'로 끝나는 문자 배열

C-스트링 문자열

```
char name1[6] = {'G', 'r', 'a', 'c', 'e', '₩0'}; // name1은 문자열 "Grace"
```

단순 문자 배열

```
char name2[5] = {'G', 'r', 'a', 'c', 'e'}; // name2는 문자열이 아니고 단순 문자 배열
```



  - **string 클래스 이용**
    - <string> 헤더 파일에 선언됨
    - 다양한 멤버 함수 제공, 문자열 비교, 복사, 수정 등

C-스트링 방식으로 문자열 다루기

- C-스트링으로 문자열 다루기
  - C 언어에서 사용한 함수 사용 가능
    - strcmp(), strlen(), strcpy() 등
  - <cstring>이나 <string.h> 헤더 파일 include

```
#include <cstring> 또는
#include <string.h>
...
int n = strlen("hello");
```

  - <cstring> 헤더 파일을 사용하는 것이 바람직함
    - <cstring>이 C++ 표준 방식

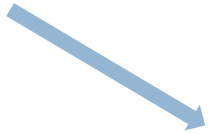
cin을 이용한 문자열 입력

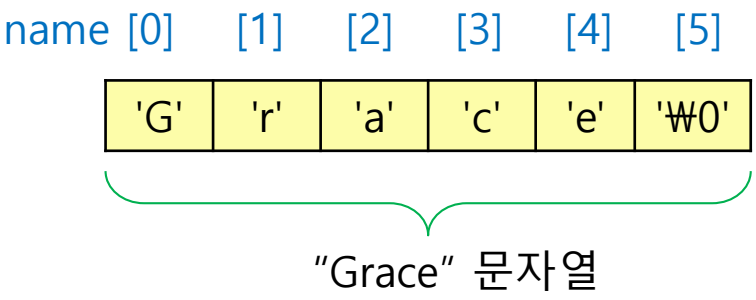
- 문자열 입력

```
char name[6]; // 5 개의 문자를 저장할 수 있는 char 배열
cin >> name; // 키보드로부터 문자열을 읽어 name 배열에 저장한다.
```

Grace

키 입력





예제 2–4 키보드에서 문자열 입력 받고 출력


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

int main() {
   cout << "이름을 입력하세요>>";

   char name[11]; // 한글은 5개 글자, 영문은 10까지 저장할 수 있다.
   cin >> name; // 키보드로부터 문자열을 읽는다.

   cout << "이름은 " << name << "입니다₩n"; // 이름을 출력한다.
}
```

```
이름을 입력하세요>>마이클
이름은 마이클입니다
```

빈 칸 없이 키 입력해야 함

```
이름을 입력하세요>>마 이 클
이름은 마입니다
```

빈 칸을 만나면 문자 열 입력 종료

예제 2-5 C-스트링을 이용하여 암호가 입력되면 프로그램을 종료하는 예


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

int main() {
    char password[11];
    cout << "프로그램을 종료하려면 암호를 입력하세요." << endl;
    while(true) {
        cout << "암호>>";
        cin >> password;
        if(strcmp(password, "C++") == 0) {
            cout << "프로그램을 정상 종료합니다." << endl;
            break;
        }
        else
            cout << "암호가 틀립니다~~" << endl;
    }
}
```

strcmp() 함수를 사용하기 위한 헤더 파일

```
프로그램을 종료하려면 암호를 입력하세요.
암호>>Java
암호가 틀립니다~~
암호>>C
암호가 틀립니다~~
암호>>C++
프로그램을 정상 종료합니다.
```

