10강

문자와 문자열

동양미래대학교 강환수 교수

본 강의 사용 및 참조 자료

- Perfect C, 3판, 강환수 외 2인 공저, 인피니티북스, 2021

12장 문자와 문자열

목차

01

문자와 문자열

문자 개념

- 영어의 알파벳이나 한글의 한 글자를 작은 따옴표로 둘러싸서 'A'와 같이 표기
  - C 언어에서 저장공간 크기 1바이트인 자료형 char로 지원
- 작은 따옴표에 의해 표기된 문자를 문자 상수

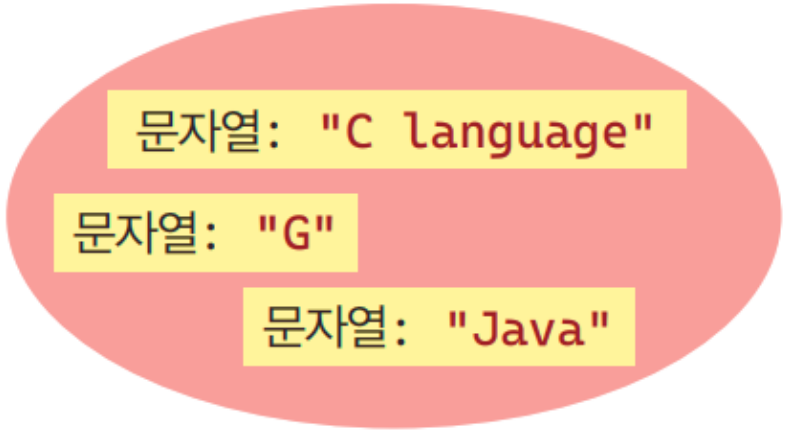
```
char ch ='A';
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 508쪽

문자열 개념

- 문자의 모임인 일련의 문자
  - 일련의 문자 앞 뒤로 큰 따옴표로 둘러싸서 “java”로 표기
- 큰 따옴표에 의해 표기된 문자열을 문자열 상수
  - “A”처럼 문자 하나도 큰 따옴표로 둘러싸면 문자열 상수
  - ‘ABC’처럼 작은 따옴표로 둘러싸도 문자가 될 수 없으며 오류가 발생

```
char c[] = "C language";
```



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 508쪽

문자

- char형 변수에 문자를 저장

```
char ch ='A';
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 509쪽

문자열 1/2

- 문자열을 저장하기 위한 자료형을 따로 제공하지 않음
  - 문자 배열을 선언하여 각각의 원소에 문자를 저장
  - 문자열의 마지막을 의미하는 NULL 문자 '₩0'가 마지막에 저장
- 문자열이 저장되는 배열크기
  - 반드시 저장될 문자 수보다 1이 커야
    널(NULL) 문자를 문자열의 마지막으로 인식
  - 문자열의 마지막에 널(NULL) 문자가 없다면
    출력과 같은 문자열 처리에 문제가 발생

문자열 2/2

- 배열 csharp의 크기를 3으로 선언한 후 배열 csharp에 문자열 “C#”을 저장
  - 마지막 원소인 csharp[2]에 ‘₩0’을 저장

```
char ch ='A';

char csharp[3];
csharp[0] = 'C'; csharp[1] = '#'; csharp[2] = '\0';
```

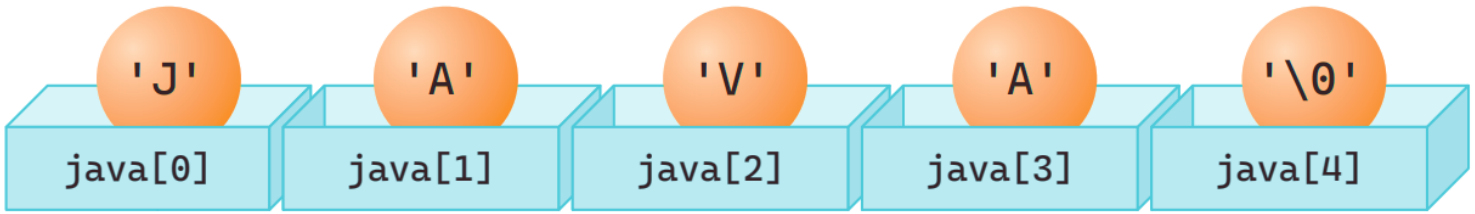
[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 509쪽

배열 선언 시 초기화 방법

- 중괄호를 사용
- 문자 하나 하나를 쉼표로 구분하여 입력하고 마지막 문자로 널(NULL)인 '₩0'을 삽입

```
//문자 하나하나 저장 시 마지막에 '\0' 문자 저장
char java[] = {'J','A','V','A', '\0'};
```

마지막에 '\0'을 빼면, 대입 시에는 문제가 없으나 출력 등에서 문제가 발생한다.



