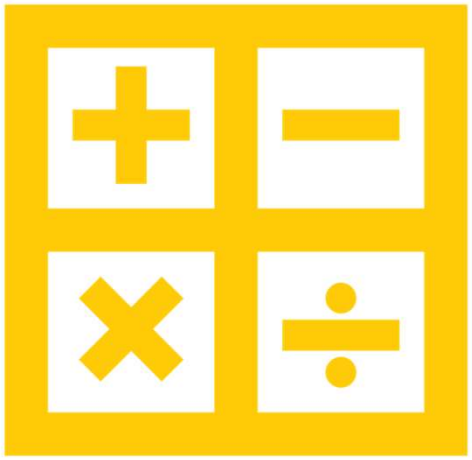
Type Conversion(Casting)

- The process of converting one data type to another
- Two Types:
  - Implicit (Done by Compiler/Automatic)
  - Explicit (Done by Programmer/Manual)

When is the casting actually performed?

- Arithmetic operations are normally performed over the same types of operands
- But when we have operands of different data, like one operand is character and other one is integer
  - C++ will convert the one operand to be the type of other and then evaluate the expression



IMPLICIT TYPE CONVERSION

- The Data Type conversion that is done by compiler automatically
  - 1. Automatic (Lower to Higher)

Lower ⟶ Higher

Char, short int	int	float	double	long double

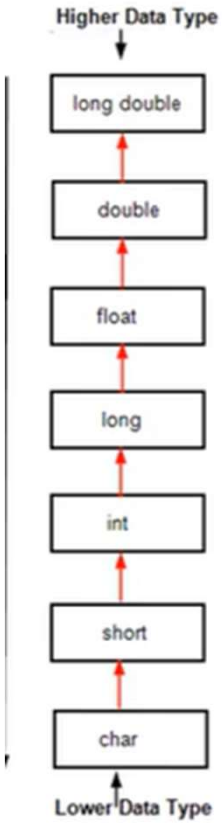
  - 2. By Assignment (Right to Left)

IMPLICIT TYPE CONVERSION(LOWER TO HIGHER)

- 2+5.6+9 so what should be the resultant data type?
- 2.0+5.6+9.6= DOUBLE TYPE

- ‘a’+1, What should be the result?
- Result will be 98

Lower to Higher



Check data type of a variable or value

- #include<typeinfo>
- typeid(variable/expression).name()

- Example:
- cout<<typeid(5.9+6).name();

IMPLICIT CONVERSION RIGHT TO LEFT

- float a=12.5;
- int b=13;
- int sum=a+b;
- What will be the result of sum?
- First Lower to Higher, then Left to Right
- a+b=12.5+13.0=25.5 (which is a float value)
- sum=25 (Hence we loose information)

IMPLICIT CONVERSION RIGHT TO LEFT

- char c1='a'+1;
- char c1=97+1;
- char c1=98;
- c1 will have value of b because of left to right conversion
- What if we write int c1='a';?
- Obviously we will get 97

EXPLICIT TYPE CONVERSION

- The Type of conversion that you as a programmer specify and you want to do.

- char c1=(char)97
- float f1=(float)9
- cout<<(double)5.3/4

Second method for explicit casting


```
#include <iostream>
using namespace std;
int main()
{
    float f = 3.5;

    // using cast operator
    int b = static_cast<int>(f);

    cout << b;
}
```

Relational Operators

- C++ Relational operators specify the relation between two variables by comparing them.
- If the results after comparison b/w two variable is true it will return 1, else it will return 0 for false.
- There are six relational operators:
- Less than (<)
- Less than or equal to (<=)
- Greater than (>)
- Greater than or equal to (>=)
- Equals Equals to (==)
- Not equal to (!=)

```
1  #include <iostream>
2  using namespace std ;
3  int main ()
4  {
5
6      cout <<"10 > 100     : "     << (10 > 100) <<endl ;
7      cout <<"20 >= 20     : "     << (20 >= 20) <<endl ;
8      cout <<"10 < 100     : "     << (10 < 100) <<endl ;
9      cout <<"30 <= 40     : "     << (30 <= 40) <<endl ;
10     cout <<"30 != 30     : "     << (30 != 30) <<endl ;
11     cout <<"40 == 30     : "     << (40 == 30) <<endl ;
12
13     system ("PAUSE") ;
14     return 0 ;
15 }
```

```
10 > 100          : 0
20 >= 20          : 1
10 < 100          : 1
30 <= 40          : 1
30 != 30          : 0
40 == 30          : 0
```