# Preklopljeni operatori

- Binarni operator @ ima dva operanda, pa može biti preklopljen kao:
  - nestatička funkcija članica klase *X* sa jednim parametrom:

```
T X::operator@ (Y);
```

tada se operacija *x@y* tumači kao

```
x.operator@ (y)
```

- prvi operand je objekat čija se funkcija članica poziva, a drugi operand se prenosi kao argument

  - funkcija nečlanica klase *X* sa dva parametra:

```
T operator@ (X, Y);
```

tada se operacija *x@y* tumači kao

```
operator@ (x,y)
```

- oba operanda se prenose kao argumenti

# Preklopljeni operatori

- Kada traži preklopljeni operator za date tipove operanada, isto kao i za ostale pozive funkcija, prevodilac sprovodi postupak *potrage zavisne od argumenata* (*argument dependent lookup*, *ADL*); pojednostavljeno rečeno, ovaj postupak traži funkciju, uključujući i preklopljen operator, ne samo prema uobičajenim pravilima potrage za nekvalifikovanim imenom u tekućoj oblasti važenja, nego i po klasama kojima pripadaju argumenti (i njihovim osnovnim klasama), ali i po prostorima imena kojima pripadaju te klase
- Ovo omogućava da preklopljeni operatori budu deklarisani u različitim prostorima imena, a da ipak budu odabrani za argumente određenog tipa; na primer:

```
namespace complex {

  class complex {
    …
    friend std::ostream& operator<< (std::ostream& os, const complex& c) {
      return os<<'('<<c.real<<','<<c.imag<<')';
    }

  };

}

int main () {
  complex::complex c1, c2;
  …
  std::cout<<c1<<‘,’<<c2;
}
```