Hijerarhijska dekompozicija

- Objekat izvedene klase u sebi sadrži podobjekat svake osnovne klase; i tako rekurzivno, ako ta osnovna klasa ima svoje osnovne klase, taj podobojekat imaće u sebi po jedan podobjekat svake od tih daljih osnovnih klasa itd. Kako je operativna memorija linearna, da bi se objekat klase izvedene iz više osnovnih klasa smestio u nju, ovi podobjekti se moraju poređati, i to po redosledu navođenja osnovnih klasa u definiciji izvedene klase:

```
class D : public B1, public B2 {…};
```

- Kada se pravi objekat izvedene klase (*D*), poziva se konstruktor te klase. Ali svaki konstruktor izvedene klase uvek poziva (pre izvršavanja svog tela) konstruktor osnovne klase; taj poziv obezbeđuje prevodilac u generisanom kodu za konstruktor izvedene klase. Analogno, konstruktor osnovne klase u sebi ima poziv konstruktora svoje osnovne klase itd.

- U slučaju višestrukog izvođenja, konstruktori osnovnih klasa pozivaju se po redosledu navođenja tih klasa u definiciji izvedene klase, bez obzira na to kako su ti pozivi navedeni:

```
D::D (…) : B1(…), B2(…) {
  …
}
```

Redosled posiva je uvek *B1*() pa *B2*(), čak i ako su ovde navedeni drugačije

Hijerarhijska dekompozicija

- Treba primetiti da konverzija pokazivača na izvedenu klasu u pokazivač na osnovnu klasu daje kao rezultat pokazivač iste vrednosti u slučaju jednostrukog izvođenja klasa, ali može da daje promenjenu vrednost u slučaju višestrukog izvođenja; analogno važi i za konverziju nadole:

```
class D1 : public B2 {...};
class D2 : public B1, public B2 {...};
```

- Kao i uvek, eksplicitna statička konverzija nadole ne proverava činjenicu da li se iza pokazivača zaista krije objekat tražene klase, pa stoga nije bezbedna iz ugla prevodioca, osim ako se to ne obezbedi nekim drugim načinom, odnosno logikom programa

- Dinamička konverzija je bezbednija, jer dinamički proverava ovo, i vraća *null* vrednost ako objekat nije tražene klase. Ona se može koristiti za konverzije nadole, nagore, ili bočno:

```
D2* d = new D2;
B1* b1 = d;
B2* b2 = dynamic_cast<B2*>(b1);
```