TYPE CONVERSION

Type conversion

- Type conversion or typecasting refers to changing an entity of one data type into another.
- **Implicit Type Conversion**

  Implicit type conversion is an automatic type conversion by the compiler.

  The type conversions are automatic as long as the data types involved are built-in types.
- **Example**

```
int y;
float x=123.45;
y = x;
```

- **Explicit Type Conversion**

  Automatic type conversion for user defined data types is not supported by the compiler hence the conversion routines are ought to be specified explicitly. Three types of situations might arise in the data conversion between incompatible types.

  - Conversion from basic type to class type.
  - Conversion from class type to basic type
  - Conversion from one class type to another class type.

Basic to Class Type

Ex: time T1; // object T1 created

int duration =85;

T1=duration; // int to class type

- This is done by using constructors in the respective classes. In these types the '=' operator is overloaded.

Basic → Class type

```
class time
{
    int hrs;
     int mins;
  public:
    ....
    ....
    time(int t)                    // constructor
    {
        hours = t/60;              // t in minutes
        mins  = t%60;
    }
};
```

Built-in variable (duration) assigned to member variables

Constructor does the job

The following conversion statements can be used in a function:

```
time T1;                           // object T1 created
int duration = 85;
T1 = duration;                     // int to class type
```

Class → Basic type

Overloaded casting operator is used to convert data from a class type to a basic data type.

Syntax:

operator typename()

{

statements.....

}

This function converts a class type to specified type name.

For example **operator int()** converts the class object to integer data type.

Conditions for a casting operator function

- It must be a class member.
- It must not specify a return type.
- It must not have any arguments

```
class time
{
int min,sec;
public:
time(int n)
{
min = n/60;
sec=n%60;
}
operator int()  //Casting Operator
{
int x;
x= min * 60 + sec;
return x;
}
operator float()    // Casting Operator
{
float y;
y = min + sec/60;
return y;
}
};
```

```
void main()
{
time t1(160);
int n = t1; //conversion
float x = t1; //conversion
}
```

One class type to another class type

```
#include <bits/stdc++.h>
using namespace std;
class A { //source class
string a;
public:
A(){a = "God is Great"; }
string get_string() {return a; }
void display() {cout << a << endl; }
};
class B { // Destination class
string b;
public:
void operator=(A a){b = a.get_string(); }
void display() {cout << b << endl; }
};
int main()
{
A a; B b; b = a;
// CLass type conversion using operator overloading
a.display(); b.display();
 return 0;
}
```