



논리 설계 및 실험

Date : 2016 . 03 . 18

김영대

자료형이란

■ 자료형(data type)

- 변수가 저장하는 데이터 형식



■ 자료형의 종류

- 정수형 : 정수를 표현하는 데이터 타입
- 실수형 : 소수점이 표현된 값을 표현하는 데이터 타입
- 문자형 : 문자를 표현하는 데이터 타입

자료형의 크기

- **sizeof 연산자** : 자료형의 크기를 구하는 연산자(변수에 저장할 수 있는 범위를 구하기 위해)
- **sizeof 연산자의 장점** : 자료형에 할당되는 메모리의 크기를 구할 수 있다.

사용법	예	설명
sizeof(자료형)	Printf("%d", sizeof(int));	자료형의 메모리 크기를 출력
sizeof(변수)	int num1 = 3; Printf("%d", sizeof(num1));	변수의 메모리 크기를 출력

자료형의 크기

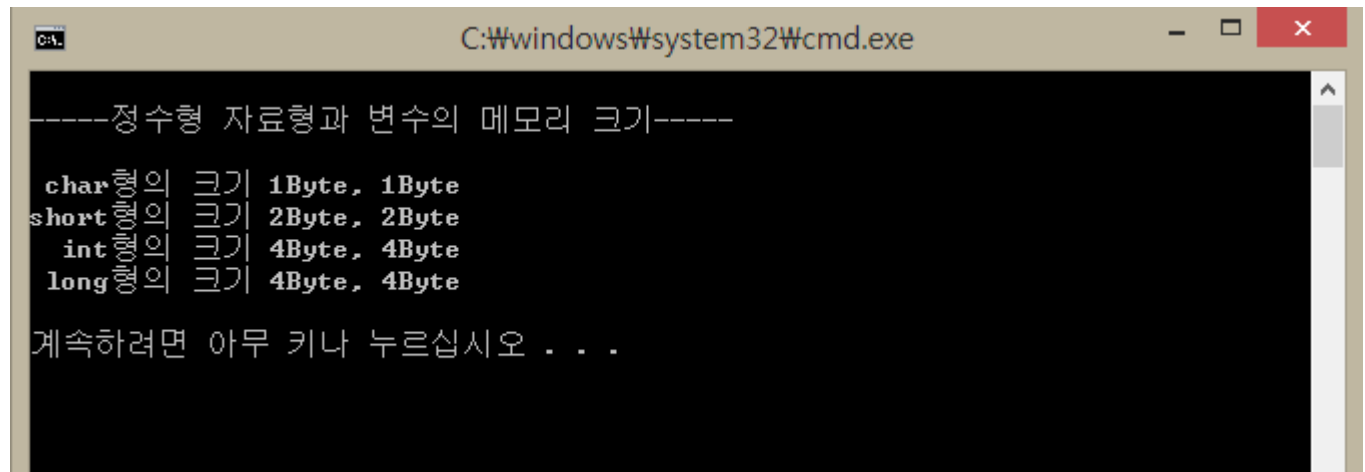
```
#include <stdio.h>

int main(void)
{
    char num1 = 10;
    short num2 = 20;
    int num3 = 30;
    long num4 = 40;

    printf("\n-----정수형 자료형과 변수의 메모리 크기-----\n\n");

    printf(" char형의 크기 %dByte, %dByte \n", sizeof(char), sizeof(num1));
    printf("short형의 크기 %dByte, %dByte \n", sizeof(short), sizeof(num2));
    printf(" int형의 크기 %dByte, %dByte \n", sizeof(int), sizeof(num3));
    printf(" long형의 크기 %dByte, %dByte \n", sizeof(long), sizeof(num4));

    return 0;
} //end main
```



```
-----정수형 자료형과 변수의 메모리 크기-----

char형의 크기 1Byte, 1Byte
short형의 크기 2Byte, 2Byte
 int형의 크기 4Byte, 4Byte
 long형의 크기 4Byte, 4Byte

계속하려면 아무 키나 누르십시오 . . .
```