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 509쪽

문자열을 선언하는 편리한 다른 방법

- 배열 선언 시 저장할 큰 따옴표를 사용해 문자열 상수를 바로 대입
  - 배열 초기화 시 배열크기는 지정하지 않는 것이 더 편리
    - 지정한다면 마지막 문자인 '₩0'을 고려해
      실제 문자 수보다 1이 더 크게 배열크기를 지정
    - 지정한 배열크기가 (문자수+1)보다 크면
      나머지 부분은 모두 '₩0' 문자로 채워짐
  - 만일 배열크기가 작으면
    - 문자열 상수가 아닌 단순한 문자 배열이 되므로
      문자열 출력 등에서 문제가 발생

문자열 구성하는 문자 참조 1/2

문자열을 처리하는 다른 방법

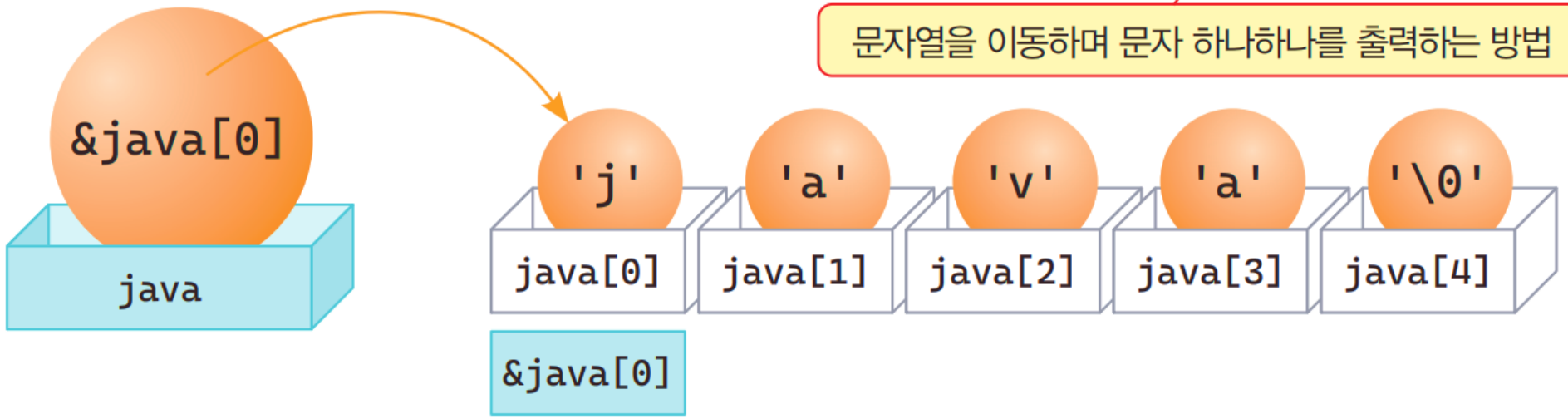
- 문자열 상수를 문자 포인터에 저장하는 방식
  - 문자열을 구성하는 문자 하나 하나의 수정은 불가능
- 문자열 출력
  - 함수 printf()에서
    포인터 변수와 형식제어문자 %s로 간단히 처리

문자열 구성하는 문자 참조 2/2

```
int i = 0;
char *java = "java";
printf("%s ", java);
```

```
while (java[i] != '\0')
    printf("%c", java[i++]);
printf("\n");
```

