Kopiranje objekata

- Pre verzije jezika C++17, svaki rezultat izraza klasnog tipa, uključujući poziv funkcije koja ima povratni tip klase i eksplicitan poziv konstruktora klase, kao rezultat proizvodi privremeni, bezimeni objekat koji se pravi na mestu poziva funkcije, tokom izvršavanja izraza koji sadrži taj poziv i inicijalizuje u trenutku povratka iz te funkcije izrazom iza naredbe *return*
- Na primer, osnovna semantika (pre verzije C++17) sledeće inicijalizacije

```
string s = s1 + s2;
```

je sledeća: rezultat izraza *s1+s2* je privremeni objekat klase *string* koji je inicijalizovan prilikom povratka iz funkcije odgovarajućim konstruktorom; nakon povratka, objekat *s* klase *string* inicijalizuje se pozivom konstruktora kopije sa tim privremenim objektom kao argumentom poziva konstruktora; naravno, ti konstruktori moraju da budu dostupni na odgovarajućim mestima

- Potpuno ista situacija je i u sledećem slučaju:

```
string s = string("Hello");
```

- U određenim situacijama, prevodiocu je na raspolaganju optimizacija *izostavljanja kopiranja* (*copy elision*) koju može, ali ne mora da izvrši (pre verzije C++17), čak i kada konstruktori čiji se pozivi izostavljaju imaju bočne efekte

Kopiranje objekata

- Jedan slučaj u kom se može (pre verzije C++17), odnosno mora (od verzije C++17) vršiti izostavljanje kopiranja jeste inicijalizacija objekta privremenim objektom koji je rezultat izraza, ukjučujući i poziv konstruktora u izrazu: kada se objekat klase *string* inicijalizuje izrazom koji vraća rezultat tipa *string*, za rezultat tog izraza ne mora da se pravi privremeni objekat, već taj rezultat može neposredno da se izgradi u objektu koji se inicijalizuje, tako što se taj objekat inicijalizuje isto kao što bi se inicijalizovao taj privremeni objekat:

```
string sa = s1 + s2;
string sb = string("Hello");
```

- Tada će naredba *return* u pozivu operatorske funkcije *operator+(s1,s2)* konstruisati rezultat u samom memorijskom prostoru objekta *sa* koji se rezultatom ovog poziva operatorske funkcije inicijalizuje, pa konstruktor kopije neće biti pozivan; slično važi i za konstruktor *string("Hello")*
- Ovo važi za sve inicijalizacije objekata, uključujući i parametre funkcija pri pozivu funkcije, kada se oni inicijalizuju stvarnim argumentima koji su rezultati izraza
- Ovo se implementira tako što se funkciji u pozivu zapravo dostavlja, kao jedan od skrivenih parametara, adresa memorijskog prostora za objekat u koji treba da konstruiše i smesti rezultat; ako je to privremeni objekat, na mestu poziva odvaja se taj prostor i pozvanoj funkciji prosleđuje njegova adresa; ako je to objekat koji se inicijalizuje uz izostavljanje kopiranja, onda se dostavlja adresa tog objekta
- Pritom, odgovarajući konstruktor kojim bi se inicijalizovao privremeni objekat, kao i konstruktor kopije moraju biti dostupni kao da se pozivaju (iako se zapravo ne izvršavaju), pa prevodilac sprovodi formalna pravila isto kao da se oni pozivaju, odnosno kao da se ova optimizacija ne vrši (tzv. *as if* pravilo)