Obrada izuzetaka

- Izuzetak na jeziku C++ može biti objekat bilo kog tipa, ugrađenog tipa ili klase
- Funkcije iz standardne biblioteke bacaju izuzetke koji su bezimeni objekti klasa izvedenih iz osnovne klase *exception*; sve ove izvedene klase imaju sledeća osnovna svojstva:
  - objekti ovih klasa mogu se kreirati pozivom konstruktora sa argumentom koji predstavlja niz znakova koji daje objašnjenje za izuzetak:

```
...throw out_of_range("Argument pos in function vector::at out of range.");
```

  - objekti ovih klasa mogu se kopirati (prilikom inicijalizacije ili operatorom dodele =)
  - polimorfnu operaciju *what* koja vraća niz znakova koja predstavlja objašnjenje (zadat inicijalizacijom)
- Po pravilu, i korisnički definisani izuzeci treba da slede isti obrazac, odnosno da budu objekti klasa (izvedenih iz klasa) iz ove hijerarhije

Obrada izuzetaka

Obrada (hvatanje) izuzetka:

- Kada se izuzetak baci, prekida se izvršavanje prvog dinamički okružujućeg *try* bloka koji je započet, a nije završen, i u njemu traži *catch* blok koji može prihvatiti tip bačenog izuzetka; pravila uparivanja (skoro) su ista kao za argumente funkcija (uz sprovođenje implicitnih konverzija)
- *catch* blokovi priduženi istom *try* bloku pretražuju se redom kako su navedeni (može ih biti više); bira se prvi koji po tipu argumenta odgovara bačenom izuzetku (može obraditi taj izuzetak) i započinje se njegovo izvršavanje
- ukoliko u datom *try* bloku nijedan *catch* blok ne može prihvatiti bačeni izuzetak, traži se sledeći dinamički okrućujući *try* blok (mogu se ugnežđivati), koji može biti i van funkcije koja se izvršava - funkcija tako baca izuzetak (prosleđuje ga svom pozivaocu), i tako redom po steku ugnežđenih poziva funkcija u tekućoj niti
- *catch* blok može baciti isti ili neki novi izuzetak, npr. tako što funkcija koja se unutar njega poziva baca izuzetak
- ako se *try* blok završi bez bačenog izuzetka, ili se *catch* blok koji je aktiviran završi bez bačenog izuzetka, izvršavanje nastavlja iza *try-catch* konstrukta

```
void readMeteo () {
  try {
    …
  }
  catch (ThermometerException& e) {
    …
  }
  catch (ManometerException& e) {
    …
  }
}

void calcMeteo () {
  try {
    …
    …readMeteo()…
    …
  }
  catch (DeviceException& e) {
    …
  }
}
```