C 프로그래밍 및 실습

[14주차 2강]
포인터(2)

학습 내용

9.2 포인터 선언과 사용(1부)

학습 목표

9.2 포인터의 선언과 사용 방법을 익힌다.

C 프로그래밍 및 실습

9.2 포인터 선언과 사용(1부)

포인터 (변수) 선언

- 구문 : 변수 명 앞에 * **(참조연산자)**만 덧붙이면 됨
  - ✓ 기존의 자료형 표시 + 포인터라는 표시

  - ✓ 예)

```
char    *pch;
int     *pnum;
```

  - ✓ pch와 pnum은 똑같이 주소를 저장하지만
    대상의 자료형이 다르기 때문에 **다른 자료형**으로 취급

  - ✓ pch는 **문자형 포인터** (변수)이고 pnum은 **정수형 포인터** (변수)

세종대학교

초기화

- 일반 변수 초기화 형태와 동일

```
int num, *pnum = &num;
```

✓(주의!!) num이 먼저 선언 되어야 함

```
int *pnum = &num, num;      // 컴파일 오류
```

포인터 변수 선언의 다양한 형태

✓동일 기본 자료형(int)에서 파생된 자료형의 변수는 모아서 선언 가능

```
int *pnum1, num1=10, *pnum2, num2, arr[10];
```

✓그러나, 가독성 때문에 추천 안 함

포인터 대입 (연결)

- 포인터 (변수)에 주소를 대입하여 특정 변수와 연결시키는 것을 "가리킨다"라고 표현하고, 그림에서는 화살표 → 로 표시

```
char ch = 'A', *pch;

pch = &ch;          // pch에 변수 ch의 주소 대입(연결)

printf("%p\n", pch); // %p: 주소 출력 서식
printf("%p\n", &ch);
```

실행 예시

```
001EA03C
001EA03C
```

세종대학교

포인터 연결 (예시 2)

```
int num;
int *pnum = &num; // 선언과 동시에 초기화

num = 3;

printf("%p\n", pnum); // %p: 주소 출력 서식
printf("%p\n", &num);
```

실행 예시

```
001EA072
001EA072
```

세종대학교

포인터 참조

- 포인터 (변수)가 가리키는 변수에 접근하는 것
- **참조 연산자 *** (간접연산자, 포인터연산자라고도 부름)를 사용
  - ✓ 예) *pch : 포인터 pch가 가리키는 변수, 0x3C번지에 저장된 값

```
char ch = 'A';
char *pch = &ch;

printf("%p %c\n", pch, *pch);
printf("%p %c\n", &ch, ch);
```

실행 예시

```
001EA03C A
001EA03C A
```

세종대학교

포인터 참조 (예시)

```
int num, *pnum = &num;

num = 3;

printf("%p %d\n", pnum, *pnum);
printf("%p %d\n", &num, num);
```

실행 예시

```
001EA072 3
001EA072 3
```

참조 연산자를 이용한 대입 예시 1

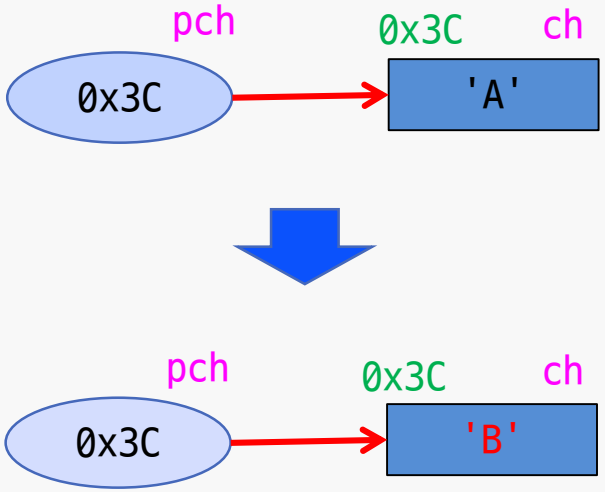
- *pch = 'B' 의 의미 : pch가 가리키는 공간에 'B' 대입
- *pch = 'B'는 ch = 'B'와 동일한 기능 수행

  전자는 **간접 접근**, 후자는 **직접 접근**

```
char ch = 'A', *pch = &ch;

printf("%c %c\n", *pch, ch);

*pch = 'B';

printf("%c %c\n", *pch, ch);
```