빈 칸 없이 키 입력해야 함

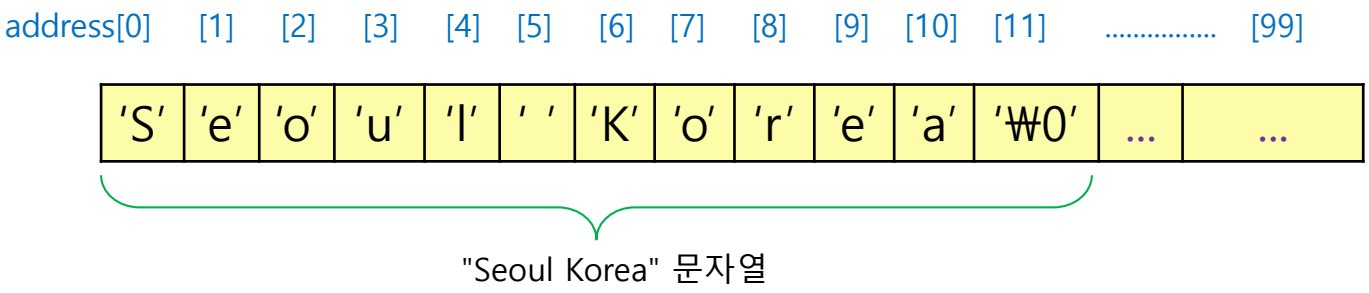
cin.getline()으로 공백이 낀 문자열 입력

- **공백이 낀 문자열**을 입력 받는 방법
- cin.getline(char buf[], int size, char delimitChar)
  - buf에 최대 size-1개의 문자 입력. 끝에 '₩0' 붙임
  - delimitChar를 만나면 입력 중단. 끝에 '₩0' 붙임
    - delimitChar의 디폴트 값은 '₩n'(<Enter>키)

```
char address[100];
cin.getline(address, 100, '₩n');
```

최대 99개의 문자를 읽어 address 배열에 저장. 도중에 <Enter> 키를 만나면 입력 중단

사용자가 'Seoul Korea<Enter>'를 입력할 때,



예제 2–6 cin.getline()을 이용한 문자열 입력


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

int main() {
   cout << "주소를 입력하세요>>";

   char address[100];
   cin.getline(address, 100, '₩n'); // 키보드로부터 주소 읽기

   cout << "주소는 " << address << "입니다₩n"; // 주소 출력
}
```

```
주소를 입력하세요>>컴퓨터시 프로그램구 C++동 스트링 1-1
주소는 컴퓨터시 프로그램구 C++동 스트링 1-1입니다
```

빈칸이 있어도 <Enter> 키가 입력 될 때까지 하나의 문자열로 인식

C++에서 문자열을 다루는 string 클래스

- string 클래스
  - C++에서 강력 추천
  - C++ 표준 클래스
  - 문자열의 크기에 따른 제약 없음
    - string 클래스가 스스로 문자열 크기게 맞게 내부 버퍼 조절
  - 문자열 복사, 비교, 수정 등을 위한 다양한 함수와 연산자 제공
  - 객체 지향적
  - <string> 헤더 파일에 선언
    - #include <string> 필요
  - C-스트링보다 다루기 쉬움

예제 2-7 string 클래스를 이용한 문자열 입력 및 다루기


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

int main() {
   string song("Falling in love with you"); // 문자열 song
   string elvis("Elvis Presley"); // 문자열 elvis
   string singer; // 문자열 singer

   cout << song + "를 부른 가수는"; // + 로 문자열 연결
   cout << "(힌트 : 첫글자는 " << elvis[0] << ")?";  // [] 연산자 사용

   getline(cin, singer); // 문자열 입력
   if(singer == elvis)  // 문자열 비교
         cout << "맞았습니다.";
   else
         cout << "틀렸습니다. "+ elvis + "입니다." << endl; // +로 문자열 연결
}
```

string 클래스를 사용하기 위한 헤더 파일

빈칸을 포함하는 문자열 입력 가능

getline()은 string 타입의 문자열을 입력 받기 위해 제공되는 전역 함수

```
Falling in love with you를 부른 가수는(힌트 : 첫글자는 E)?Elvis Pride
틀렸습니다. Elvis Presley입니다.
```

빈칸 포함