문자열을 이동하며 문자 하나하나를 출력하는 방법



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 512쪽

'₩0' 문자에 의한 문자열 분리 1/2

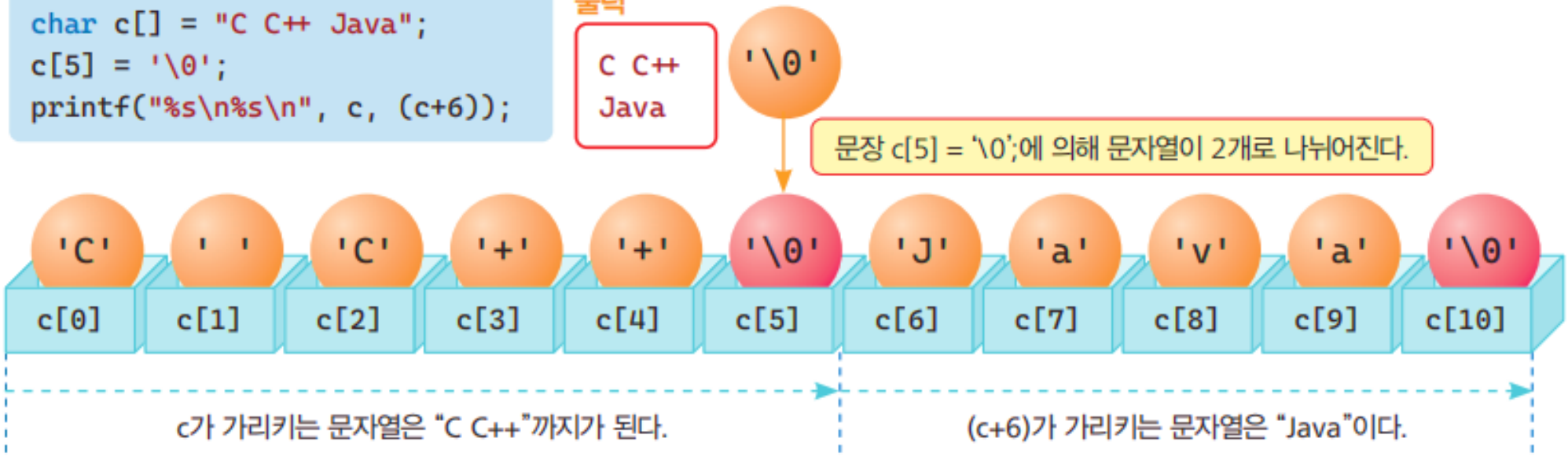
- 함수 printf()에서 %s
  - 문자 포인터가 가리키는 위치에서 NULL 문자까지를 하나의 문자열로 인식
- 만일 배열 c에 문장 c[5] = '₩0';을 실행
  - c를 출력하면 무엇이 출력될까?
  - c[5]에 저장된 '₩0' 문자에 의해
    c가 가리키는 문자열은 "C C++"까지
    - 즉 문자열은 시작 문자부터
      '₩0' 문자가 나올 때까지 하나의 문자열로 처리
  - (c+6)로 문자열을 출력하면 "Java"가 출력

'₩0' 문자에 의한 문자열 분리 2/2

```
char c[] = "C C++ Java";
c[5] = '\0';
printf("%s\n%s\n", c, (c+6));
```

출력

```
C C++
Java
```



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 513쪽

02

문자열 관련 함수

다양한 문자열 라이브러리 함수 1/2

- 헤더파일 string.h에 함수원형으로 선언된 라이브러리 함수로 제공
  - 문자열 비교와 복사, 그리고 문자열 연결 등과 같은 다양한 문자열 처리 함수
- 함수에서 사용되는 자료형: 64비트 윈도우 시스템인 경우
  - size_t
    - 비부호 정수 long long형(unsigned __int64)
  - void *
    - 아직 정해지지 않은 다양한 포인터를 의미

다양한 문자열 라이브러리 함수 2/2

- size_t strlen(const char *str)
  - 포인터 src 위치에서부터
    널(NULL) 문자를 제외한 문자열의 길이 반환
- void *memcpy(void *dest, const void *src, size_t n)
  - 포인터 src 위치에서
    dest에 n바이트를 복사한 후 dest 위치 반환

