Inline funkcije

- Međutim, takve funkcije koje samo delegiraju poziv drugim funkcijama prave nepotreban režijski trošak u vreme izvršavanja: smeštanje povratne adrese na stek, izvršavanje instrukcije skoka u potprogram, indirektan povratak iz potprograma preko povratne adrese skinute sa steka, pa u mnogim slučajevima i kopiranje argumenata na stek i njihovo skidanje sa steka su potpuno nepotreban trošak za funkciju koja će samo ponovo izvršiti poziv druge funkcije ili prosto pročitati ili upisati podatak
- Zbog ovoga je još odavno osmišljena optimizaciona tehnika u prevodiocima *neposrednog ugrađivanja koda* pozvanog potprograma na mesto poziva (*inlining*): umesto koda za skok u potprogram, sa prenosom argumenata i čuvanjem povratne adrese na steku, kod pozvanog potprograma se neposredno ugrađuje u kod pozivaoca; ovo uzrokuje kraće vreme izvršavanja, ali i nešto veći "memorijski otisak" (*memory footprint*), odnosno veličinu programa
- U svakom slučaju, navedeni primeri trivijalnih metoda koje delegiraju pozive drugim ili samo čitaju/upisuju podatak sprovođenjem ove optimizacije ne prave nikakav trošak, a imaju značaj u dizajnu programa (bolja dekompozicija, enkapsulacija)
- Na neki način, ovo se može smatrati korakom unazad u evolutivnom razvoju programiranja: koncept potprograma je u najstarije programske jezike uveden da bi se smanjila redundantnost i unapredila dekompozicija, tj. da se ne bi ponavljao isti kod na svakom mestu korišćenja, već se on izdvaja u potprogram koji se poziva, potencijalno parametrizovano, sa različitih mesta. Neposredno ugrađivanje u kod radi obrnutu stvar i pravi redundansu, ali je razlika velika:
    - ovaj postupak radi prevodilac potpuno atuomatski i skriveno od programera
    - redundansa postoji samo u mašinskom kodu, dok je izvorni kod dekomponovan i bez redundanse

Inline funkcije

- U drugim jezicima, neposredno ugrađivanje u kod je isključivo optimizaciona tehnika čije je sprovođenje diskreciono pravo prevodioca (može da ga sprovodi, ali ne mora, po svom nahođenju) i potpuno je nevidljiva za programera
- U jezik C++ je ova tehnika uvedena kako koncept jezika, tako što programer može deklarisati funkciju kao *inline*, kao preporuku prevodiocu da izvrši ovu optimizaciju, odnosno telo te funkcije ugradi u kod na mestu poziva
- Međutim, važno je naglasiti da se semantika programa ni na koji način ne menja, bez obzira na to da li je funkcija *inline* ili ne: sva semantička pravila važe na potpuno isti način; na primer, formalni parametri se inicijalizuju stvarnim argumentima i na kraju funkcije uništavaju na potpuno isti način
- Štaviše, prevodilac može, ali uopšte ne mora da ispoštuje zahtev za neposredno ugrađivanje u kod. Neki prevodioci će, na primer, odbiti da to urade ako funkcija ima petlju ili lokalni statički objekat, a svakako ne mogu to da urade ako je funkcija rekurzivna
- Bez obzira na to da li prevodilac uradi ovu optimizaciju ili ne, semantika programa se svakako ne menja
- U svakom slučaju, kao *inline* treba deklarisati samo funkcije koje su jednostavne i kratke, poput onih navedenih u datim primerima