Podrazumevani konstruktor

- Ako klasa nema nijedan eksplicitno deklarisan konstruktor (tj. korisnički definisan konstruktor), prevodilac će implicitno deklarisati jedan podrazumevani konstruktor koji je javan, *inline* i koji vrši podrazumevanu inicijalizaciju podobjekata osnovnih klasa i objekata članova. Na primer:

```
struct X {
  …
};

struct Y {
  X x1, x2;
  Y () {}
  …
};

X x;
```

Ovde nema deklarisanih konstruktora, prevodilac automatski generiše podrazumevani konstruktor

Sve ovo je u redu, jer klasa *X* ima podrazumevani konstruktor

- Ako klasa ima neki konstruktor (pa prevodilac ne generiše implicitni podrazumevani konstruktor), programer ipak može forsirati automatsko generisanje podrazumevanog konstruktora koji bi prevodilac implicitno generisao specifikatorom *=default*:

```
struct X {
  X (int);
  X () = default;
};
```

Forsiranje generisanja podrazumevanog konstruktora sa podrazumevanom inicijalizacijom podobjekata

- Može se i sprečiti automatsko generisanje ovog podrazumevanog konstruktora, ako se on označi kao obrisan specifikatorom *=delete*. Tada će prevodilac sprečiti pokušaj inicijalizacije svakog objekta te klase koja bi zahtevala poziv podrazumevanog konstruktora
- Ukoliko prevodilac implicitno deklariše podrazumevani konstruktor ili je on eksplicitno deklarisan kao *=default*, a neki od podobjekata nema podrazumevanu inicijalizaciju (npr. nema podrazumevani konstruktor, ili je referenca ili konstantni objekat i slično), prevodilac će smatrati ovaj podrazumevani konstruktor obrisanim: to znači da će prijaviti grešku samo ako se pravi objekat ove klase za koji se traži podrazumevana inicijalizacija

Konstruktor kopije

- Konstruktor klase *X* čiji je prvi parametar tipa *X&*, *const X&*, *volatile X&* ili *const volatile X&*, a koji ili nema druge parametre, ili svi ostali parametri imaju podrazumevane vrednosti, naziva se *konstruktor kopije* (*copy constructor*)
- Ovaj konstruktor poziva se kada se objekat klase *X* inicijalizuje objektom istog tipa (osim ako postupak odabira konstruktora ne odabere neki drugi konstruktor koji više odgovara pozivu), kao što su sledeće situacije:
  - inicijalizacija objekta: *X x1 = x2* ili *X x1(x2)*, gde je *x2* objekat klase *X* ili iz nje izvedene klase
  - prenos argumenta pri pozivu funkcije *f(x)* koja ima taj parametar tipa *X*, gde je *x* objekat klase *X* ili iz nje izvedene klase
  - povratak vrednosti iz funkcije *f* koja ima povratni tip *X* naredbom *return x*, gde je *x* objekat klase *X* ili iz nje izvedene klase
- Ako se objekat izvedene klase inicijalizuje objektom osnovne klase, i ako se odabere ovaj konstruktor, kao i na svim drugim mestima, referenca na osnovnu kalsu inicijalizuje se referencom na podobjekat osnovne klase unutar objekta izvedene klase:

```
struct Base {
  Base (const Base&);
  …
};

struct Derived : Base {
  Derived ();
  …
};

Derived d;
Base b(d);
```