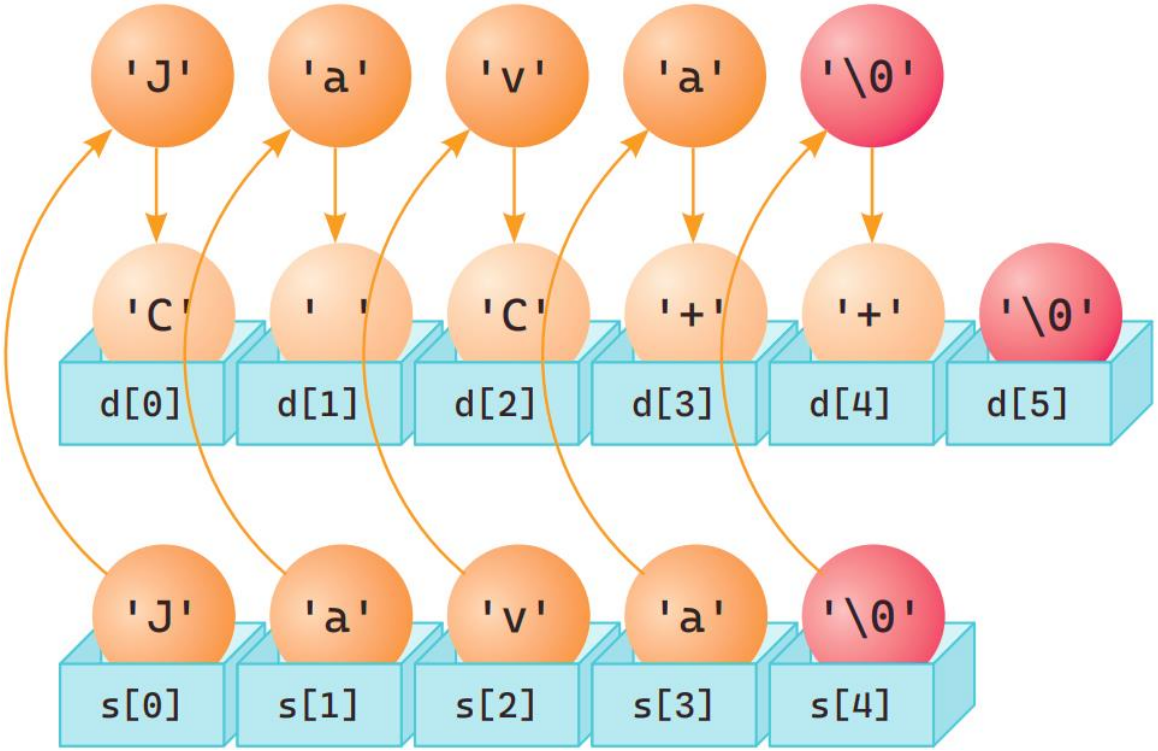
문자열 관련 함수

- 문자열 관련 함수는 대부분 strOOO()로 명명
  - 대표적인 문자열 처리 함수, 두 문자열을 비교하는 함수
- int strcmp(const char *s1, const char *s2);
  - 헤더파일 string.h 삽입
  - 두 인자인 문자열에서
    같은 위치의 문자를 앞에서부터 다를 때까지 비교하여
    같으면 0을 반환하고,
    앞이 크면 양수를, 뒤가 크면 음수를 반환

문자열 복사

- 함수 strcpy()
  - 문자열을 복사하는 함수
    - 앞 인자 문자열 dest에 뒤 인자 문자열 source를 복사
    - 항상 문자열은 마지막 널 문자까지 포함하므로 NULL 문자까지 복사
    - 첫 번째 인자 dest는 복사 결과가 저장될 수 있도록 충분한 공간을 확보
- 함수 strncpy()
  - 복사되는 최대 문자 수를 마지막 인자 maxn으로 지정하는 함수

문자열 복사



결과는 d에도 "java"가 저장된다.

```
char d[] = "C C++";
char s[] = "Java";
strcpy(d, s);
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 526쪽

문자열 연결 함수 strcat()

- 하나의 문자열 뒤에 다른 하나의 문자열을 연이어 추가해 연결
  - 앞 문자열에 뒤 문자열의 null 문자까지 연결
    - 앞의 문자열 주소를 반환하는 함수
  - 인자 dest의 저장공간이 연결된 문자열의 길이를 모두 수용할 수 있는 공간보다 부족: 문제 발생
- 문제를 예방하기 위한 함수가 strncat() 함수
  - 전달 인자의 마지막에 연결되는 문자의 수를 지정하여 그 이상은 연결되지 않도록
    - 여기서 지정하는 문자 수는 널 문자를 제외한 수

문자열 길이와 변환 함수

- 함수 strlen()
  - NULL 문자를 제외한 문자열 길이를 반환하는 함수
- 함수 strlwr()
  - 인자를 모두 소문자로 변환하여 반환
- 함수 strupr()
  - 인자를 모두 대소문자로 변환하여 반환

실습예제

Prj09 09strstr.c 다양한 문자열 관련 함수의 이해 난이도: ★

```c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>

int main(void)
{
    char str[] = "JAVA 2022 Python C";
    printf("%zu\n", strlen("python"));//python 길이: 6
    printf("%s, ", _strlwr(str));   //모두 소문자로 변환
    printf("%s\n", _strupr(str));   //모두 대문자로 변환

