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

Inline funkcije

- Funkcija se može deklarisati kao *inline* navođenjem ovog specifikatora u deklaraciji. Funkcije članice klase *X*, kao i prijateljske funkcije klasi *X* koje su definisane u definiciji date klase *X* (u definiciji klase im je navedeno i telo) su implicitno *inline*, čak i ako se to ne naglasi:

```
inline void strcpy(char* to, const char* from) { while (*to++ = *from++); }

class Person {
public:
  string getName () const { return name; }
  Person& setName (const string& newName) { name = newName; return *this; }
  …
};
```

ili:

```
class Person {
public:
  inline string getName () const;
  inline Person& setName (const string& newName);
  …
};

string Person::getName () const { return name; }
Person& Person::setName (const string& newName) { name = newName; return *this; }
```