Type conversion using operator overloading concepts. (Source from Internet)

1. Conversion from basic data type to user-defined data type (class type)
2. Conversion from class type to basic data type
3. Conversion from one class type to another class type

```
#include<iostream.h>
#include<conio.h>
class data
{
    int x;
    float f;
    public :
    data()
    {
    x=0;
    f=0;
    }
    data ( float m)
    {
    x=2;
    f=m;
    }
    void show()
    {
    cout<<"\n x= "<<x<<" f="<<f;
    cout<<"\n x= "<<x<<"f="<<f;
    }
};
int main()
{
    clrscr();
    data z;
    z=1;
    z.show();
    z=2.5;
    z.show();
    return 0;
}
```

```
#include<iostream.h>
#include<conio.h>
class data
{
   int x;
   float f;
   public:
   data()
   {
   x=0;
   f=0;
   }
   operator int()
   {
   return (x);
   }
   operator float()
   { return f; }
   data ( float m)
   {
   x=2;
   f=m;
   }
   void show()
   {
   cout<<"\n x= "<<x<<"f="<<f;
   cout<<"\n x= "<<x<<"f="<<f;
   }
};
int main()
{
   clrscr();
   int j;
   float f;
   data a;
   a=5.5;
   j=a; // operator int() is executed
   f=a; // operator float() is executed
   cout<<"\n Value of j :"<<j;
   cout<<"\n Value of f :"<<f;
   return 0;
}
```

```
#include<iostream.h>
#include<conio.h>
class minutes
{
    int m;
    public:
    minutes()
    { m=240; }
    get()
    { return (m);}
    void show()
    { cout<<"\n Minutes="<<m; }
};
class hours
{
    int h;
    public:
    void operator = (minutes x);
    void show()
    { cout<<"\n Hours="<<h; }
};
void hours:: operator = (minutes x)
{
    h=x.get()/60;
}
int main()
{
    clrscr();
    minutes minute;
    hours hour;
    hour=minute;
    minute.show();
    hour.show();
    return 0;
}
```