    //문자열 VA가 시작되는 포인터 반환: VA 2022 PYTHON C
    printf("%s, ", strstr(str, "VA"));
    //문자 A가 처음 나타나는 포인터 반환: AVA 2022 PYTHON C
    printf("%s\n", strchr(str, 'A'));

    return 0;
}
```

```
6
java 2022 python c, JAVA 2022 PYTHON C
VA 2022 PYTHON C, AVA 2022 PYTHON C
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 532쪽

03

여러 문자열 처리

문자 포인터 배열

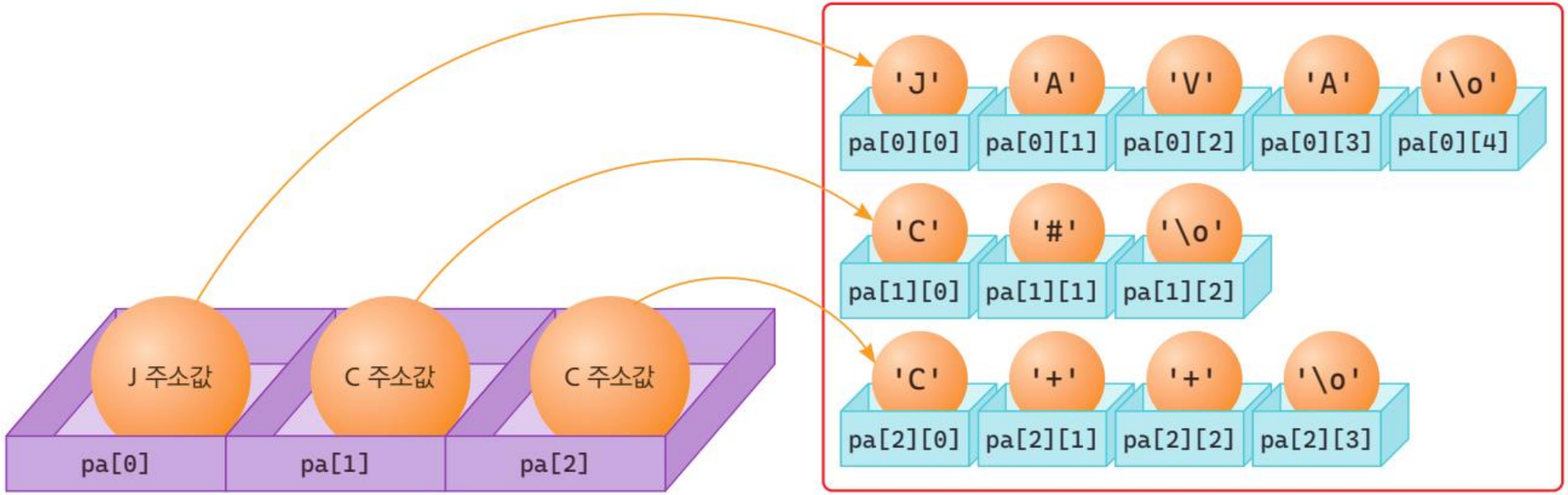
- 여러 개의 문자열을 처리하는 하나의 방법
  - 하나의 문자 포인터가 하나의 문자열을 참조 가능
    - 문자 포인터 배열은 여러 개의 문자열을 참조 가능
- 장 단점
  - 문자 포인터를 사용해서는 문자열 상수의 수정은 불가능
    - 문장 pa [0] [2] = 'v';와 같이 문자열의 수정은 실행오류가 발생
    - 문자 포인터 배열 pa를 이용
      - 각 문자열을 출력하려면 pa [i]로 형식제어문자 %s를 이용

문자 포인터 배열을 이용, 여러 개의 문자열 처리

```
char *pa[] = {"JAVA", "C#", "C++"};
//각각의 3개 문자열 출력
printf("%s ", pa[0]); printf("%s ", pa[1]); printf("%s\n", pa[2]);
```

배열의 크기는 문자열 개수인 3을 지정하거나 빈 공백으로 한다.

이 문자열의 수정은 실행 문제가 발생한다.



