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) {
    …
  }
}
```

Ukoliko je bačen izuzetak nekog trećeg tipa, biće prosleđen pozivaocu ove funkcije, tj. bačen iz nje

Ova funkcija može baciti izuzetak koji se može uhvatiti u nekom od *catch* blokova dole, ili dalje proslediti pozivaocu, ukoliko se tu ne uhvati

Obrada izuzetaka

- Zbog ovog redosleda, *catch* blok koji prima objekat izvedene klase (po vrednosti ili preko reference ili pokazivača na izvedenu klasu) mora da bude ispred onog koji hvata objekat osnovne klase (po vrednosti ili preko reference ili pokazivača na osnovnu klasu), jer u suprotnom nikada neće biti aktiviran:

```
try {
  …
}
catch (Derived& e) {
  …
}
catch (Base& e) {
  …
}
```

- *catch(…)* hvata izuzetke bilo kog tipa i, shodno tome, mora uvek biti poslednji u nizu (inače ostali nikada ne bi bili aktivirani); ovakav hvatač može poslužiti kao obezbeđenje za to da nijedan izuzetak ne bude bačen iz date funkcije:

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