Deklaracije

- Deklaracije koje uvode varijable ili funkcije imaju sintaksu kao na jeziku C; one mogu biti vrlo komplikovane i nečitke za složene tipove, ali neka osnovna orijentaciona pravila za tumačenje tipa ovakvih imena jesu sledeća:
  - Prilikom tumačenja tipa u deklaraciji, kreće se od mesta imena; ako se radi o složenom tipu za *type-id*, kreće se od mesta gde bi se nalazilo ime, da se radi o deklaraciji imena
  - Iterativno se izgrađuje rečenica koja opisuje dati složeni tip; počinje se od rečenice "… je tipa -"
  - Od tekuće pozicije uvek se ide najpre nadesno, dokle god se ne naiđe na zatvorenu zagradu ili na kraj; onda se ide nalevo, sve dok se ne dođe do otvorene zagrade ili do kraja, čime se zaključuje jedan nivo i ponovo kreće nadesno
  - Ako se, pri kretanju nadesno, naiđe na znake *[]*, na rečenicu se dodaje iskaz "niz od … elemenata tipa -"
  - Ako se, pri kretanju nadesno, naiđe na listu argumenata unutar zagrada *()*, na rečenicu se dodaje iskaz "funkcija koja prima argumente tipa (i za svaki argument u listi pročita se tip) i vraća rezultat tipa -"
  - Ako se, pri kretanju nalevo, naiđe na znak *, na rečenicu se dodaje iskaz "pokazivač na tip -"
  - Ako se, pri kretanju nalevo, naiđe na znak *&*, na rečenicu se dodaje iskaz "referenca na tip -"
  - Ako se, pri kretanju nalevo, naiđe na ime tipa, na rečenicu se dodaje ime tog tipa
  - Ime tipa na početku deklaracije odnosi se na sva deklarisana imena, dok se ostali elementi tzv. deklaratora (npr. znaci za reference i pokazivače) odnose samo na svakog pojedinačno
- Primeri:

Ovo nije dobar stil programiranja!

```
int *pi, &ri, i, *a[10], (*pa)[], (*pap)[], f(int*), (*pf)(int), (*apf[])(int), &(*pf)(int&(*)(int&));
```

- Naravno, preterano složene i nečitke iskaze treba svakako izbegavati i učiniti čitljivijim uvođenjem *typedef* sinonima

Deklaracije

- Veoma često je potrebno deklarisati varijablu koja se inicijalizuje vrednošću nekog izraza, uključujući i poziv funkcije. Relativno često, a posebno u slučaju korišćenja šablonskih klasa iz standardne biblioteke i njihovih operacija, tip takve povratne vrednosti je veoma složen i uključuje parametrizovane šablonske klase, cv-kvalifikatore i slično; slična situacija je i u definisanju samih metoda složenih šablonskih klasa, gde tip neke varijable zavisi od parametara šablona
- U takvim situacijama programeru može biti teško i nepraktično da obavezno precizira taj tip u deklaraciji varijable, ili bi time program postao teže čitljiv. Sa druge strane, prevodilac tip te povratne vrednosti izraza ili funkcije sasvim pouzdano zna za vreme prevođenja
- Za ovakve potrebe, moguće je deklarisati varijablu bez eksplicitnog navođenja tipa, uz ključnu reč *auto*. Time varijabla ima i dalje statički definisan tip, ali taj tip određuje sam prevodilac na osnovu tipa inicijalizatora te varijable, i dalje ga koristi kao da je on eksplicitno naveden u deklaraciji
- Ovo se može koristiti u bilo kojoj deklaraciji koja definiše varijablu sa inicijalizatorom, što može da olakša pisanje koda i učini ga kompaktnijim. Na primer:

```
template<typename T>
T sum (const list<T>& array) {
  T sum = 0;
  for (auto it=array.cbegin(); it!=array.cend(); it++)
    sum += *it;
  return sum;
}
```