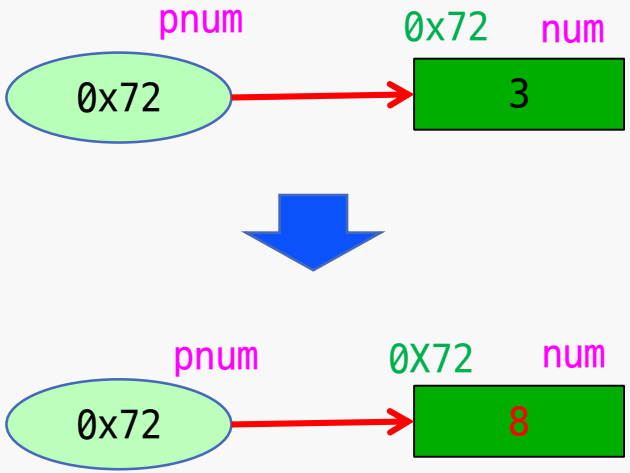
실행 결과

```
A A
B B
```



참조 연산자를 이용한 대입 예시 2

```
int num, *pnum = &num;

num = 3;

printf("%d %d\n", *pnum, num);

*pnum += 5;

printf("%d %d\n", *pnum, num);
```



실행 결과

```
3 3
8 8
```

참조 연산자 추가 예시

- *pnum은 정수를 나타내므로, 정수를 사용하는 어떤 형태든 가능
- 단, 참조연산자와 다른 연산자와의 우선 순위에 주의해서 사용

```
int num = 3, *pnum = &num;

*pnum = *pnum / 2 + 4;     ⇨ 정수 연산: num에 num/2+4 = 5 대입

if( *pnum == 5)              ⇨ 정수 비교: ( num == 5 )
  ++*pnum;                   ⇨ 정수 연산: ++(*pnum) ⇨ ++num
printf("%d", *pnum);         ⇨ 함수의 인자로 사용
```

실행 결과

```
6
```

실습 하기

[예제 9.3]
1단계 : 다음에 해당하는 문장들을 차례로 작성하고,
2단계 : 메모리 그림을 그려보시오. (그림1 ~ 그림6)

- ✓ (그림 1) int 변수 num1, num2 선언,
  int 포인터 변수 p 선언 및 num1의 주소로 초기화(하나의 문장으로)
- ✓ (그림 2) p가 가리키는 변수에 3000 대입
- ✓ (그림 3) num2에 p가 가리키는 변수값 대입
- ✓ (그림 4) p가 num2를 가리키도록 변경

실습 하기

[예제 9.3] (계속)
1단계 : 다음에 해당하는 문장들을 차례로 작성하고,
2단계 : 메모리 그림을 그려보시오. (그림1 ~ 그림6)

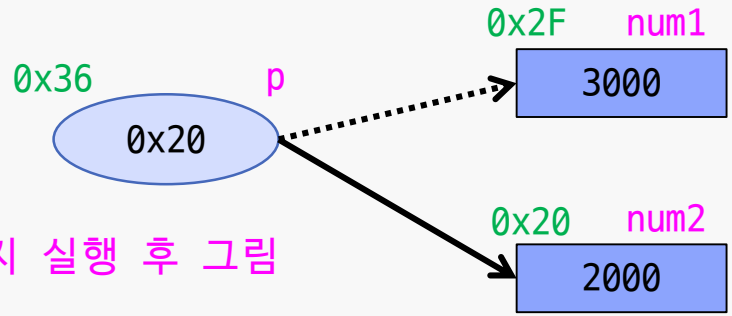
- ✓ (그림 5) p에 연결된 변수에 p가 가리키는 변수 값 - 1000 대입
- ✓ (그림 6) num1에 p가 가리키는 변수 값의 2배 대입
- ✓ num1, num2, p를 출력한다.
- ✓ num1의 주소, num2의 주소, p의 주소를 출력한다.

실습 하기

[예제 9.3] (코드)
1단계 : 다음에 해당하는 문장들을 차례로 작성하고,
2단계 : 메모리 그림을 그려보시오. (그림1 ~ 그림6)

```
int num1, num2, *p = &num1;

*p = 3000;
num2 = *p;
p = &num2;
*p = *p - 1000;
num1 = *p * 2; // 첫 번째 *은 참조연산자, 두 번째 *은 곱셈연산자

printf("값: num1=%d num2=%d p=%p\n", num1, num2, p);
printf("주소: num1=%p num2=%p p=%p\n", &num1, &num2, &p);
```

이 문장까지 실행 후 그림



학습 정리

- 포인터 변수는 참조 연산자(*)를 사용하여 **선언**함
- 포인터 변수는 다른 변수에 **연결**을 시킨 후 사용해야 함
- 포인터 변수가 가리키는 변수에 접근하기 위해서는 **참조 연산자(*)**를 사용함