[4주차 2강]
문자열(2)

학습 내용

- 10.2 문자열 저장 및 기본 입출력 (2)
- 10.3 문자열과 포인터

학습 목표

- 10.2 문자열의 기본 사용법을 익힌다.
- 10.3 문자열과 포인터의 관계를 이해한다.

C 프로그래밍 및 실습

1. 문자열 개요
2. **문자열 저장 및 기본 입출력 (2)**
3. 문자열과 포인터
4. 문자열의 배열
5. 문자열 및 문자 처리 함수
6. 문자열 및 문자 입출력

2. 문자열 저장 및 기본 입출력 (2)

scanf() 함수를 이용한 문자열 입력

- 서식 지정자 : %s
- 인자 : 문자열을 저장할 시작 주소(보통 배열의 이름)
- 사용자로부터 입력 받은 문자열을 인자로 전달된 주소부터 차례로 저장

```
char str[20];

scanf("%s", str);
printf("%s!!\n", str);
scanf("%s", str+5);
printf("%s!!\n", str);
```

실행 예시

```
Hello          → 입력
Hello!!        → 출력
World          → 입력
HelloWorld!!   → 출력
```

2. 문자열 저장 및 기본 입출력 (2)

scanf의 %s 서식

- 개행 문자, 공백 문자, 탭 문자 직전까지를 하나의 문자열로 인식
- 마지막에 널 문자를 자동으로 추가

```
char str[20];

scanf("%s", str);
printf("%s!!\n", str);
```

실행 예시

```
Hello World          → 입력
Hello!!              → 출력
```

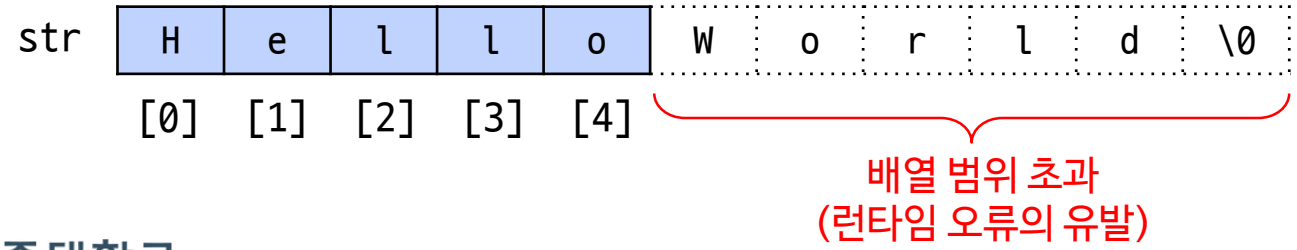
2. 문자열 저장 및 기본 입출력 (2)

주의 사항

- **문자열(널 문자 포함)을 저장할 충분한 공간이 확보**되어 있어야 함
  (모든 배열에 공통적인 사항)

```
char str[5];  // 크기 5인 배열
scanf("%s",str);
```

- 만약, 사용자가 "HelloWorld"를 입력한다면?
  ✓배열 범위를 벗어난 메모리 영역에 입력 받은 문자 저장
  ✓위 문자열을 저장하기 위해서는 배열의 크기가 11이상 이어야 함
  (왜 10 이상이 아니고, 11 이상일까?)



※ 실습하기

[예제 10.2]
다음 프로그램을 작성하여 실행해보자.

- ✓ 크기가 6인 문자 배열 str을 선언
- ✓ 사용자로부터 문자열 "Hello"를 입력 받아 str에 저장
- ✓ 문자열 str을 화면에 출력
- ✓ str[5]에 물음표 문자 '?' 대입
- ✓ 문자열 str을 화면에 출력

  ➢ 왜 이런 출력 결과가 나오는지 생각해보자.

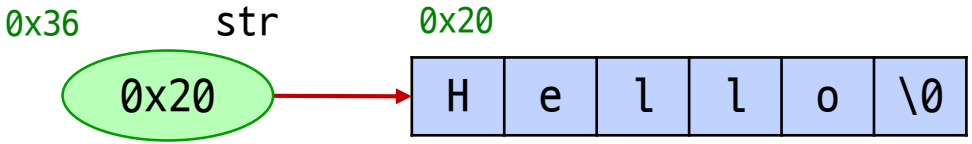
C 프로그래밍 및 실습

1. 문자열 개요
2. 문자열 저장 및 기본 입출력 (2)
3. **문자열과 포인터**
4. 문자열의 배열
5. 문자열 및 문자 처리 함수
6. 문자열 및 문자 입출력

문자형 포인터를 활용한 문자열 처리문

- 문자형 포인터를 사용한 간단한 코드
  - ✓문자형 포인터 str을 선언하고, 문자열 (상수) "Hello"를 가리키도록 초기화
  - ✓str에 주소가 저장되어 있으므로, printf의 %s 서식 이용해 출력

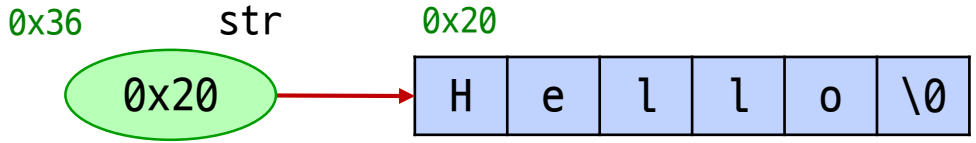
```
char *str = "Hello";          // 초기화

printf("%s!!\n", str);        // 출력
```



문자형 포인터를 배열처럼 사용하기

- 배열과 포인터의 관계 (9장에서 학습)를 이용

```
char *str = "Hello";

for (i=0 ; i<5 ; i++)
   printf("%c", str[i]);  // 문자 출력
```



※ 실습하기

[예제 10.3]
다음 프로그램을 작성하시오.

