2.0.3. DMAak. Interfazeak eta klase generikoak

Programaziorako tresnak

- Orokorrean, egitura bat erabiltzeko, eragiketa-multzo txiki batera mugatzeak seguruago (zuzentasunean) egiten du bere erabilera

"Mundu errealaren" moldaketa

- Problema bat ebazteko objektuak/kontzeptuak erabiltzen ditugunean, normalean eragiketa-multzo txiki bat nahiko izaten da. Interesgarria da jakitea zein den multzo hori kasu bakoitzean

Datu mota abstraktua (DMA)

- DMA: datu multzo baten eta datu horiekin egin daitezkeen eragiketa multzo baten zehaztapena (espezifikazioa) da
- Abstraktua: indarra egin daitezkeen zein eragiketetan jartzen da, nola egin daitekeen kontuan izan gabe
- DMA bat independentea da bere inplementazio desberdinekiko

Adibidea: kontagailu DMA

(Gogoratu: programazioaren metodologia)

mota kontagailu

erabiltzen_du nat

eragiketak

hutsa: → kontagailu *(eraikitzailea)*

inkr: kontagailu → kontagailu *(eraikitzailea)*

dekr : kontagailu → kontagailu *(modifikatzailea)*

reset : kontagailu → kontagailu *(modif.)*

balioa: kontagailu → nat *(konsultakoa)*

(Gogoratu: programazioaren metodologia)

ekuazioak

(1) dekr(hutsa) = errorea *(errore ekuazioa)*

(2) dekr (inkr (x)) = x

(3) reset(hutsa) = hutsa

(4) reset(inkr(x)) = hutsa

(5) balioa(hutsa) = zero

(6) balioa(inkr(x)) = suc(balioa(x))

Ez-eraikitzaile bakoitza eraikitzaile bakoitzeko

DMAak diseinu-tresna bezala

- Zein eragiketa behar ditut nire aplikazioko datuak erabiltzeko?
  - Sartutako azkenaren atzipena
  - Lehentasun handienekoaren atzipena
  - Lehenengoa nola inplementatuko diren pentsatu behar da kendu
  - Bi elementu konbinatu
  - ...
- Ondoren, eragiketa horiek
  - Inplementazio bat aldatu ahal izango da, DMAa erabiltzen ari den aplikazioa aldatu gabe
- Inplementaziorako aukerak eraginkortasun-irizpideek gidatuta egongo dira

DMA baten espezifikazioa Javaz

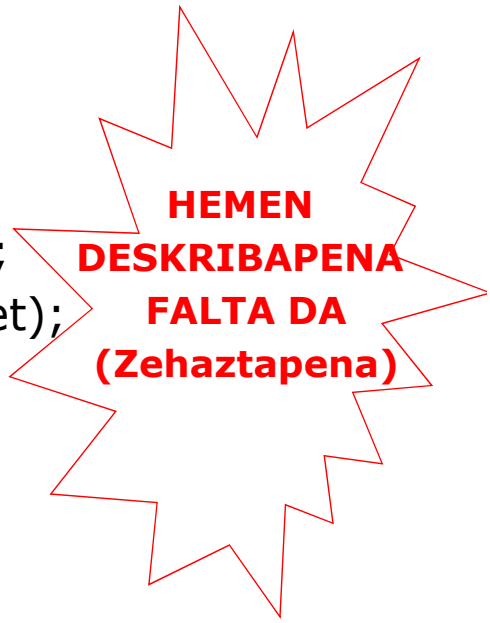
- Softwarearen garapenerako ingurune batean, DMA baten espezifikazioari DMAaren interfazea esaten zaio
- Javaz, *interface* eraikuntza konstante eta metodo abstraktuen bilduma da
  - Metodo abstraktu batean bere signatura (goiburukoa) erazagutuko da, baina ez bere inplementazioa
- *interface* eraikuntzak DMAak zehazteko balio dezake

Adibidea

- Zenbaki osokoen multzo finitua
- Eragiketak:
  - Osoko bat gehitu multzo batean
  - Osoko bat kendu multzo batetik
  - Osoko bat multzo bat dagoen erabaki
  - Multzo bat beste batekin bildu
  - Multzo batek beste baten osoko berdinak dituen ala ez erabaki
  - Esan ea multzo bat hutsa den
  - Multzo bateko osoko-kopurua eman
  - Multzo bateko osokoen balioen irudi testuala eman

Interfazea: osokoen multzoa

```
public interface IntSet {
   public void add (int x);
   public int remove (int x);
   public boolean contains (int x);
   public IntSet union (IntSet set);
   public boolean equals (IntSet set);
   public boolean isEmpty ();
   public int size ();
   public String toString ();
}
```



IntSet DMAaren inplementazioa

```
public class MultzoB implements IntSet {
    private int[] multzoa;
    private int kardinala;

    public MultzoB() {
        multzoa = new int[100];
        kardinala = 0;
    }
    public void add (int x) {
        multzoa[kardinala] = x;
         kardinala++;
    }
    public boolean isEmpty () {
        return (kardinala == 0);
    }
    …ETA HORRELA BESTE METODO GUZTIEKIN
}
```

IntSet INPLEMENTATZEN DUTEN BESTE KLASE BATZUK ERE DEFINITU DITZAKEGU

Eta liburuen multzoak nahi baldin baditugu?

- IntSet interfazeak ez digu laguntzen, nahiz eta behar ditugun eragiketak berdinak diren
- Posiblea litzateke <gauza>-ren multzoak definitzea?
  - Geroago, esango genuke zein den <gauza> objektuen klasea
- Honek klase generikoetara darama
- Honek softwarearen berrerabilgarritasuna handitzen du

Klase generikoak

- Klase bat generikoa izango da mota-aldagai bat edo gehiago definitzen baditu

```
public interface Set<T> {
    public void add (T x);
    public T remove (T x);
    public Set<T> union (Set<T> set);
    public boolean contains (T x);
    public boolean equals (Set<T> set);
    public boolean isEmpty ();
    public int size ();
    public String toString ();
}
```


Generikotasuna

- T mota generiko baten bidez parametrizatutako klaseak definitzeko aukera ematen du
- Klase parametrizatuaren egitura eta metodoek T mota generikoaren abstrakzioa egiten dute
- Mota generikoa ez dago definituta konpilazio-denboran, baina bai egon beharko du exekuzio-denboran

Multzoaren inplementazio generikoa

```
public class ArraySet<T> implements Set<T> {
    private T[] multzoa;
    private int elemKop;

    public ArraySet() {
            multzoa = (T[]) new Object[100];
            elemKop = 0;
    }
    public void add (T x) {
            multzoa [elemKop] = x;
            elemKop ++;
    }
    public boolean isEmpty () {
           return (elemKop == 0);
    }
    … ETA HORRELA BESTE METODO GUZTIEKIN
}
```


Objektuen erazagupena eta sorkuntza klase generikoekin

- Erazagupena:
  - Nodo<Integer> nodoI;
  - Nodo<Data> nodoD;
- Sorkuntza:
  - nodoI = new Nodo<Integer>(15);
  - Fecha f = new Data(10, 11);
  - nodoF = new Nodo<Data>(f);

```
public class Nodo <T> {
        T dato;
        Nodo<T> next;

        public Nodo(T dd)
          {dato = dd;}
}
```