Izrazi

- Postupak određivanja operanada operatora zasniva se na:
  - prioritetu operatora
  - načinu grupisanja (asocijativnosti) operatora: ako se rezultat operatora može koristiti kao operand tog operatora, da li se operacije u nizu izračunavaju sleva nadesno ili zdesna nalevo
  - podrazumevani način se može promeniti zagradama koje uokviruju podizraze
- Na primer:

```
a+b*c
```
Izračunava se kao *a+(b*c)*, jer operator * ima viši prioritet nego operator +

```
(a+b)*c
```
Promena načina izračunavanja zagradama (podizrazima)

```
cout<<p->inc()
```
Izračunava se kao *cout<<((p->inc))*, jer operatori -> i *()* imaju viši prioritet nego operator <<

```
**p = 3
```
Izračunava se kao*(*p)*, jer operator * grupiše zdesna nalevo

```
a+b+c
```
Izračunava se kao *(a+b)+c*, jer operator + grupiše sleva nadesno

Operatori

- Jezici C i C++ su veoma bogati operatorima. Zapravo je najveći deo obrade u tipičnom C/C++ programu definisan izrazima
- *Bočni efekat* (*side effect*) se naziva pojava da funkcija/operacija, za koju se izračunavanje rezultata (povratne vrednosti) smatra primarnim efektom, izvrši neku promenu u svom okruženju, npr. promenu svojih argumenata/operanada. U klasičnoj teoriji programiranja, bočni efekti se smatraju lošom praksom, jer smanjuju razumljivost programa: funkcija/operacija ima efekte koje čitalac ne očekuje i ne vidi direktno, jer je fokusiran na njen rezultat
- Potpuno suprotno ovom uverenju, mnogi operatori na jezicima C i C++ imaju bočne efekte, tj. menaju neki od svojih operanada. Zapravo je za većinu njih taj bočni efekat upravo njihova primarna uloga!
- Ovo je posledica izvorne orijentacije jezika C prema konciznim i efikasnim izrazima u kojima programer sugeriše optimizacije prevodiocu: raniji prevodioci nisu bili toliko napredni i prevođenje u optimizovani kod je bilo jednostavnije ukoliko sam izraz definiše način korišćenja neke vrednosti koja je pročitana ili proizvedena kao rezultat u jednoj operaciji kao operand neke naredne