Today

1. 지난 시간 복습 - 배열
2. 다차원 배열
3. 포인터
4. 실습

메모리와 주소

- 메모리는 바이트(Byte) 단위로 나뉘며, 각 바이트에는 주소가 지정되어 있음
- 예시) 정수형 메모리 할당
  - 정수형(int) 변수의 크기는 4바이트
    →정수형 변수 a를 선언하면 임의의 위치에 4바이트가 자리잡음
  - 변수가 위치하는 곳을 '주소'라고 하며, 주소를 알려면 변수앞에 '&'를 붙임 (&a)

```
int a = 100;
int b = 200;
```

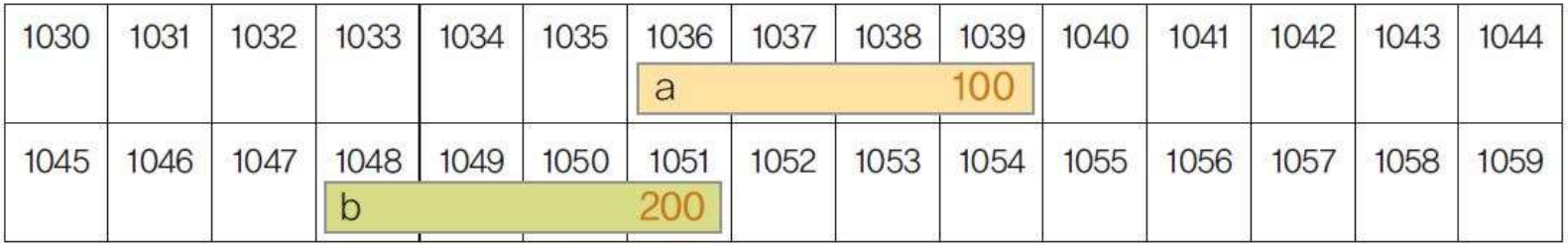


그림 9-6 메모리에 할당된 정수형 변수의 예

메모리와 주소

- scanf() 함수를 이용하여 데이터를 입력받을 때, 정수의 경우에는 &a와 같이 변수 앞에 &를 붙이고 문자열의 경우에는 &를 붙이지 않는다
- 변수 앞에 위치하는 &의 의미는 무엇일까?

컴퓨터의 메모리 내에 변수가 할당된 곳의 주소(address)

```c
#include <stdio.h>

int main( )
{
    int a;
    char aa[20];

    printf("숫자를 입력하세요 : \n");
    scanf("%d", &a);

    printf("문자열을 입력하세요 : \n");
    scanf("%s", aa);
}
```

변수의 주소 알아내기

```c
int main( )
{
    int a = 100;
    int b = 200;

    printf("변수 a의 주소는 %d\n", &a);
    printf("변수 b의 주소는 %d\n", &b);
}
```

- &a를 출력하면 변수 a가 저장되어 있는 메모리의 주소가 출력됨
- 출력되는 주소는 컴퓨터에 따라 달라질 수 있음

배열인 변수의 주소

```
int aa[3] = { 10, 20, 30 };
```

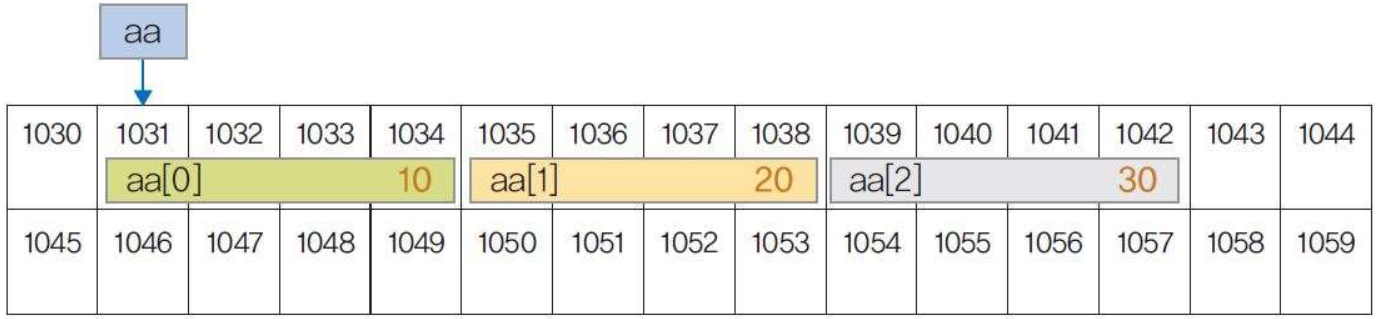


그림 9-7 메모리에 할당된 정수형 배열의 예

배열(aa)의 주소는 배열의 첫번째 변수(aa[0])의 주소와 같다

- 배열의 주소 표현
  - aa[0]의 주소(&aa[0]) = 1031번지
  - aa[1]의 주소(&aa[1]) = 1035번지
  - aa[2]의 주소(&aa[2]) = 1039번지
- 배열 이름 aa = 전체 배열의 주소 = 1031번지
  배열 aa의 주소를 구할 때는 '&'를 쓰지 않고 단순히 'aa'로 표현

```
int aa[3] = {10, 20, 30};

printf("aa[0]의 주소는 %d\n", &aa[0]);
printf("aa[1]의 주소는 %d\n", &aa[1]);
printf("aa[2]의 주소는 %d\n", &aa[2]);
printf("배열 aa의 주소는 %d\n", aa);
```

배열은 이름 자체가 주소를 의미하기 때문에 &aa가 아니라
단순히 aa로 표현하면 주소값을 얻을 수 있다!!

