Korisnički definisane konverzije

- Konverzionim konstruktorom klase *X* može se definisati korisnička konverzija iz bilo kog drugog tipa, pa i ugrađenog (neklasnog) tipa, u tip *X*, ali ne i obrnuto, iz tipa *X* u neki ugrađeni tip
- Takva konverzija može se definisati *operatorom konverzije*, kao nestatičkom operatorskom funkcijom članicom klase *X*, koja nema parametre, nema eksplicitan povratni tip, i ima ime u kome se koristi deklarator tipa (*type-id*); ovakva funkcija treba da vrati vrednost datog tipa:

```
class X {
public:
  operator int ();
  operator Y* ();
};
```

Korisnička konverzija iz tipa *X* u tip *int*

Korisnička konverzija iz tipa *X* u tip *Y**

- Kao *type-id* može se navesti bilo koji fundamentalni ili složeni tip, ali se ne mogu upotrebljavati simboli za niz *[]* i funkciju *()*, osim indirektno, kroz *typedef* sinonime, s tim da odredišni tip ne može biti niz ili funkcija čak ni tako
- I ova konverzija može se vršiti eksplicitno, bilo kojim operatorom konverzije, ali i implicitno, gde god se vrši implicitna konverzija; na primer:

```
X x;
int i = x;
Y* py = x;
```

Korisnički definisana implicitna konverzija iz tipa *X* u tip *int*; poziva se *x.operator int()*

Korisnički definisana implicitna konverzija iz tipa *X* u tip *Y**; poziva se *x.operator Y*()*

Copyright 2018-2024 by Dragan Milićev

Korisnički definisane konverzije

- Ukoliko se želi zabraniti implicitna korisnički definisana konverzija definisana konstruktorom ili operatorom konverzije, odgovarajuća funkcija (konstruktor ili operator) označava se specifikatorom *explicit*; takav konstruktor nije više konverzioni konstruktor, jer ne definiše implicitnu konverziju; na primer:

```
class X {
public:

  explicit X (bool);
  explicit operator bool ();

};

X f (X x) {
  bool b = x;

  return false;
}
```


Copyright 2018-2024 by Dragan Milićev