쉬프트 연산자를 사용한 바이트 나눠쓰기

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
void main(void)
{
	unsigned int uiData = 0;			// 본 데이터
	unsigned int uiMask = 0;			// 바이트 쪼갤때 사용 할 마스크 데이터
	unsigned char byByteData = 0;		// 바이트 단위 쪼개 넣을 공간

	int ilnputPos;
	int ilnputData;
	unsigned char bylnputData;
	int iCount;

	while (1)
	{
		printf("위치 (1~4) : ");
		scanf("%d", &ilnputPos);
		printf("값 [0~255] : ");
		scanf("%d", &ilnputData);
		//------------------------------------------------------------
		// 위치 범위 오류 체크
		//------------------------------------------------------------
		if (ilnputPos <= 0 || ilnputPos >= 5)
		{
			printf("위치 범위를 초과하였습니다. WnWn");
			continue;
		}
		//------------------------------------------------------------
		// 값 오류 체크
		//------------------------------------------------------------
		if (ilnputData < 0 || ilnputData > 255)
		{
			printf("데이터 범위를 초과하였습니다. WnWn");
			continue;
		}
		bylnputData = ilnputData;
		//------------------------------------------------------------
		// 입력 위치에 해당하는 비트 마스크 생성
		// 0xff = 8bit 11111111 로 되어있는 값.
		// 입력 위치에 - 1 하여 0 ~ 3으로 바꾸고 1당 8Bit (1Byte) 씩 옮긴다.
		// 이렇게 만들면 나오는 uiMask 값은 다음과 같다.
		// 00000000 00000000 00000000 11111111  -> ilnputPos = 1
		// 00000000 00000000 11111111 00000000  -> ilnputPos = 2
		// 00000000 11111111 00000000 00000000  -> ilnputPos = 3
		// 11111111 00000000 00000000 00000000  -> ilnputPos = 4
		// 이처럼 다른 비트에 있는 값은 버리고 특정
		// 위치의 값만 빼내는 용도의 비트집합을 마스크 라고 표현한다.

		uiMask = 0xff << (ilnputPos - 1) * 8;

		//------------------------------------------------------------
		// 입력 할 위치의 값을 먼저 지운다.
		// 그리해야... 새로운 값을 넣을 수 있으므로.

		// 대신 다른 위치의 값들은 그대로 유지를 해야한다.
		// 그러므로 지우고자 하는 위치만 0 으로, 나머지는 1로 마스크를 만들어서 AND 연산을 한다.

		uiData = uiData & ~uiMask;
		// 위에서 만든 Mask 에 대해 NOT (~) 연산을 하면 비트가 반전된다.

		// 예를 들면 다음과 같다.
		// 11111111 00000000 11111111 11111111  이런 비트를 AND 로 씌우면 0이 있는 비트는 모두 지워진다.

		//------------------------------------------------------------
		// 이제 입력 받은 데이터를 해당 위치에 넣자.
		// 입력받은 1Byte 의 값을 넣을 위치까지 왼쪽으로 쉬프트 시킨 후에....
		// 이를 uiData 에 OR 연산으로 값을 넣는다.

		uiData = uiData | (bylnputData << (ilnputPos - 1) * 8);

		//------------------------------------------------------------
		// 바이트별로 쪼개서 값 출력하기.
		// 4바이트, 4번 돌자

		for ( iCount = 0; iCount < 4; iCount++)
		{
			// 바이트 단위로 뽑기 위한 마스크를 새로 만든다.
			// 이번에는 특정 위치의 8비트만 뽑을 것이므로 바로 AND 연산을 사용한다.

			uiMask = 0xff << iCount * 8;

			// 이를 그냥 AND 로 씌워버리면 특정 위치의 값만 뽑아지지만....
			// 11111111 00000000 00000000 00000000
			// 이 처럼 상위 비트에 대해 값을 뽑으면 여전히 상위 비트에 데이터가 존재한다.
			// 8비트짜리 변수로 만들기 위해서는 이를 끝으로 끌어 내려야 한다.

			byByteData = (uiData & uiMask) >> iCount * 8;
			printf("%d 번째 바이트 값 : %d", iCount + 1, byByteData);
			printf("Wn");
		}
		printf("Wn");
		printf("전체 4바이트 값 : 0x%08x WnWn", uiData);

	}
}
```