Type Casting

A cast is a special operator that forces one data type to be converted into another. As an operator, a cast is unary and has the same precedence as any other unary operator.

Implicit Type Conversion Also known as 'automatic type conversion'

- Done by the compiler on its own, without any external trigger from the user.
- Generally takes place when in an expression more than one data type is present. In such condition type conversion (type promotion) takes place to avoid lose of data.
- All the data types of the variables are upgraded to the data type of the variable with largest data type.

```
• bool -> char -> short int -> int ->
•
• unsigned int -> long -> unsigned ->
•
• long long -> float -> double -> long double
```

- It is possible for implicit conversions to lose information, signs can be lost (when signed is implicitly converted to unsigned), and overflow can occur (when long long is implicitly converted to float).

main.cpp Ctrl+S

```cpp
// An example of implicit conversion

#include <iostream>
using namespace std;

int main()
{
    int x = 10; // integer x
    char y = 'a'; // character c

    // y implicitly converted to int. ASCII
    // value of 'a' is 97
    x = x + y;

    // x is implicitly converted to float
    float z = x + 1.0;

    cout << "x = " << x << endl
         << "y = " << y << endl
         << "z = " << z << endl;

    return 0;
}
```

Explicit Type Conversion: This process is also called type casting and it is user-defined. Here the user can typecast the result to make it of a particular data type.
In C++, it can be done by two ways:

Converting by assignment: This is done by explicitly defining the required type in front of the expression in parenthesis. This can be also considered as forceful casting.

```
(type) expression
```

where *type* indicates the data type to which the final result is converted.

main.cpp Ctrl+S

```cpp
// C++ program to demonstrate
// explicit type casting

#include <iostream>
using namespace std;

int main()
{
    double x = 1.2;

    // Explicit conversion from double to int
    int sum = (int)x + 1;

    cout << "Sum = " << sum;

    return 0;
}
```