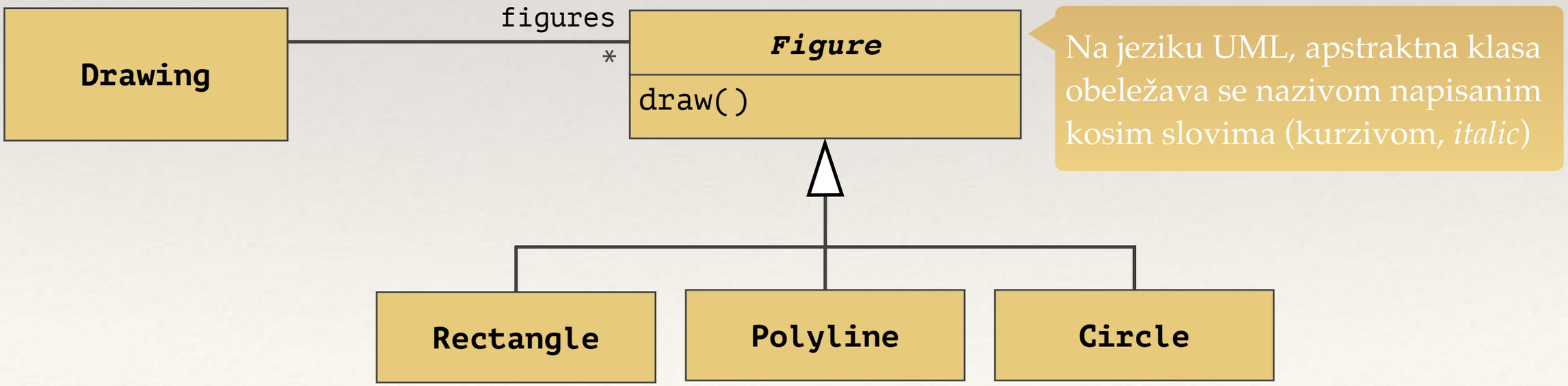
Hijerarhijska dekompozicija

- Klasa *Figure* neće imati svoje direktne instance, već samo indirektne instance (koje su direktne instance izvedenih klasa)
- Ovakva klasa se naziva *apstraktnom klasom* (*abstract class*) ili apstraktnom generalizacijom
- Neki jezici omogućavaju da se apstraktna klasa posebno označi
- Na jeziku C++ nema posebne oznake za apstraktnu klasu, ali se pravljenje objekata te klase (direktnih instanci) može sprečiti deklarisanjem *svih* postojećih konstruktora zaštićenim (*protected*). Zašto baš zaštičenim?
  - ne mogu se kreirati nezavisni objekti te klase, jer konstruktor nije javan (*public*), pa bi prevodilac generisao grešku pri svakom pokušaju kreiranja tog objekta van te klase
  - kada se pravi objekat izvedene klase, poziva se konstruktor te izvedene klase; ali svaki konstruktor izvedene klase obavezno poziva neki konstruktor osnovne klase; ako bi on bio privatan (*private*), ne bi bio dostupan ni u izvedenim klasama, pa se data klasa ne bi mogla nasleđivati



Hijerarhijska dekompozicija

- Kako treba da izgleda implementacija operacije (metoda) *draw* klase *Figure*? Može li se nacrtati figura koja "nije ništa posebno" (a takva i ne postoji, jer je ova klasa apstraktna)?
- U nekim slučajevima, polimorfna operacija apstraktne klase se može implementirati u toj klasi:
  - kada ova operacija ima neko logično podrazumevano ponašanje, makar i prazno
  - ona je pomoćna (*helper*) metoda koja se može, ali ne mora redefinisati; recimo, opciono proširenje podrazumevano praznog koraka neke složenije metode osnovne klase
- Ako to nije slučaj, operacija je *apstraktna* (*abstract operation*): predstavlja samo *specifikaciju usluge* koja se može tražiti od objekata, ali ne daje *implementaciju* te usluge (metodu); tu implementaciju daće izvedene klase