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 536쪽

문자 이차원 배열

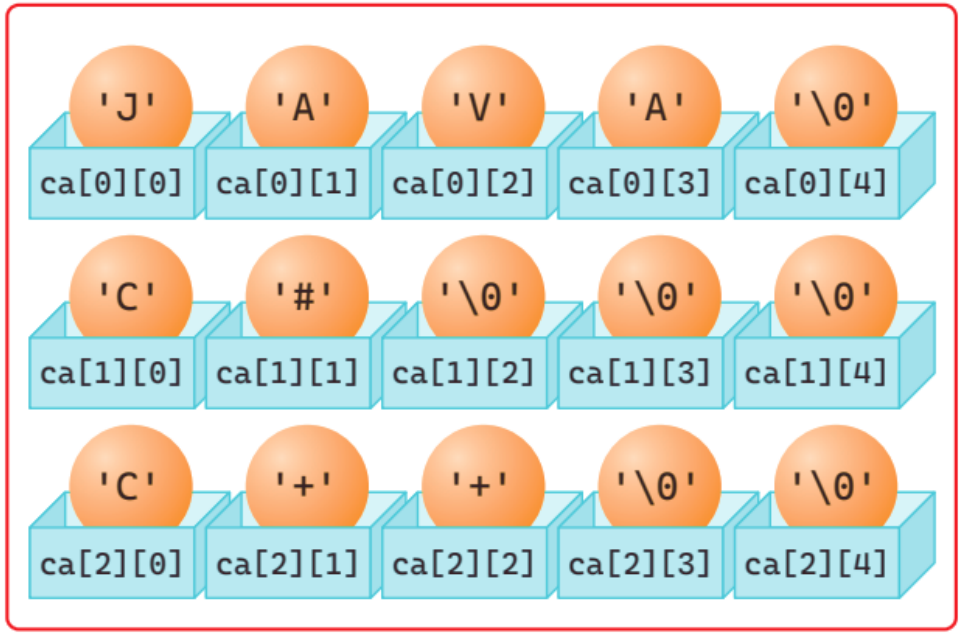
- 이차원 배열의 열 크기
  - 문자열 중에서 가장 긴 문자열의 길이보다 1 크게 지정
    - 가장 긴 문자열 "java"보다
      1이 큰 5를 2차원 배열의 열 크기로 지정
- 물론 이차원 배열의 행의 크기는 문자열 갯수
  - 3으로 지정하거나 공백으로 비워 둠

2차원 문자배열을 이용한 문자열 처리

```
char ca[][5] = {"JAVA", "C#", "C++"};
//각각의 3개 문자열 출력
printf("%s ", ca[0]); printf("%s ", ca[1]); printf("%s\n", ca[2]);
```

첫 번째(행) 크기는 문자열 개수를 지정하거나 빈 공백으로 두며, 두 번째(열) 크기는 문자열 중에서 가장 긴 문자열의 길이보다 1크게 지정한다.

이 문자열의 수정될 수 있다.



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 537쪽

실습예제 1/2

Prj10 10strary.c 여러 개의 문자열을 선언과 동시에 저장하고 처리하는 방법 난이도: ★★

```c
#include <stdio.h>

int main(void)
{
    char *pa[] = { "JAVA", "C#", "C++" };
    char ca[][5] = { "JAVA", "C#", "C++" };

    //pa[0][2] = 'v';   //실행 문제 발생
    //ca[0][2] = 'v';   //수정 가능
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 537쪽

실습예제 2/2

```
10      //각각의 3개 문자열 출력
11      printf("%s ", pa[0]); printf("%s ", pa[1]); printf("%s\n", pa[2]);
12      printf("%s ", ca[0]); printf("%s ", ca[1]); printf("%s\n", ca[2]);
13
14      //문자 출력
15      printf("%c %c %c\n", pa[0][1], pa[1][1], pa[2][1]);
16      printf("%c %c %c\n", ca[0][1], ca[1][1], ca[2][1]);
17
18      return 0;
19  }
```

문자열을 구성하는 각각의 문자를 출력하려면 pa[i][j]와 ca[i][j]을 형식제어문자 %c로 출력

```
JAVA C# C++
JAVA C# C++
A # +
A # +
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 538쪽

명령행 인자

- main(int argc, char *argv[])
  - 도스 명령어 dir를 프로그램으로 개발한다면 옵션 “/w” 등은 어떻게 인식할까?
    - 명령행 인자(command line arguments)를 사용하는 방법
      - 명령행에서 입력하는 문자열을 프로그램으로 전달하는 방법

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 537쪽

프로그램에서 명령행 인자를 받으려면

- 두 개의 인자 argc와 argv를 (int argc, char * argv[ ])로 기술
  - 매개변수 argc
    - 명령행에서 입력한 문자열의 수
  - argv[ ]
    - 명령행에서 입력한 문자열을 전달 받는 문자 포인터 배열
  - 실행 프로그램 이름도 하나의 명령행 인자에 포함

