Esimerkki: Pino 1(3)

```
/** Pino on alkiokokoelma, joka noudattaa
 *  LIFO-käyttäytymistä.
 */
public interface Pino<T> {
  //-- Havainnointioperaatiot
  /** Palauttaa alkioiden määrän.
   *  @.pre   true
   *  @.post  RESULT ==
   *              (pinossa olevien alkioiden lukumäärä)
   */
  public int annaKoko();

  /** Palauttaa pinon päällimmäisen alkion.
   *  @.pre   !onTyhjä()
   *  @.post  RESULT == (pinon päällimmäinen alkio)
   */
  public T päällimmäinen();
```

Esimerkki: Pino 2(3)

```
  /** Palauttaa pinon alimmaiset alkiot.
   *  @.pre   !onTyhjä()
   *  @.post  RESULT == (pinon alimmaiset alkiot pinona)
   */
  public Pino<T> alimmaiset();

  /** Tarkistaa onko pino tyhjä.
   *  @.pre   true
   *  @.post  RESULT == (annaKoko() == 0)
   */
  public boolean onTyhjä();

  /** Tarkistaa onko pino täysi.
   *  @.pre   true
   *  @.post  RESULT == (pinoon mahtuu vielä alkioita)
   */
  public boolean onTäysi();
```

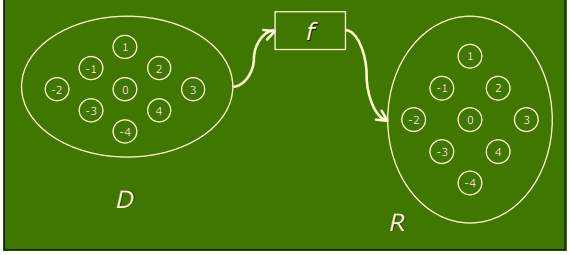
Esimerkki: Pino 3(3)

```
  //-- Muutosoperaatiot
  /** Lisää alkion pinon päälle.
   *  @.pre   !onTäysi()
   *  @.post  OLD(this).equals(this.alimmaiset()) &
   *          (päällimmäinen() == alkio)
   */
  public void lisää(T alkio);
  /** Poistaa pinon päällimmäisen alkion.
   *  @.pre   !onTyhjä()
   *  @.post  this.equals(OLD(this).alimmaiset())
   */
  public void poista();
  /** Tyhjentää pinon.
   *  @.pre   true
   *  @.post  onTyhjä()
   */
  public void tyhjennä(); }
```

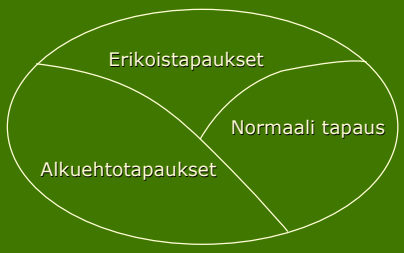
Erikoistilanteiden hallinta

- Rutiini matemaattisena funktiona
  - määrittelyjoukko (*domain*) D
  - arvojoukko (*range*) R
- Rutiini on kuvaus $f : D \rightarrow R$
- Esimerkkejä:
  - itseisarvo: $D = \mathbf{R}$, $R = \mathbf{R}_+$
  - kokonaislukujakolasku: $D = \mathbf{Z} \times \mathbf{Z}$, $R = \mathbf{Z}$

