Preporučeni načini preklapanja operatora

- Operator *()* se preklapa za klase čiji se objekti mogu posmatrati kao tzv. *funkcijski objekti* (*function object*), tj. objekti koji se mogu koristiti kao funkcije, jer se na njih može primeniti operator poziva funkcije (kao za obične funkcije):

```
x(expression1, expression2)
```

tumači se kao:

```
x.operator()(expression1, expression2)
```

- Za razliku od običnih funkcija, funkcijski objekti, kao i svaki drugi objekti, mogu imati (i tipično imaju) svoje stanje, koje "nose" sa sobom (svaki objekat svoje nezavisno stanje) i mogu da ga menjaju i akumuliraju
- Ovakvi objekti se tipično prenose, slično kao i pokazivači na funkcije, kao argumenti nekih operacija koje sprovode određene postupke, odnosno algoritme, za koji su im potrebne implementacije elementarnih operacija-koraka tog algoritma; zato takav algoritam poziva nazad dostavljenu funkciju, odnosno funkcijski objekat (tzv. *callback* mehanizam), ili ga primenjuje na neke elemente koje obilazi tokom algoritma (npr. obilasci raznih struktura)
- Na primer, funkcija *for_each* iz standardne biblioteke iterira kroz kolekciju, dok kao treći parametar očekuje funkcijski objekat koga primenjuje na svaki posećeni element, dostavljajući mu taj posećeni element kao argument operatora *()*:

```
struct Sum {
    int sum;
    Sum () : sum(0) {}
    void operator() (int n) { sum += n; }
};

std::vector<int> v{...};

Sum s = std::for_each(v.begin(), v.end(), Sum());
```

Operatori *new* i *delete*

- Kao što je već objašnjeno, izraz oblika

```
new (placement_params) T (init)
```

uvek radi sledeće tri stvari:

1. Poziva neku od ugrađenih globalnih alokatorskih funkcija oblika

```
void* ::operator new (size_t size, placement_params)
void* ::operator new [] (size_t size, placement_params)
```

2. Inicijalizuje objekat odgovarajućom inicijalizacijom
3. Vraća pokazivač na napravljeni dinamički objekat

- Ovaj postupak je uvek isti i ne može se promeniti, u smislu skupa i redosleda koraka, ali se poziv alokatorske funkcije u prvom koraku može preusmeriti na drugu funkciju, pa i onu definisanu posebno za neku klasu preklapanjem alokatorske funkcije *new* za tu klasu
- U prvom koraku, ugrađene alokatorske funkcije *::new* podrazumevano rade sa slobodnim delom memorije u kom pronalaze slobodan prostor tražene veličine *size* i alociraju ga
- Ove funkcije mogu da se zamene drugim, korisnički definisanim, sa istim potpisom; dovoljno je definisati neku od njih u nekom fajlu