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 537쪽

명령어 옵션 실행

```
관리자: C:₩Windows₩System32₩cmd.exe

D:₩Kang C₩ch12₩x64₩Debug>dir /w
 D 드라이브의 볼륨: DATADRIVE0
 볼륨 일련 번호: 0AEE-91E2

 D:₩Kang C₩ch12₩x64₩Debug 디렉터리

[.]             [..]            Lab1.exe        Lab1.ilk        Lab1.pdb        Lab2.exe        Lab2.ilk        Lab2.pdb
Lab3.exe        Lab3.ilk        Lab3.pdb        Lab4.exe        Lab4.ilk        Lab4.pdb        Prj01.exe       Prj01.ilk
Prj01.pdb       Prj02.exe       Prj02.ilk       Prj02.pdb       Prj03.exe       Prj03.ilk       Prj03.pdb       Prj04.exe
Prj04.ilk       Prj04.pdb       Prj05.exe       Prj05.ilk       Prj05.pdb       Prj06.exe       Prj06.ilk       Prj06.pdb
Prj07.exe       Prj07.ilk       Prj07.pdb       Prj08.exe       Prj08.ilk       Prj08.pdb       Prj09.exe       Prj09.ilk
Prj09.pdb       Prj10.exe       Prj10.ilk       Prj10.pdb       Prj11.exe       Prj11.ilk       Prj11.pdb       Prj12.exe
Prj12.ilk       Prj12.pdb       Prj13.exe       Prj13.ilk       Prj13.pdb       Prj14.exe       Prj14.ilk       Prj14.pdb
Prj15.exe       Prj15.ilk       Prj15.pdb       Prj16.exe       Prj16.ilk       Prj16.pdb       Prj17.exe       Prj17.ilk
Prj17.pdb       test01.exe      test01.ilk      test01.pdb      test02.exe      test02.ilk      test02.pdb      test03.exe
test03.ilk      test03.pdb
               72개 파일              22,968,768 바이트
                2개 디렉터리  1,825,650,806,784 바이트 남음

D:₩Kang C₩ch12₩x64₩Debug>
```

도스 창에서 프로그램이 저장된 폴더의 경로와 >을 합쳐 프롬프트(prompt)라 하고, 명령어 프롬프트가 나타나는 이 줄에 명령어를 입력할 수 있으며, 이 줄을 명령행(command line)이라 한다.

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 539쪽

실습예제

Prj11 11cmdarg.c 명령행 인자 출력 난이도: ★★

```
01  #include <stdio.h>
02
03  int main(int argc, char* argv[])
04  {
05     int i = 0;
06
07     printf("실행 명령행 인자(command line arguments) >>\n");
08     printf("argc = %d\n", argc);
09     for (i = 0; i < argc; i++)
10        printf("argv[%d] = %s\n", i, argv[i]);
11
12     return 0;
13  }
```

argc(argument count)에 인자의 수가, argv(argument variables)에는 인자인 여러 개의 문자열의 포인터가 저장된 배열이 전달

```
실행 명령행 인자(command line arguments) >>
argc = 4
argv[0] = C:\Kang C\ch12\x64\Debug\Prj11.exe
argv[1] = Python
argv[2] = Go
argv[3] = Kotlin
```

비주얼 스튜디오에서 실행하면 전체 경로를 포함한 실행파일의 이름이 첫 번째 인자로 표시된다.