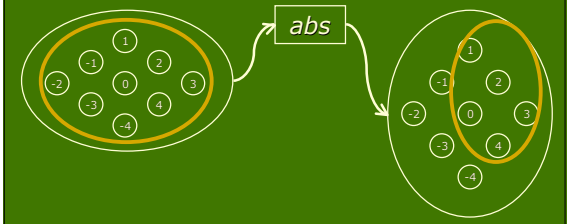
Rutiini matemaattisena funktiona



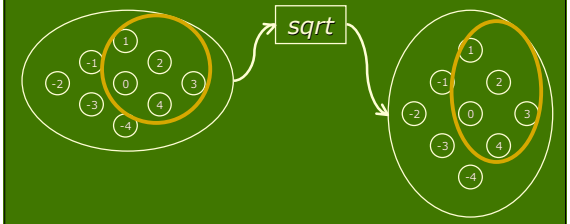
Määrittelyjoukon jako eri osiin



Naiivi esimerkki: Itseisarvo



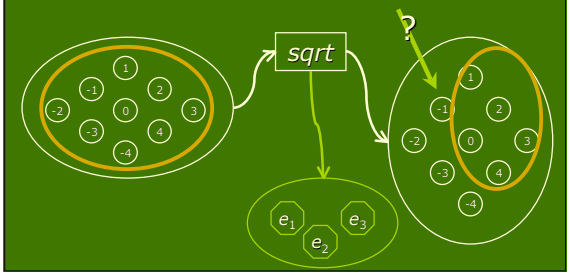
Naiivi esimerkki: Neliöjuuri



Erikoistilanteesta tiedottaminen

- Poikkeuksen nostaminen
  - tarkistettavat poikkeukset
  - tarkistamattomat poikkeukset
- Viittaustyyppisen parametrin kautta
- Toimittajaluokan attribuutin kautta
- Globaalia tietoa sisältävän olion kautta
- Palauttamalla tieto normaalitapauksen tapaan

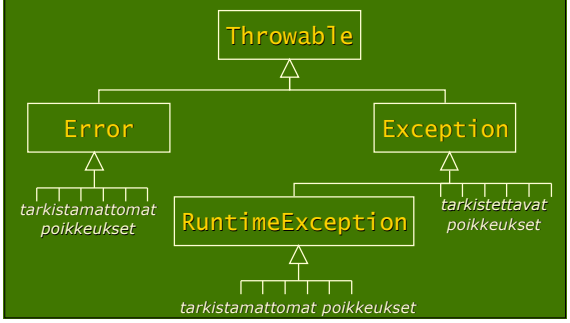
Totaalisen neliöjuurirutiinin erikoistilanteesta tiedottaminen



Nyrkkisääntö erikoistilanteiden käsittelyyn

1. Siirrä erikoistilanne alkuehtoon, mikäli se asiakkaan kannalta selkeää ja ymmärrettävää.
2. Mikäli sääntö 1 ei toimi, erikoistilanteet hoidetaan käsiteltävillä poikkeuksilla, jolloin asiakas pakotetaan ottamaan niihin kantaa.
3. Mikäli säännöt 1 ja 2 eivät toimi, hoidetaan erikoistilanne muulla tavoin.

Javan poikkeushierarkia



Tarkistamattomat poikkeus (*unchecked exception*)

- Ei tarvitse esitellä signatuurissa
- Ei yleensä napata kiinni
- `Error`: vakavat virheet
- `RuntimeException`: ajoaikainen virhe

Tarkistettava poikkeus (*checked exception*)

- Esiteltävä signatuurissa
- Otettava aina ekspliittisesti kantaa
  - nappaa kiinni
  - päästä eteenpäin
- Periytyvät `Exception`-luokasta

Esimerkki poikkeusten käytöstä

```
/** Palauttaa taulukon minimialkion.
 *  @.pre  true
 *  @.post FORALL(a : taulu; RESULT <= a)
 *  @throws NullPointerException
 *                  Nostetaan jos taulu == null.
 *  @throws TyhjäTaulukkoPoikkeus
 *                  Nostetaan jos taulu.length == 0.
 */
public static int minimi(int[] taulu)
     throws NullPointerException, TyhjäTaulukkoPoikkeus {
    try {
        int pienin = taulu[0];
        for (int i = 1; i < taulu.length; i++)
            if (taulu[i] < pienin) pienin = taulu[i];
        return pienin;
    } catch (IndexOutOfBoundsException e) {
        throw new TyhjäTaulukkoPoikkeus("tyhjä taulu");
}   }
```