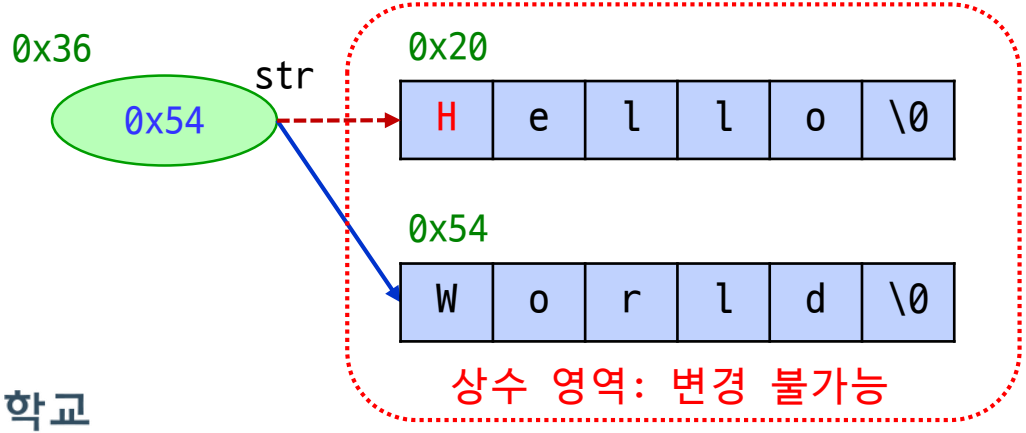
- 문자 포인터 변수 pc를 선언하고, 다음 문자열로 초기화
  - ✓ "To be, or not to be : that is the question"
- 반복문을 사용하여 영어 소문자 't' 가 몇 번 나오는 지 계산
  - ✓ 힌트 : 널 문자 여부를 반복 종료 조건으로 사용
- 다음과 같이 출력
  - ✓ 힌트 : 큰 따옴표와 작은 따옴표를 출력하기 위해 \" 과 \' 사용

문자열 "To be, or not to be : that is the question"에
문자 't'가 6번 나타남

문자 배열과 문자열 상수 비교

- "Hello"는 문자열 상수로, 사용자 프로그램에서 **변경 불가능**
- 반면, str은 사용자 변수로 값을 변경할 수 있음

```
char *str = "Hello";

str[0] = 'h';  // 변경 불가능 (런타임 오류 발생)
str = "World";  // str에 저장된 값 변경 (가능)
```

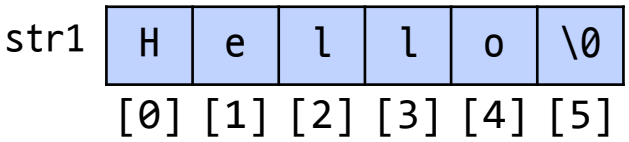


문자 배열과 문자형 포인터 비교

- 외우려고 하지 말고, 메모리 그림을 그려서 이해하자!!!

```
char str1[6] = "Hello";
printf("%c", str1[0]);// O
printf("%s", str1);   // O

str1[0] = 'h';        // O
scanf("%s", str1);    // O

str1 = "World";       // X
```

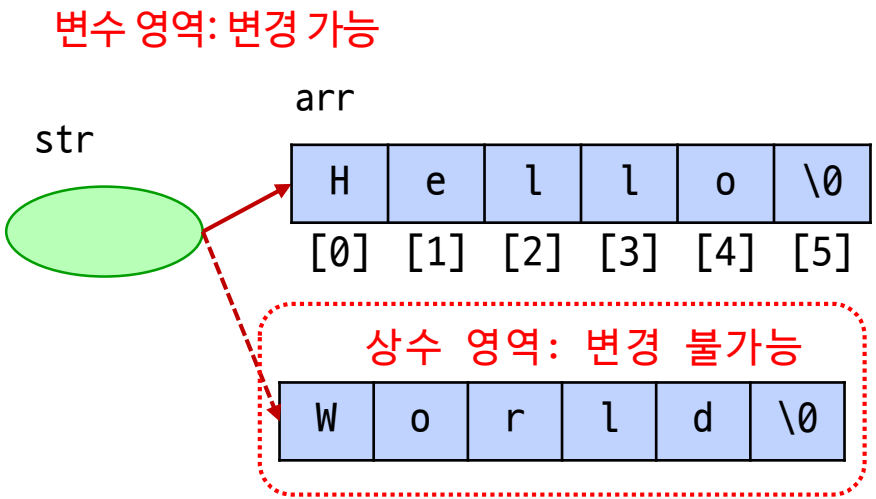
```
char *str2 = "Hello";
printf("%c", str2[0]);// O
printf("%s", str2);   // O

str2[0] = 'h';        // X
scanf("%s", str2);    // X

str2 = "World";       // O
```



3. 문자열과 포인터

주의!!

- str이 포인터여서 문자 변경이 안 되는 것이 아님
- **str이 가리키는 영역의 성질**에 따라 달라짐




```
char arr[6] = "Hello";
char *str = arr;

str[0] = 'h';        // O
scanf("%s", str);    // O

str = "World";       // O

str[0] = 'w';        // X
scanf("%s", str);    // X
```

학습 정리

- printf() 함수와 scanf() 함수에서 문자열 입출력을 위한 서식지정자는 **'%s'**
- scanf를 이용하여 문자열을 입력받는 경우, **공백 문자, 개행 문자, 탭 문자 직전까지** 입력된 문자들을 문자열로 인식하여 저장
- **문자 포인터**를 활용하여 문자열을 처리할 수 있음