# Obrada izuzetaka

- Paradigma programiranja i projektovanja softvera pod nazivom *programiranje po ugovoru* (*programming by contract/design by contract*, Bertrand Meyer, 1986, programski jezik *Eiffel*) pretpostavlja preciznu, formalnu specifikaciju interfejsa softverskih komponenata (npr. potprograma ili klasa) za koje se definišu logički uslovi (kao logički izrazi) sledećih vrsta:
  - *preduslovi* (*precondition*): logički uslovi koji moraju da budu zadovoljeni prilikom poziva potprograma da bi taj potprogram mogao da obavi svoj zadatak; ove uslove mora da zadovolji pozivalac pri pozivu potprograma; na primer, uslovi u pogledu stanja globalnih promenljivih ili argumenata poziva potprograma (npr. indeks mora biti unutar granica niza)
  - *postuslovi* (*postcondition*): logički uslovi koji moraju da budu zadovoljeni na kraju izvršavanja potprograma; ove uslove mora da zadovolji pozvani potprogram; na primer, uslovi u pogledu stanja globalnih podataka ili validnosti povratne vrednosti funkcije
  - *invarijante* (*invariant*): logički uslovi vezani za neku komponentu (npr. instancu apstraktnog tipa podataka, objekat klase, modul, strukturu podataka) koji moraju da budu zadovoljeni uvek, osim u tranzijentnim periodima kada se ta komponenta prevodi iz jednog validnog stanja (kada invarijanta važi) u drugo validno stanje (kada invarijanta ponovo važi); na primer, metode klase koje prevode objekat iz jednog regularnog u drugo regularno stanje, tokom koje invarijanta privremeno ne mora da važi
- Prema tome, preduslovi su obaveze koje mora da izvrši pozivalac prema pozvanom potprogramu, da bi taj potprogram mogao da izvrši svoju obavezu; postuslovi su obaveze koje potprogram mora da ispuni ukoliko je pozivalac ispunio svoje obaveze; zato oni predstavljaju *ugovor* (*contract*) između dve strane

# Obrada izuzetaka

- Preporučeni stil tretiranja izuzetaka jeste taj da se, prema ovoj paradigmi, izuzetkom tretira nemogućnost zadovoljenja nekog od ovih logičkih uslova, i *samo* to (sve drugo nisu izuzeci):
  - preduslovi (*precondition*): pozvani potprogram, ukoliko nije ispunjen njegov preduslov (npr. neregularan argument) treba da podigne izuzetak; na primer, neispravan parametar
  - postuslovi (*postcondition*): ukoliko potprogram nije u mogućnosti da ispuni svoje postuslove, treba da signalizira izuzetak svom pozivaocu; na primer, funkcija ne može da kreira povratnu vrednost ili ne može da uspostavi njenu invarijantu, ne može da pronađe ono što bi morala da pronađe i vrati, i slično
  - invarijante (*invariant*): ukoliko metoda klase ne može da očuva invarijantu, treba da baci izuzetak; invarijante su zapravo preduslovi i postuslovi koji važe za objekat na ulazu i izlazu svake metode klase
- Jedna tehnika za apstrahovano ispitivanje uslova - tzv. *tvrdnje* (*assertion*):

```
assert(i>=0 && i<this->size);
```

- U standardnoj biblioteci za C++, *assert* je definisan kao makro koji zavisi od drugog definisanog makroa *NDEBUG* koji nije definisan u biblioteci, već se može definisati (uključiti ili isključiti) u okruženju prevodioca:
  - ako je *NDEBUG* definisan (uključen), makro *assert* nema nikavog efekta (zamenjuje se sa `((void)0)`)
  - u suprotnom, zamenjuje se implementacijom koja ispituje vrednost datog skalarnog izraza; ako je ta vrednost nula, na standardni izlaz za greške ispisuje informacije o mestu u programu i završava program
- Ovakvi slični makroi ili funkcije mogu da se koriste za ispitivanje preduslova, postuslova i invarijanti, ali tako da podižu izuzetke ukoliko uslov nije ispunjen