Prevođenje

- Kada naiđe na novu deklaraciju, prevodilac dodaje deklarisani identifikator u strukturu podataka koju izgrađuje tokom prevođenja i koja se tradicionalno naziva *tabela simbola* (*symbol table*); u ovoj strukturi prevodilac čuva informacije o svakom deklarisanom identifikatoru: o tome kojoj jezičkoj kategoriji pripada (tip, objekat, funkcija itd.), kog je tipa, kao i sva ostala svojstva deklarisanog entiteta definisana pravilima jezika
- Kada naiđe na neki upotrebljen identifikator, prevodilac:
  - proverava da li je taj identifikator deklarisan i da li je dostupan, po pravilima jezika; ako nije, prijavljuje grešku;
  - proverava da li je identifikator upotrebljen u skladu sa pravilima jezika i ako nije, prijavljuje grešku; na primer, ne može se vršiti operacija *f*++ ako je *f* funkcija, ili operacija *a()* ako je *a* objekat tipa *int* i slično;
  - ako je to definisano semantikom jezika, zna kako da generiše kod za upotrebu tog identifikatora u odgovarajućem kontekstu

Prevođenje

- Na primer, deklaracija globalnog statičkog objekta jeste i *definicija*, koja ima efekat pravljenja objekta:

```
int n = -16;
```

- Ova definicija ima:
  - deklarativni deo (pre znaka =), koji deklariše ime (prevodilac uvodi ime u tabelu simbola)
  - inicijalizator (izraz iza znaka =), kojim se inicijalizuje objekat
- Po pravilima jezika, ovakav objekat ima *statičko trajanje skladištenja* (*static storage duration*) i *statički životni vek*; prema semantičkim pravilima jezika C++, za ovakve objekte važi to da postoji jedna instanca objekta za svaku definiciju (za razliku od npr. definicija automatskih objekata, za koje se kreira nova instanca svaki put kada izvršavanje dođe do takve definicije)
- Zbog toga, za ovakve objekte prevodilac može (i to po pravilu radi) da alocira prostor *statički*, u vreme prevođenja; taj prostor odvaja se u prevedenom objektnom fajlu (obično u posebnom segmentu za podatke, odvojenom od segmenta za instrukcije): za svaki takav objekat odvoji se prostor u prevedenom zapisu za smeštanje tog objekta
- U navedenom primeru, inicijalizator objekta je *konstantan izraz* (*constant expression*) — izraz čiji se rezultat može izračunati u vreme prevođenja
- Za ovakve statičke objekte fundamentalnog tipa, inicijalizovane konstantnim izrazom, prevodilac po pravilu inicijalizuje statički alocirani prostor vrednošću izraza još u vreme prevođenja (ovde je konstantni izraz trivijalan - celobrojni literal *-16*, čiji se binarni zapis upisuje u alocirani prostor)
- Za funkcije, definicija je ona deklaracija koja daje i telo funkcije; za definiciju funkcije, prevodilac generiše binarni mašinski kod za instrukcije koje predstavljaju prevod naredbi iz tela funkcije

A.cpp

```
int n = -16;

void f () {
  n++;
}
```

A.obj

```
n: ff ff ff f0
f: ld r1,n
   inc r1
   st r1,n
   ret
```