포인터의 개념

- 포인터는 주소를 담는 변수로, 선언은 *을 붙임

```
int a;
```
정수(int)형 변수 a를 선언

```
int* p;
```
정수(int)형 변수의 주소를 저장하는 포인터 p를 선언

```
char ch;
```
문자(char)형 변수 ch를 선언

```
char* q;
```
문자(char)형 변수의 주소를 저장하는 포인터 q를 선언

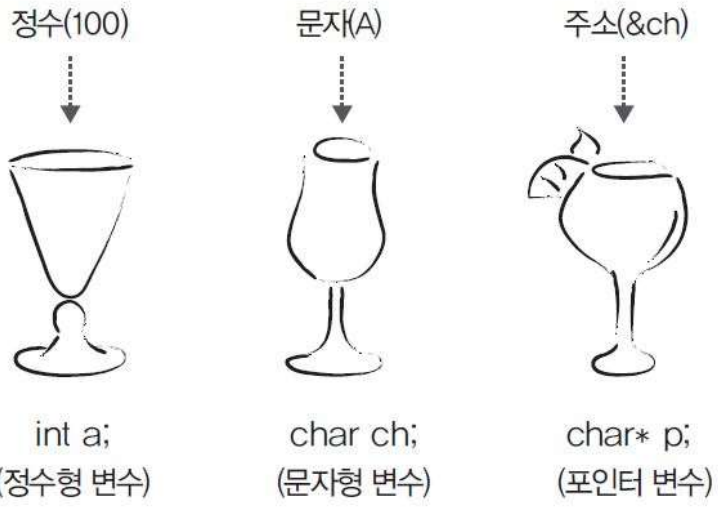


그림 9-9 변수의 종류

포인터 사용해보기

```c
int main( )
{
    int a;        //int형 변수 a 선언
    int* p;       //int형 변수의 주소를 저장하는 포인터 p 선언

    a = 100;      //변수 a에 100을 대입
    p = &a;       //포인터 p에 변수 a의 주소를 대입

    printf("변수 a의 값 (a) = %d\n", a);
    printf("변수 a의 주소 (&a) = %d\n", &a);
    printf("포인터 p의 값 (p) = %d\n", p);
    printf("포인터 p가 가리키는 변수의 값 (*p) = %d\n\n", *p);

    *p = 200;     //포인터 p가 가리키는 변수(a)에 200을 대입

    printf("포인터 p의 값 (p) = %d\n", p);
    printf("변수 a의 값 (a) = %d\n", a);
}
```

a == *p	포인터 p 앞에 *를 붙이면 포인터가 가리키는 변수(a)의 값을 의미
&a == p	변수 a앞에 &를 붙이면 변수의 주소(즉, 포인터의 값 p)을 의미

일반 변수와 포인터 변수 관계 **(1/2)**

```c
#include <stdio.h>

int main()
{
    double a;                   //double형 변수 a 선언
    double* q;                  //double형 변수의 주소를 저장하는 포인터 q 선언

    q = &a;                     //포인터 q에 변수 a의 주소를 대입

    puts("실수를 입력하세요.");
    scanf("%lf", q);            //&a == q

    printf("%f\n", *q);         //a == *q
}
```

일반 변수와 포인터 변수 관계 (2/2)

```
int main(){
    char ch;
    char* pch;

    ch = 'A';
    pch = &ch;

    printf("ch 값: %c\n", ch);
    printf("ch 주소: %d \n", &ch);
    printf("pch 값: %d \n", pch);
    printf("*pch 값: %c \n", *pch);
}
```

