C Data Types

Type	**Length**	**Range**
unsigned char	8 bits	0 to 255
char	8 bits	-128 to 127
enum	16 bits	-32768 to 32767
unsigned int	16 bits	0 to 65535
short int	16 bits	-32768 to 32767
int	16 bits	-32768 to 32767
unsigned long	32 bits	0 to 4,294,967,295
long	32 bits	-2,147,483,648 to 2,147,483,647
float	32 bits	3.4×10^{-38} to $3.4 \times 10^{+38}$
double	64 bits	1.7×10^{-308} to $1.7 \times 10^{+308}$
long double	80 bits	3.4×10^{-4932} to $3.4 \times 10^{+4932}$

Type Modifiers

A type modifier alters the meaning of the base type and gives a new type. When the base type is omitted the *int* type is assumed. Various type modifiers are:

unsigned

The ***unsigned*** keyword indicates that the most significant bit of an integer variable represents a data bit rather than a signed bit. This keyword is optional and can be used with any of the character and integer types *char, int, long, short.* If it unsigned it omitted the value defaults to signed value.

short

The ***short*** keyword designates a 16-bit integer. The ***short*** keyword can be preceded by either the keyword *signed* or the keyword *unsigned*. The *int* keyword is optional and can be omitted.

long

The ***long*** keyword designates a 32-bit integer. It can be preceded by either the keyword *signed* or the keyword *unsigned*. The *int* keyword is optional and can be omitted.

Overflow situations

Image a situation where an integer is defined and its value is continuously incremented (or decremented). At some time the value will be on the edge of the range defined above. What happens if value is changed from the edge value? The value overflows to the other edge.

e.g.

```
int a;
a = 32767;
a = a +1;
gives the value of a as -32768 (and not 32768 as expected)
```