Unary operator overloading

Lecture 13

Unary operator overloading

- Unary operator can be overloaded as a member function with no arguments or as a global function with one argument

Example

```
class point
{  int x,y,z;
  public:
  point(int d,int e, int f)
 { x=d; y=e; f=z; }
void display()
{cout<<x<<y<<z; }
void operator –()
{ x=-x; y=-y; z=-z; }
};
```

```
void main()
{
  point a(10,20,15);
  a.display();
  -a ;
  a.display();
}
```

Class assignment

- What changes to operator function are required to make a call as follows

  -a.display();

Solution

```
class point
{ int x;
  int y;
  int z;
  public:
  point(int c,int d,int f)
  { x=c;y=d;z=f; }
  void display()
  { cout<<"\n"<<x<<" "<<y<<"
   "<<z;}
  point & operator -()
  { x=-x; y=-y;z=-z;
    return *this; }
};
```

```
void main()
{
 clrscr();
 point p(10,20,30);
 p.display();
 (-p).display();
 getch();
}
```

Using global function

- We need to pass one argument of the object type to do this.

Example

```
class point
{
  friend void operator -(point &);
  private:
  int x;
  int y;
  int z;
  public:
  point(int c,int d,int f)
  { x=c;y=d;z=f; }
  void display()
  { cout<<"\n"<<x<<" "<<y<<" "<<z;}
};
```

```
void operator -(point
     & obj)
{ obj.x=-obj.x;
 obj.y=-obj.y; obj.z=-
     obj.z ;
}
```

```
void main()
{ point p(10,20,30);
 p.display();
-p;
p.display();
}
```

Class assignment

- What changes to operator function (written using global function) are required to make a call as follows

  -a.display();

Solution

```
class point
{
  friend point & operator -(point);
  private:
  int x;
  int y;
  int z;
  public:
  point(int c,int d,int f)
  { x=c;y=d;z=f; }
  void display()
  { cout<<"\n"<<x<<" "<<y<<" "<<z;}
};
```

```
point & operator -(point
obj)
{ static point
obj2(0,0,0);
 obj2.x=-obj.x;
 obj2.y=-obj.y;
 obj2.z=-obj.z;
 return obj2;
}
```

```
void main()
{ clrscr();
 point p(10,20,30);
 p.display();
 p=-p;
 p.display();
 getch(); }
```