ASSIGNMENT

COURSE TITLE	:	Object Oriented Programming
COURSE CODE	:	CSE 111
ASSIGNMENT NO.	:	02
SUBMISSION DATE	:	N/A

SUBMITTED TO

NAME	:	N/A
DEPT. OF	:	N/A Bangladesh University of Business & Technology (BUBT)

SUBMITTED BY

NAME	:	Shadman Shahriar
ID NO.	:	20245103408
INTAKE	:	53
SECTION	:	9
PROGRAM	:	B.Sc. Engg. in CSE

Object Oriented Programming Assignment 2

Theory

Write down the difference between **pass value** and **pass by reference** with code example.

Ans. In **pass-by-value**, a *copy* of the variable is passed to the function. Modifying the parameter inside the function **does not** affect the original variable.

```
#include <iostream>
using namespace std;

void edit(int x)
{
    x = 10;
    cout << "Value inside the function: " << x << endl;
}

int main()
{
    int a = 5;
    cout << "Before function call: " << a << endl;
    edit(a);
    cout << "After function call: " << a << endl;
    return 0;
}
```

In **pass-by-reference**, the *actual variable* (not a copy) is passed to the function. Any changes made to the parameter inside the function **will affect** the original variable.

```cpp
#include <iostream>
using namespace std;

void edit(int &x)
{
    x = 10;
    cout << "Value inside the function: " << x << endl;
}

int main()
{
    int a = 5;
    cout << "Before function call: " << a << endl;
    edit(a);
    cout << "After function call: " << a << endl;
    return 0;
}
```

Program

Design a class called `Distance` with an attribute `length`. Now **overload** + operator to add two `Distance` objects. If the `Distance` objects aren't equal, make the lesser object equal to the greater.

Ans. Here is a C++ program that satisfies the conditions above:

```
#include <iostream>
using namespace std;

class Distance
{
public:
    float length;
    Distance(float l)
    {
        length = l;
    }

    Distance operator+(Distance &other)
    {
        if (length < other.length)
        {
            length = other.length;
        }
        else
        {
            other.length = length;
        }
        return Distance(length + other.length);
    }
};

int main()
{
```

```
    Distance d1(10.5), d2(20.3);
    Distance d3 = d1 + d2;

    cout << "d1 length: " << d1.length << endl;
    cout << "d2 length: " << d2.length << endl;
    cout << "d3 length: " << d3.length << endl;

    return 0;
}
```

Output: The code yields the following output in the terminal:

```
d1 length: 20.3
d2 length: 20.3
d3 length: 40.6
```