

## Simple Scalar Data Types (32-bit compilers)

### Commonly used

int	integer in the range -2,147,483,648 to 2,147,483,647 or $-(2^{31})$ to $2^{31}-1$
double	floating-point decimal in the range $\pm 1.7 \times 10^{-308}$ to $\pm 1.7 \times 10^{308}$ 15-digit precision
char	character
bool	true or false
string	s series of char variables

### Mathematical Operators

(some operators omitted)

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Unary increment
--	Unary decrement

### Logical Operators

&&	and
	or
!	not

### Order of Operations

(some operators omitted)

Higher	!, ++, --
2	*, /, %
3	+, -
4	==, !=
5	&&
6	
Lower	=

## Input and Output

The `cin` and `cout` objects provide high-level input and output in C++, respectively. `cin`, which verifies that input is of the correct type, is used with the extraction operator (`>>`):

```
cin >> x >> y >> z >> ch;
```

`cout` provides output functionality by means of the insertion (`<<`) operator:

```
cout << "This is a number: " << n;
```

## Formatting Output and Input with the `iomanip` Library `<iomanip>`

Function	Computes
<code>setbase(int base)</code>	Sets <i>basefield</i> to hex, dec or oct depending on <i>base</i> parameter.
<code>setfill(char ch)</code>	Fill the white space with character <i>ch</i>
<code>setprecision(int n)</code>	Set decimal precision to <i>n</i> places
<code>setw(int w)</code>	Sets a value to be used as the <i>field width</i> ( <i>w</i> ) for the next insertion operation.
<code>resetiosflags(flag)</code>	Unsets the format flags specified by parameter.
<code>setiosflags(flag)</code>	Sets the format flags specified by parameter.

## ios::base flags options for setiosflags and resetiosflags

Function	Computes
ios_base::boolalpha	input/output <b>bool</b> objects as alphabetic names ( <b>true</b> , <b>false</b> ).
ios_base::fixed	output floating point values in fixed-point notation.
ios_base::left	the output is filled at the end enlarging the output up to the <i>field width</i> .
ios_base::right	the output is filled at the beginning enlarging the output up to the <i>field width</i> .
ios_base::scientific	output floating-point values in scientific notation.
ios_base::showpoint	output floating-point values including always the decimal point.
ios_base::showpos	output non-negative numeric preceded by a plus sign (+).
ios_base::uppercase	output uppercase letters replacing certain lowercase letters.

Output is unformatted by default but may be modified by using the manipulators found in the header file `iomanip`:

```
cout << setiosflags(ios_base::fixed)      // do not use E notation
      << setiosflags(ios_base::showpoint) // always show decimal pt
      << setprecision(2);                // round to two decimal places
```

Once used, the manipulators above “stick” for the rest of the program—that is, they apply to all numeric output thereafter unless the `resetiosflags` manipulator is called.

To output a number right aligned in a certain size field, use the `setw` manipulator. They apply to only the next output.

```
cout << setw(10)
      << number1
      << number2;
```

Ten columns are allocated for the next output by the `setw(10)` manipulator. Number 1 would appear right aligned within those 10 columns and number2 would appear immediately to the right of number1 (no spaces between them). The previous formatting commands would apply to both number1 and number2.