Reference

- U principu, reference se upotrebljavaju onda kada se želi notacija prenosa argumenta po vrednosti (umesto da se prenosi pokazivač kao "eksplicitan" posrednik), ali je potrebno da:
  - se izbegne kopiranje stvarnog argumenta u formalni
  - se podrži supstitucija, tj. da stvarni argument može biti specijalizacija tipa formalnog parametra
  - argument bude polimorfan
- Druga situacija jeste ta kada se želi da funkcija vraća nešto što se može dalje menjati, tipično kod preklapanja operatora, kako bi se ponašali slično operatorima za ugrađene tipove; na primer:

```
vector<complex> v(100);
v[i]++;
```

Operator *[]* definisan za klasu *vector<complex>* vraća referencu tipa *complex&*

Klasni tipovi

- Na jeziku C++, klasni tipovi su *strukture* (*struct*) i *klase* (*class*). Strukture i klase su skoro potpuno izjednačene na jeziku C++, jer se tretiraju na potpuno identičan način (osim dole navedenih izuzetaka):
  - i strukture i klase mogu imati podatke članove i funkcije članice, uključujući i konstruktore, destruktore i operatorske funkcije
  - i strukture i klase mogu imati javne, zaštićene i privatne članove
  - i jedne i druge mogu se izvoditi, mogu imati polimorfne operacije itd.
- Jedine razlike između strukture i klase su sledeće:
  - ako se u definiciji strukture ne navede specifikator prava pristupa, podrazumeva se *public*; ako se u definiciji klase ne navede specifikator prava pristupa, podrazumeva se *private*
  - ako se u definiciji klase ili strukture, prilikom izvođenja iz druge klase ili strukture (dozvoljeno je sve), za osnovnu strukturu ne navede specifikator prava pristupa, podrazumeva se *public*, dok se za osnovnu klasu podrazumeva *private*
- Ovo su, ipak, krajnje sporedne razlike i ne treba se na njih oslanjati, jer to može da učini program slabije razumljivim: svakako je bolje uvek navoditi specifikatore prava pristupa eksplicitno, radi razumljivosti
- Zbog svega ovoga, strukture (*struct*) se koriste samo u izuzetnim situacijama, kada treba predstaviti sasvim jednostavne apstraktne tipove podataka, po pravilu onda kada se oni koriste samo za implementaciju nekih drugih struktura ili apstrakcija; strukture tada po pravilu imaju samo podatke članove i eventualno konstruktore, retko kada i neke jednostavne operacije ili destruktore
- U svim drugim slučajevima, posebno kada je potrebno da imaju iole složenije operacije ili predstavljaju apstrakciju, treba koristiti klase