

11. 문자와 문자열

6주차

문자열

◆ 문자

- 한 글자를 작은 따옴표로 둘러싸서 표기('A')
 - ❖ C 언어에서 : 1byte 자료형 char로 지원
- 작은 따옴표에 의해 표기된 문자 = 문자 상수

◆ 문자열 상수

- “ ” 안쪽에 문자들을 나열
“C language!”
- 일반문자, 제어문자
- 맨 뒤에 눈에 보이지 않는 NULL문자('\0')가 있음.

◆ 문자열 변수

- C언어에서는 문자열 자료형이 따로 없음
- 문자 자료형의 배열을 문자열 변수로 사용
- 맨 끝에 NULL문자('\0')를 붙여야 온전한 문자열

문자 배열(문자열)의 초기화

◆ 문자열 상수로서 초기화

```
char a2[] = "abc" ;
```

맨 뒤에 자동으로
NULL문자가 붙게 됨

◆ 문자의 나열로서 초기화

```
char a1[] = { 'a' , 'b' , 'c' , '\0' } ;
```

NULL문자를
붙여줘야 됨

◆ 위 두 문자 배열이 정확히 같은 저장장소(4 byte씩)를 잡게 됨.

'a'	'b'	'c'	0x00
-----	-----	-----	------

문자와 문자열 출력 (chararray.c)

```
#include <stdio.h>
int main(void) {
    char ch = 'A';
    printf("%c %d\n", ch, ch);

    char java[] = { 'J', 'A', 'V', 'A', '\0' };
    printf("%s\n", java);
    char c[] = "C language";
    printf("%s\n", c);
    char csharp[5] = "C#";
    printf("%s\n", csharp);

    printf("%c%c\n", csharp[0], csharp[1]);
    return 0;
}
```

문자 선언과 출력

3가지 문자열 선언 방법

[결과]

A 65

JAVA

C language

C#

C#

문자열 상수를 문자 포인터에 저장하는 방식 (charpointer.c)

```
#include <stdio.h>
int main(void) {
    char *java = "java";
    printf("%s ", java);
```

[결과]

java java java

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

문자 포인터가 가리키는 문자부터
NULL문자까지 하나 하나 출력

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

```
//java[0] = 'J';
return 0;
```

java는 문자열 상수를 가리키고 있으므로
수정 불가능! 실행오류 발생!

}

'\0' 문자에 의한 문자열 분리 (string.c)

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

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

[결과]

C C++ Java

C C++

Java

C C++ Java

문자열 2개로서 출력

```
c[5] = ' '; //널 문자를 빈 문자로 바꾸어 문자열 복원
```

```
char *p = c;
```

```
while (*p != '\0')
```

```
    printf("%c", *p++);
```

```
printf("\n");
```

```
return 0;
```

```
}
```

문자 배열의 각 원소를 하나
하나 문자로 출력

문자입력 함수 비교

◆ getchar()

- 라인 버퍼링(line buffering) 방식 사용

- ❖ [enter] 키 입력까지 buffer에 저장하다가 [enter] 들어오면 buffer의 내용을 한꺼번에 프로그램 buffer로 줌
- ❖ 즉각적(interactive)인 입력을 요구할 때는 사용 불가능

- #include <stdio.h>

◆ _getche()

- 버퍼를 사용하지 않으므로 문자 하나를 입력하면 바로 함수가 return됨.

- 입력된 문자를 모니터에 표시

- #include <conio.h>

◆ _getch()

- 버퍼를 사용하지 않으므로 문자 하나를 입력하면 바로 함수가 return됨.

- 입력된 문자를 모니터에 표시하지 않음

- #include <conio.h>

다양한 문자 입력 (getche.c)

```
#include <stdio.h>
#include <conio.h>
int main(void) {
    char ch;
    printf("문자를 계속 입력하고 Enter를 누르면 >>\n");
    while ((ch = getchar()) != 'q')
        putchar(ch);
    printf("\n문자를 누를 때마다 두 번 출력 >>\n");
    while ((ch = _getche()) != 'q')
        putchar(ch);
    printf("\n문자를 누르면 한 번 출력 >>\n");
    while ((ch = _getch()) != 'q')
        _putch(ch);
    printf("\n");
    return 0;
}
```

[결과]

문자를 계속 입력하고 Enter를 누르면 >>

abc

abc

deq

de

문자를 누를 때마다 두 번 출력 >>

aabbccq

문자를 누르면 한 번 출력 >>

abc

문자열 입출력

- ◆ **scanf(), printf()**
 - 변환문자 %s 사용
 - 공백으로 구분되는 하나의 문자열 입력 받음
 - ❖ 이름과 성을 분리하여 입력한다면 성만 저장됨
 - 입력 받을 문자열은 충분한 공간이 준비되어 있어야 함
 - 단순히 문자 포인터로는 문자열 저장이 불가능
 - printf()에서 %10s : 폭이 10, 우측정렬
- ◆ **gets()**
 - #include <stdio.h>
 - [enter] 키를 누를 때까지 버퍼에 저장 한 후 입력처리
 - 마지막에 입력된 '\n' 가 '\0' 로 교체되어 저장
- ◆ **puts()**
 - 오류가 발생하면 EOF를 반환
 - ❖ EOF(End Of File) : #define EOF (-1)
 - 문자열의 마지막에 저장된 '\0' 를 '\n' 로 교체하여 버퍼에 전송
-> puts()를 연속으로 사용하면 모니터에 한 행씩 보임

문자열 입출력 : scanf(), printf() (stringput.c)

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

```
int main(void) {
    char name[20], dept[30];
```

문자열 저장할 충분한 공간 확보!
char *name, *dept;
실행 오류 발생

```
printf("%s", "학과 입력 >> ");
```

```
scanf("%s", dept);
```

```
printf("%s", "이름 입력 >> ");
```

```
scanf("%s", name);
```

공백 없이 입력해야 함

```
printf("출력: %10s %10s\n", dept, name);
```

```
return 0;
```

```
}
```

[결과]

학과 입력 >> computer

이름 입력 >> jychoi

출력: computer jychoi

문자열 입출력 : gets(), puts() (gets.c)

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

```
int main(void) {
    char line[101];
```

문자열 저장할 충분한 공간 확보!
char *line 으로는 오류발생.

```
printf("입력을 종료하려면 새로운 행에서 (ctrl + Z)를 누르십시오.\n");
while (gets(line))
    puts(line);
printf("\n");
```

공백 없이 입력해야 함

```
/*while (gets_s(line, 101))
    puts(line);
printf("\n");*/
return 0;
}
```

[결과]

입력을 종료하려면 새로운 행에서 (ctrl + Z)를

abc

abc

de

de

^Z