자료형의 크기

```
#include <stdio.h>

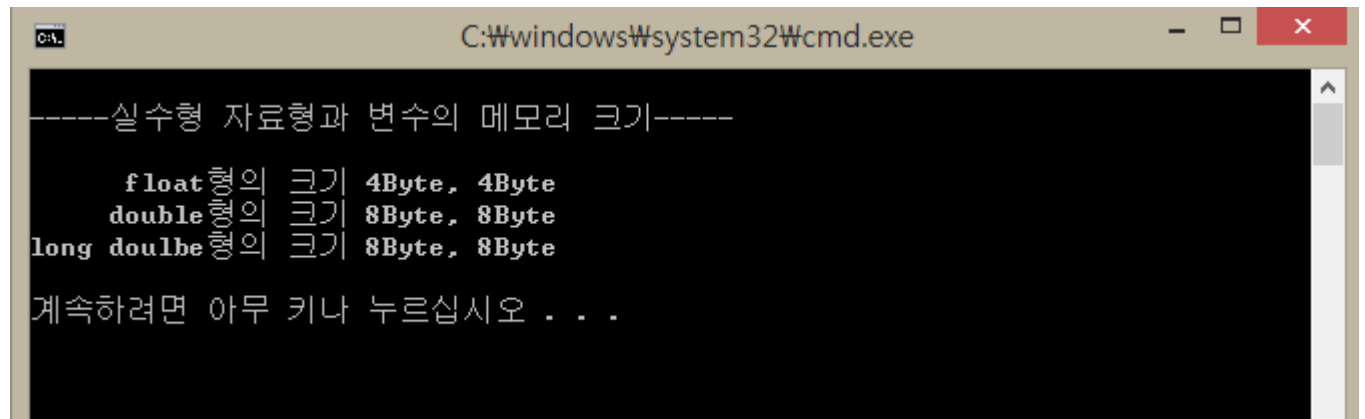
int main(void)
{
    float num5 = 3.14;
    double num6 = 3.15;
    long double num7 = 3.16;

    printf("\n-----실수형 자료형과 변수의 메모리 크기-----\n\n");

    printf("      float형의 크기 %dByte, %dByte \n", sizeof(float), sizeof(num5));
    printf("      double형의 크기 %dByte, %dByte \n", sizeof(double), sizeof(num6));
    printf("long double형의 크기 %dByte, %dByte \n", sizeof(long double), sizeof(num7));

    return 0;

} //end main
```



```
C:\windows\system32\cmd.exe

-----실수형 자료형과 변수의 메모리 크기-----

      float형의 크기 4Byte, 4Byte
      double형의 크기 8Byte, 8Byte
long double형의 크기 8Byte, 8Byte

계속하려면 아무 키나 누르십시오 . . .
```

데이터 표현 범위

자료형	메모리 크기	데이터 표현 범위
char	1Byte(8Bit)	-128 ~ 127
short	2Byte (16Bit)	-32768 ~ 32767
int	4Byte (32Bit)	-2147483648 ~ 2147483647
long	4Byte (32Bit)	-2147483648 ~ 2147483647

➤ 데이터의 표현 범위를 구하는 공식

n은 비트 수(1바이트는 8비트)

$$\boxed{- 2^{n-1}} \sim \boxed{+ 2^{n-1}-1}$$

최소값(MIN)

최대값(MAX)

자료형 데이터 표현 범위

■ limits.h

- 정수형 데이터 표현 최소값(MIN)과 최대값(MAX)상수 제공
- 자동으로 알려주는 함수가 있기 때문에 굳이 계산하지 않아도 된다.

자료형	상수(최소값)	상수(최대값)
char	CHAR_MIN	CHAR_MAX
short	SHRT_MIN	SHRT_MAX
int	INT_MIN	INT_MAX
long	LONG_MIN	LONG_MAX

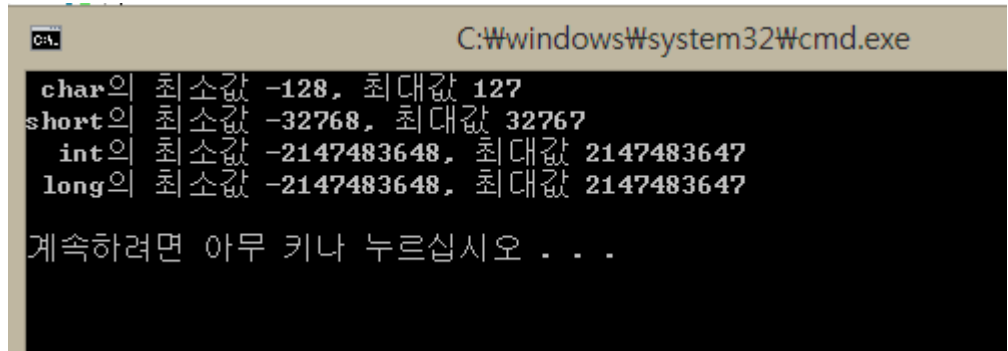
자료형 데이터 표현 범위

```
#include <stdio.h>
#include <limits.h> //정수형의 최소값, 최대값 상수정의

int main(void)
{
    printf(" char의 최소값 %d, 최대값 %d\n", CHAR_MIN, CHAR_MAX);
    printf("short의 최소값 %d, 최대값 %d\n", SHRT_MIN, SHRT_MAX);
    printf(" int의 최소값 %d, 최대값 %d\n", INT_MIN, INT_MAX);
    printf(" long의 최소값 %d, 최대값 %d\n\n", LONG_MIN, LONG_MAX);

    return 0;

} //end main
```



```
C:\Windows\System32\cmd.exe

char의 최소값 -128, 최대값 127
short의 최소값 -32768, 최대값 32767
 int의 최소값 -2147483648, 최대값 2147483647
 long의 최소값 -2147483648, 최대값 2147483647

계속하려면 아무 키나 누르십시오 . . .
```


