```
#include<iostream>
using namespace std;
class equality
{
int x,y;
public:
equality()
{
        x=0;
        y=0;
}
equality(int a,int b)
{
x=a;
y=b;
}
bool operator==(equality o)
{
if((x==o.x) && (y==o.y))
return 1;
else
return 0;
}
};
int main()
{
equality o1(3,3),o2(2,3);
if(o1==o2)
{
// or if(o1.operator==(o2))
//{
cout<<"\n Values of x and y for o1 and o2 are same";
}
else
{
cout<<"\n Values of x and y for o1 and o2 are not same";
}
}
#include<iostream>
using namespace std;
class equality
{
int x,y;
public:
equality(int a,int b)
{
x=a;
y=b;
}
```

```
friend bool operator==(equality,equality);
};
bool operator==(equality o1,equality o2)
{
if((o1.x==o2.x) && (o1.y==o2.y))
return 1;
else
return 0;
}
int main()
{
equality o1(2,3),o2(3,3);
if(o1==o2)
{
// or if(operator==(o1,o2))
cout<<"\n Values of x and y for o1 and o2 are same";
}
else
{
cout<<"\n Values of x and y for o1 and o2 are not same";
}
}
#include<iostream>
using namespace std;
class greater1
{
int x;
public:
greater1(int a)
{
x=a;
}
bool operator>(greater1 o)
{
if(x>o.x)
return 1;
else
return 0;
}
friend bool operator!=(greater1,greater1);
};
bool operator!=(greater1 o1,greater1 o2)
{
 if(o1.x!=o2.x)
 {
    return 1;
 }
 else
 {
```

```
return 0;
 }
}
int main()
{
greater1 o1(3),o2(2);
if(o1>o2)
{
// or if(o1.operator>(o2))
//{
cout<<"\n o1 is greater than o2";
}
else
{
cout<<"\n o2 is greater than o1";
}
if(o1!=o2)
{
cout<<"\n o1 is not equal to o2";
}
else
{
   cout<<"\no1 and o2 are equal";
}
}
#include<iostream>
using namespace std;
class equality
{
int x;
public:
equality()
{
x=0;
}
equality(int a)
{
x=a;
}
bool operator>(equality o){
if((x>o.x))
return 1;
else
return 0;
}
equality operator+(equality o)
{
equality temp;
temp.x= x + o.x;
return temp;
}
friend bool operator==(equality o1,equality o2);
void showdata(){
cout<<x;
}

};
bool operator==(equality o1,equality o2){

if((o1.x==o2.x))
return 1;
else
return 0;

}
int main()
{
equality o1(3),o2(2),o3;
o3 = (o1>o2)+(o1==o2);
o3.showdata();
}
#include<iostream>
using namespace std;
class logical
{
int x;
public:
logical()
{
        x=0;
}
logical(int a)
{
x=a;
}
bool operator&&(logical o)
{
return (x&&o.x);
}
friend bool operator||(logical,logical);
};
bool operator||(logical o1,logical o2)
{
        return(o1.x||o2.x);
}
int main()
{
logical o1(2),o2(0);
```

```
int x=o1&&o2;
cout<<x<<endl;
int y=o1||o2;
cout<<endl<<y;
return 0;
}
//Basic to class is done with the help of constructor function
#include<iostream>
using namespace std;
class Time
{
private:
int h,m;
public:
Time()
{
h=m=0;
}
Time(int t)
{
h=t/60;
m=t%60;
}
void show_data()
{
cout<<h<<"hrs"<<m<<"mins";
}
};
int main()
{
int min;
cout<<"\n Enter the minutes:";
cin>>min;
Time T1;
T1=min;//It will call the constructor function after taking minutes(basic type) and T1 is of class type
T1.show_data();
}
//Basic to class is done with the help of constructor function
#include<iostream>
using namespace std;
class Distance1
{
private:
int km,m;
public:
Distance1()
{
km=m=0;
}
```

```
Distance1(int d)
{
km=d/1000;
m=d%1000;
}
void show_data()
{
cout<<km<<"Kilometers"<<m<<"meters";
}
};
int main()
{
int d;
cout<<"\n Enter the distance:";
cin>>d;
Distance1 d1;
d1=d;//It will call the constructor function after taking minutes
d1.show_data();
}
#include<iostream>
using namespace std;
class circle
{
private:
float area;
public:
circle()
{
area=0.0f;
}
circle(float d)
{
area=3.14*d*d;
}
void show_data()
{
cout<<area;
}
};
int main()
{
float radius;
cout<<"\n Enter the radius";
cin>>radius;
circle d1;
d1=radius;
d1.show_data();
}
```