Deklaracija i definicija funkcije

- Funkcije su jedini oblik potprograma na jezicima C i C++: procedure su specijalne vrste funkcija koje imaju povratni tip *void* (ne vraćaju rezultat)
- Deklaracija funkcije koja nije definicija može da se pojavi u bilo kom opsegu važenja (na bilo kom mestu)
- Deklaracija funkcije u opsegu važenja klase deklariše funkciju članicu, osim ako je deklarisana specifikatorom *friend*
- Ako deklaracija funkcije uključuje i njeno telo, onda je definicija
- Definicija funkcije može da se pojavi samo u opsegu važenja prostora imena i unutar definicije klase. Definicija funkcije ne može da se pojavi unutar druge funkcije - ne postoji statičko ugnežđivanje funkcija kao npr. na jeziku Pascal. Naravno, dozvoljeno je dinamičko ugnežđivanje poziva funkcija, pa i rekurzija
- Definicija funkcije može, umesto tela, da sadrži = *delete*. Ovakva funkcija naziva se *obrisanom* (*deleted*). Svaka upotreba ovakve funkcije, npr. njen poziv, je neispravna i uzrokuje grešku u prevođenju; ovako se mogu sprečiti npr. neke implicitne konverzije ili inicijalizacije implicitno generisanim konstruktorima; na primer:

```
class X {
public:
  X (const X&) = delete;
  …
};
```

Konstruktor kopije je obrisan, pa se objekti ove klase ne mogu inicijalizovati kopiranjem

Deklaracija i definicija funkcije

- Funkcija može, a ne mora imati parametre. Funkcija koja nema parametre deklariše se kao *f()* ili *f(void)*, svejedno (radi se o istoj stvari)
- U tip funkcije ulazi povratni tip, kao i tipovi svih parametara, pri čemu se ne pravi razlika između sledećih tipova parametara:
  - niza elemenata tipa *T* sa dimenzijom ili bez nje i pokazivača na *T*
  - funkcije nekog tipa i pokazivača na funkciju istog tipa
  - parametra sa cv-kvalifikacijom i bez nje

Na primer, sledeće deklaracije istih imena su deklaracije istih funkcija (a deklaracije različitih imena su deklaracije različitih funkcija):

```
void f(int);
void f(const int);

void g(const int*);
void g(const int* const);

void h(int[]);
void h(int[5]);
void h(int*);
```