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 540쪽

명령행 인자 실행

- ₩ch12₩x64₩Debug 폴더
  - 비주얼 스튜디오의 실행파일이 생성된 폴더에서 명령 프롬프트를 실행

명령행 인자 전달

```
int main(int argc, char *argv[])
{
    ...
}
```

도스 창

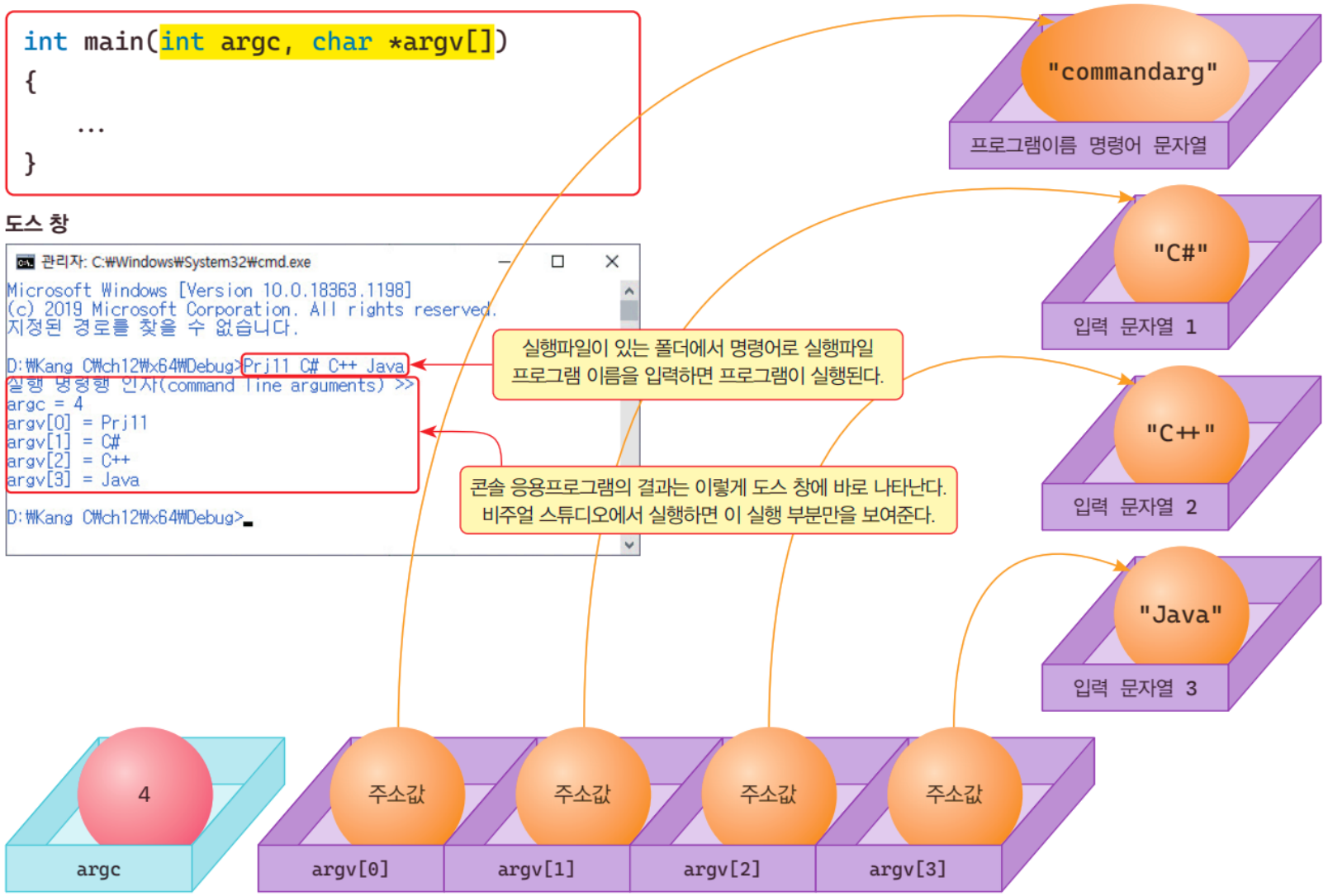
```
관리자: C:₩Windows₩System32₩cmd.exe
Microsoft Windows [Version 10.0.18363.1198]
(c) 2019 Microsoft Corporation. All rights reserved.
지정된 경로를 찾을 수 없습니다.

D:₩Kang C₩ch12₩x64₩Debug>Prj11 C# C++ Java
실행 명령행 인자(command line arguments) >>
argc = 4
argv[0] = Prj11
argv[1] = C#
argv[2] = C++
argv[3] = Java

D:₩Kang C₩ch12₩x64₩Debug>_
```

실행파일이 있는 폴더에서 명령어로 실행파일 프로그램 이름을 입력하면 프로그램이 실행된다.

콘솔 응용프로그램의 결과는 이렇게 도스 창에 바로 나타난다. 비주얼 스튜디오에서 실행하면 이 실행 부분만을 보여준다.



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 541쪽

정 리 하 기

정리하기

- 문자는 작은 따옴표, 문자열은 큰 따옴표를 사용해 각각 표현한다.
- 문자열을 저장하기 위한 자료형을 따로 제공하지 않으며,
  문자 배열을 선언하여 각각의 원소에 문자를 저장한다.
- 문자열의 마지막을 의미하는 NULL 문자 '₩0'가
  마지막에 저장되도록 한다.
- 문자열 연결은 strcat(), 문자열 복사는 strcpy() 함수를 사용한다.
- 여러 문자열은 문자 포인터 배열이나 2차원 문자 배열을 활용한다.
- 명령행 인자를 사용하기 위해서는
  두 개의 인자 argc와 argv를 main(int argc, char * argv[])로
  기술해 사용한다.

11강

구조체와 공용체

다음시간안내