

Overloading stream insertion and stream extraction operators

Lecture 15

Global functions

- If the left operand is not the object, (it may be object of different class or a fundamental type), then the operator is overloaded using a global function

$5+p$

- To allow commutative functioning of operator
 $p+5$ as well as $5+p$

What does stream insertion operator overloading mean?

```
class point { };  
void main()  
{  
    point p, q;  
    cin>>p;  
    cout<<p;  
}
```

Stream operators are overloaded as global functions

- As the left of the operator is not an object of point (user defined) class, these operators are overloaded using global functions

```
cin>>p;
```

Example

```
class point
```

```
{ int x; int y; int z;
```

```
  public:
```

```
  point(int c,int d,int f) { x=c;y=d;z=f; }
```

```
  friend ostream & operator <<(ostream &,point obj);
```

```
};
```

```
ostream & operator <<(ostream & x,point obj)
```

```
{ x<<"\n New point : "<<obj.x<<" "<<obj.y<<" "<<obj.z;
```

```
  return x; }
```

```
void main()
```

```
{ point p(10,20,30); cout<<p; }
```

Example

```
class point
```

```
{ int x; int y; int z;
```

```
  public:
```

```
  point(int c,int d,int f) { x=c;y=d;z=f; }
```

```
  friend istream & operator >>(istream &,point &);
```

```
};
```

```
istream & operator >>(istream & x,point & obj)
```

```
{ x>>"\n New point : ">>obj.x>>" ">>obj.y>>" ">>obj.z;
```

```
  return x; }
```

```
void main()
```

```
{ point p(10,20,30); cin>>p; }
```