문자형 변수

문자형 포인터변수

문자형 변수에 값 할당

문자형 포인터변수에 주소값 할당

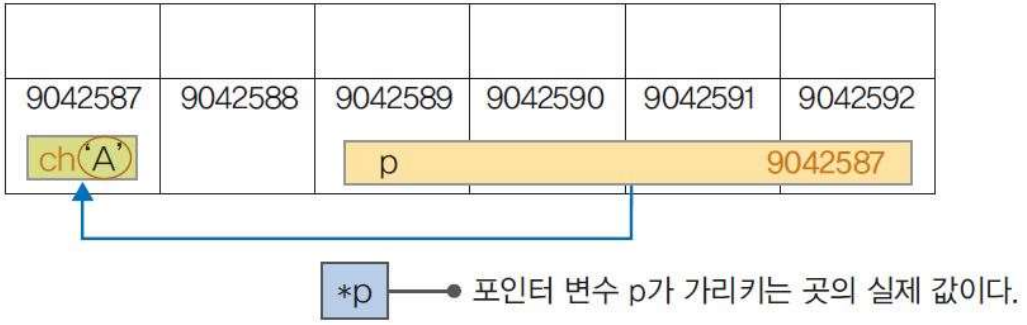


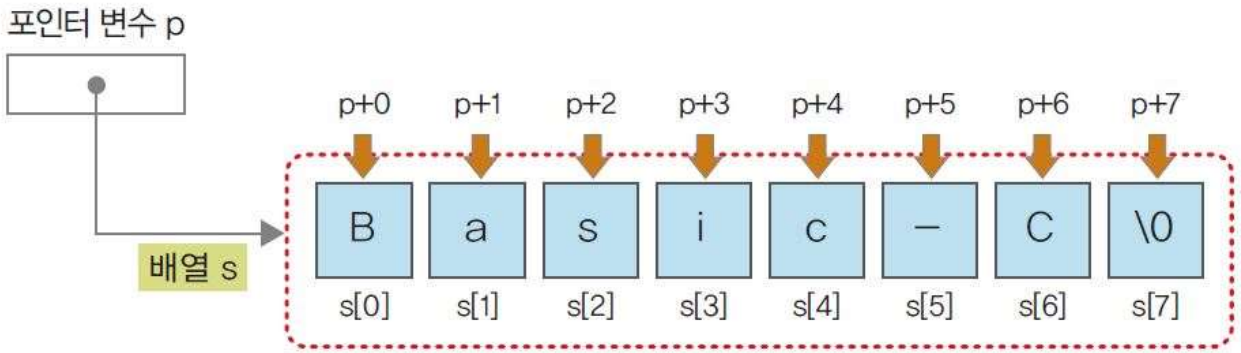
그림 9-11 [기본 9-6]의 변수와 포인터의 관계

포인터 변수

```c
#include <stdio.h>

int main()
{
    int a, b;
    int* p;
    int* q;

    p = &a;
    q = &b;

    puts("두 정수를 입력하세요.");
    scanf("%d %d", p, q);
    printf("두 수의 합은 %d입니다.\n", *p + *q);
}
```

문자열과 포인터

```c
int main(){
    char s[8] = "Basic-C";
    char* p;
    p = &s;
    printf("%s\n", p);

    printf("&s[3]: %s \n", &s[3]);
    printf("p+3: %s \n", p+3);

    printf("s[3]: %c \n", s[3]);
    printf("*(p+3): %c \n", *(p + 3));
}
```

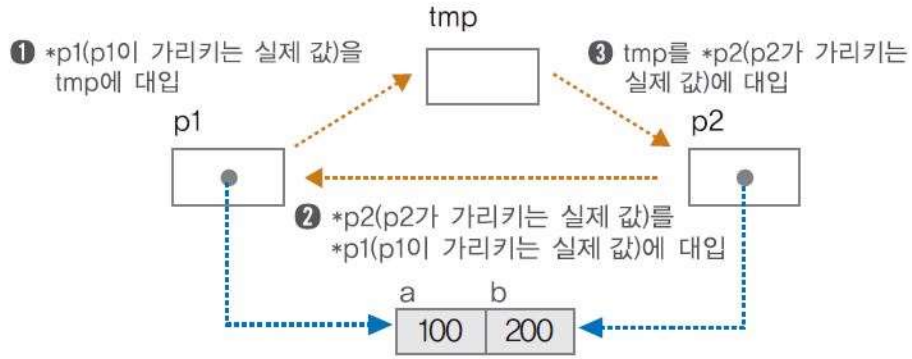



```
Basic-C
&s[3]: ic-C
p+3: ic-C
s[3]: i
*(p+3): i
```

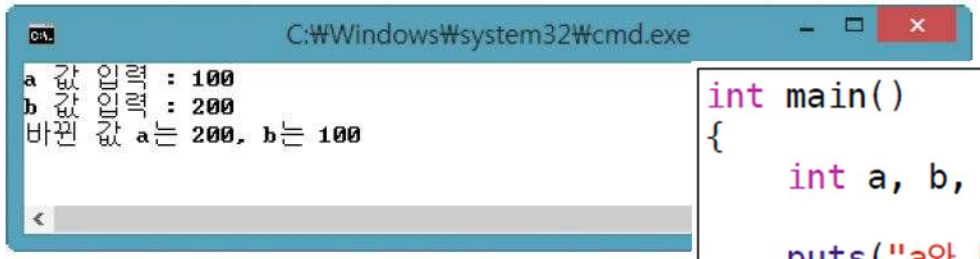
그림 9-15 [응용 9-9]의 변수와 포인터의 관계

[실습25] 포인터를 이용한 두 값 교환

예제 설명 입력한 두 값을 포인터를 활용하여 교환하는 프로그램이다. 값을 바꾸는 개념은 다음과 같다.



실행 결과




```c
int main()
{
    int a, b, temp;

    puts("a와 b값을 입력하세요.");
    scanf("%d %d", &a, &b);

    temp = a;
    a = b;
    b = temp;

    printf("a = %d, b = %d\n", a, b);
}
```