**Operator overloading**

**What:**

It is a type of polymorphism in which an operator is overloaded to give user defined meaning to it. Overloaded operator is used to perform operation on user-defined data type.

**Note:**

When we use,

1. Assignment operator(=), reference(&) operator we use the same type as the return type.

2. When relational operator (< , > , <=, >=, != ) is been used we return bool.

3. When performing arithmetic operation of same type, same type is returned.

**operator which CANNOT be overloaded,**

1. Scoper resolution operator (::)

2. Size of operator (sizeof)

3. member selector operator (.)

4. member pointer selector operator (\*)

**#01\_Pgm**

#include <iostream>

using namespace std;

class Distance {

private:

int feet; // 0 to infinite

int inches; // 0 to 12

public:

// required constructors

Distance(){

feet = 0;

inches = 0;

}

Distance(int f, int i){

feet = f;

inches = i;

}

// method to display distance

void displayDistance() {

cout << "F: " << feet << " I:" << inches <<endl;

}

// overloaded minus (-) operator

Distance operator- () {

feet = -feet;

inches = -inches;

return Distance(feet, inches);

}

// overloaded < operator

bool operator <(const Distance& d) {

if(feet < d.feet) {

return true;

}

if(feet == d.feet && inches < d.inches) {

return true;

}

return false;

}

};

int main() {

Distance D1(11, 10), D2(5, 11);

if( D1 < D2 ) {

cout << "D1 is less than D2 " << endl;

} else {

cout << "D2 is less than D1 " << endl;

}

return 0;

}

Output;

D2 is less than D1