unsigned 자료형

- unsigned : 0과 양수만을 표현

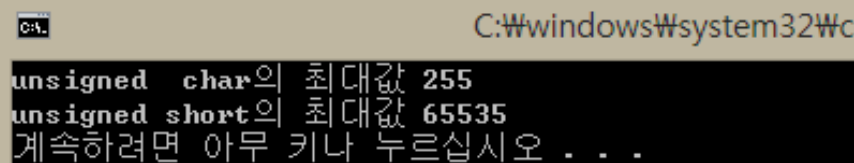
자료형	메모리 크기	데이터 표현 범위
char(signed char) unsigned char	1Byte(8Bit)	-128 ~ 127 0 ~ (127 + 128)
short(signed short) unsigned short	2Byte(16Bit)	-32768 ~ 32767 0 ~ (32767 + 32768)
int(signed int) unsigned int	4Byte(32Bit)	-2147483648 ~ 2147483647 0 ~ (2147483647+2147483648)
long(signed long) unsigned long	4Byte(32Bit)	-2147483648 ~ 2147483647 0 ~ (2147483647+2147483648)

unsigned 자료형

```
#include <stdio.h>
#include <limits.h> //정수형의 최소값, 최대값 상수정의

int main(void)
{
    printf("unsigned char의 최대값 %d\n", UCHAR_MAX);
    printf("unsigned short의 최대값 %d\n", USHRT_MAX);
    return 0;
} //end main
```

자료형	상수(최대값)
unsigned char	UCHAR_MAX
unsigned short	USHRT_MAX
unsigned int	UINT_MAX
unsigned long	ULONG_MAX



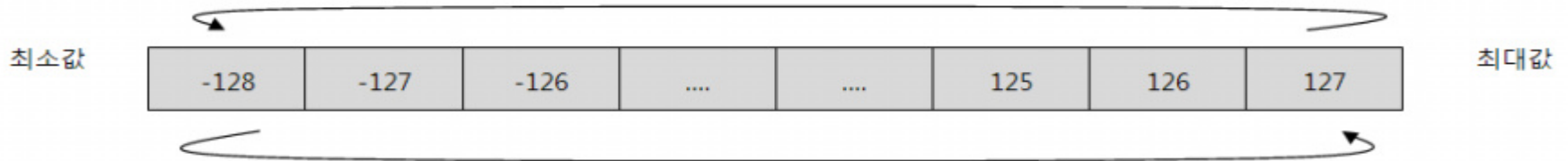
```
C:\windows\system32\cmd.exe
unsigned char의 최대값 255
unsigned short의 최대값 65535
계속하려면 아무 키나 누르십시오 . . .
```

