G. H. Raisoni College Of Engineering And Management, Wagholi Pune Assignment no :- 3 2021- 2022			
Department	CE [SUMMER 2022 (Online)]		
Term / Section	III/B	Date Of **submission**	04-10-2021
Subject Name /Code	Object Oriented Programming/ UTIL201/UITP201		
Roll No.	SCOB77	Name	Pratham Rajkumar pitty
Registration Number	2020AC0E1100107		

Assignment No. 3

Aim :→ Create two classes DM and DB which stores values of distances.

DM stores distances in meters and centimeters and DB in feet and inches.

Write a program that can read values for the class objects and add one object of DM with another object of DB. Use a friend function to carry out addition operation.

Theory:

C++ Friend Function

- A Friend Function of a class is defined outside that class scope but it has the right to access all private and protected members of the class Even though the prototypes for Friend Functions appear in the class definition, friend are not member Function

- If a function is defined as a Friend function then, the private and protected data of a class can be accessed using the function.

- The compiler knows a given function is a friend function by the use of the keword Friend.

- For accessing the data, the declaration of a Friend Function should be made inside the body of the class (can be anywhere inside class either in private or public section) starting with keword Friend.

Declaration of Friend Function in C++

```
class class_name
{
.....
Friend return_type function_name (argument/s);
.....
}
```

We can define the Friend function as a normal function to access the data of the class.
No keyword is used in the defination.

consider in above case.

```
class class_Name
{
....
Friend return_type function Name (argument/s);
...
}

retor_type Function Name (argument/s)
{
.....
// Poivate and protected data of classname can be accessed from this Function because it is a Friend Function of class_Name.
}
```

Program code

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

class DB;

class DM{
   public:
    float centimeters,meters;

   void Dist(){
      cout<<"\nIn Centimeters it is = "<< centimeters <<endl;
      cout<<"\nIn Meters it is = "<< meters <<endl;
      }

   DM(float CentiMeters, float Meters ){
    centimeters = CentiMeters;
      meters = Meters;
   }
   DM(){
    centimeters = 0;
      meters = 0;
   }
    friend void add(DM &,DB &, DM &d);
};

class DB{
   public:
    float ft,inc;
```

```
    void Dist(){
       std::cout<<"\nIn feet it is = "<< ft << std::endl;
       std::cout<<"\nIn inches it is = "<< inc << std::endl;
       }

    DB(float FT, float Inc ){
       ft = FT;
       inc = Inc;
}

};

  void add(DM &d2, DB &d1, DM &d)
{


    int c=(d2.meters*100+d2.centimeters+d1.ft*30.48+d1.inc*2.54);
    if(c>=100){
       d.meters=c/100;
       d.centimeters=c%100;
    }
    else
    {
       d.meters=0;
       d.centimeters=c;
    }
    cout<<"------------------------------------------------------------------\n";
    cout<<"After adding  one object of DM with another object of DB "<<endl;
    d.Dist();
}
```

```
int main(){
cout<<"------------------------------------------------------------------\n";
cout<<"\nSCOB77_PRATHAM PITTY_OOP_Assignment no 3\n";
cout<<"------------------------------------------------------------------\n";
   DM DM1= DM(35.0,4.7);
   DB DB1= DB(5.5,8.4);


   DM1.Dist();


   DB1.Dist();
   DM d = DM();
   add(DM1, DB1,d);
   cout<<"------------------------------------------------------------------\n";
}
```