Obrada izuzetaka

Bacanje izuzetka:

- Operatorom *throw*:

```
template<typename T>
T& vector<T>::at (int pos) {
  if (pos<0 || pos>=this->size) throw out_of_range;

  return this->array[pos];
}
```

Unutar *catch* bloka, tekući izuzetak koji se obrađuje može biti ponovo bačen operatorom *throw* bez operanda:

```
try {
  …
}
catch (…) {
  device->reset();
  throw;
}
```

- Mnoge funkcije iz standardne biblioteke jezika C++ mogu da bace izuzetke, npr. `vector::at`, `string::substr` itd, čime signaliziraju greške (nemogućnost da urade zahtevano npr. zbog nekorektnosti argumenta)
- Neki operatori ugrađeni u jezik mogu da bace izuzetke, npr. `dynamic_cast` (ako rezultat izraza koji se konvertuje nije zahtevanog odredišnog tipa za reference) ili `new` (ako ne može da alocira prostor u memoriji za objekat koji se kreira), opet signalizirajući grešku

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