over flow & under flow

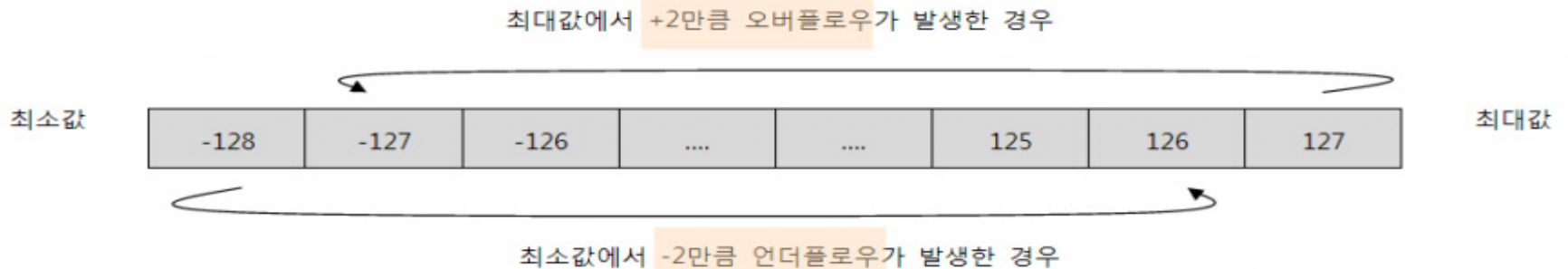
■ over flow

- 자료형에 저장할 수 있는 최대 범위보다 큰 수 저장

최대값에서 +1만큼 오버플로우가 발생한 경우



최소값에서 -1만큼 언더플로우가 발생한 경우



over flow & under flow

```
#include <stdio.h>

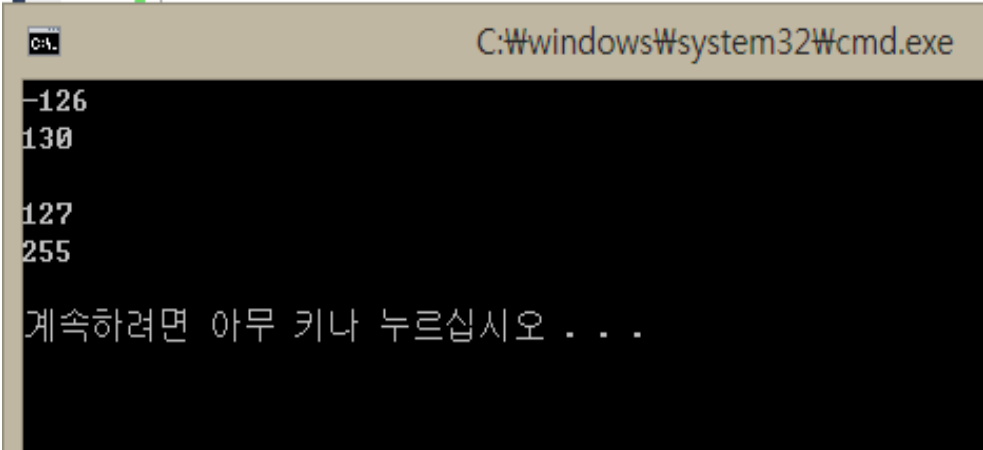
int main(void)
{
    //signed char의 데이터 표현 범위 : -128 ~ 127
    //unsigned char의 데이터 표현 범위 : 0 ~ 255
    signed char num1 = 130; //over flow
    unsigned char num2 = 130;

    printf("%d %n", num1);
    printf("%d %n%n", num2);

    num1 = 127;
    num2 = -1; //under flow

    printf("%d %n", num1);
    printf("%d %n%n", num2);

    return 0;
} //end
```



```
C:\windows\system32\cmd.exe
-126
130
127
255
계속하려면 아무 키나 누르십시오 . . .
```

over flow & under flow

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

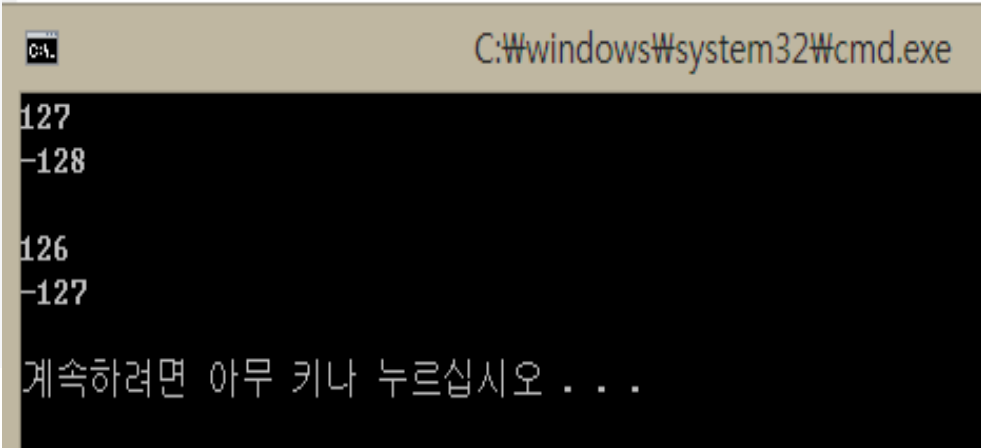
```
int main(void)
{
    //signed char의 데이터 표현 범위 : -128 ~ 127
    //unsigned char의 데이터 표현 범위 : 0 ~ 255
    char num1 = -129;    //under flow
    char num2 = 128;     //over flow

    printf("%d \n", num1);
    printf("%d \n\n", num2);

    num1 = -130;        //under flow
    num2 = 129;         //over flow

    printf("%d \n", num1);
    printf("%d \n\n", num2);

    return 0;
} //end|
```



```
C:\windows\system32\cmd.exe
127
-128

126
-127

계속하려면 아무 키나 누르십시오 . . .
```