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
```

Ovo je izmišljen, pojednostavljen format *obj* fajla. Za mašinske instrukcije generisani kod je binaran.

Prevođenje

- Međutim, osim toga, prevodilac u prevedenom fajlu ostavlja i informacije o svim imenima (simbolima) koji su definisani u datom fajlu, a mogu se koristiti u drugim fajlovima; ovakva imena nazivaju se imena sa *spoljašnjim vezivanjem* (*external linking*)
- U posebnom delu objektnog fajla, tipično u zaglavlju, prevodilac pravi tabelu takvih, *izvezenih* simbola, ostavljajući samo informaciju o:
  - imenu (simbolu, prost niz znakova), bez ikakvih informacija o tome kakav je entitet predstavljalo to ime u programu (šta je, kog je tipa itd.)
  - adresi, relativnoj u odnosu na početak binarnog prevoda (ili segmenta) u koji se to ime preslikava
- Na primer, po pravilima jezika C++, globalni objekat ili funkcija ima spoljašnje vezivanje, osim ako je eksplicitno deklarisana kao *static* (tada ima interno vezivanje)
- Imena koja imaju *interno vezivanje* (*internal linking*) ne mogu se koristiti u drugim fajlovima; prevodilac za ovakva imena ne ostavlja ovakve informacije u generisanoj tabeli simbola u objektnom fajlu

A.cpp

```
int n = -16;

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

A.obj

```
symbols
  ↑n: data:0
  ↑f: code:0
end symbols

seg data
n: ff ff ff f0
end seg

seg code
f: ld r1,n
   inc r1
   st r1,n
   ret
end seg
```