COLLECTIE

ArrayList

Voorbeeld van een collectieobject.

Object dat een willekeurig aantal andere objecten kan bevatten, een lijst.

- Ongesorteerd
- Geordend: de klasse houdt de volgorde bij van de opgeslagen items → index
- Flexibele lengte: altijd mogelijk om objecten toe te voegen of te verwijderen
- Klasse houdt een teller bij van het huidig aantal opgeslagen items
- Wordt vaak gebruikt als wachtrij of om een lijst af te drukken

Is een generieke klasse: definieert niet één type maar potentieel een aantal types

VB: <String>, <Ticketmachine>,… Zo kan men meerdere velden maken.

HashMap

Map concept: sleutel-waardeparen opslaan als records, zo kan met waarden opzoeken aan de hand van de sleutel.

Is een generieke klasse, kan type van sleutelobjecten en type van waardeobjecten bepalen bij de declaratie.

Verschil met ArrayList, hier is elk record een objectenpaar ipv een enkel object.

- Flexibele lengte
- Geen index gebruiken maar sleutel
- Put en get gebruiken!

HashSet

Set concept: een collectie waarin elk afzonderlijk element slechts éénmaal kan worden opgeslagen, geen volgorde bij de elementen van een set.

Generieke klasse: één parameter voor het type van de objecten in de collectie.

Vergelijkbaar met ArrayList maar

- Volgorde van toevoegen niet van belang
- Geen indexnummer nodig om elementen te benaderen
- Hetzelfde element kan maar één keer opgeslagen worden, dus het heeft geen enkel effect als een element voor de tweede keer wordt ingevoerd
- Er hoef dus niet worden nagegaan of element al aanwezig is in HashSet
- Wel: flexibele lengte
- Kan remove

In voorbeeld is er een verzameling doorgegeven van één of meerdere woorden (zin), deze worden dan opgesplitst in afzonderlijke woorden → split methode.

Na split methode worden de substrings teruggegeven in een Array.

Array

Speciaal type collectie waarin een onveranderlijk aantal items wordt opgeslagen

- Geen flexibele lengte: maximum aantal items staat vast
- Items kunnen efficiënter worden benaderd
- Items kunnen waarden van een primitief type zijn

Wrapperclasse en auto(un)boxing is niet nodig

- Items worden benaderd door gebruik te maken van een index, geplaatst tussen [] direct na de naam van de arrayvariabelen → geen .get!!
- .length in plaats van .size()
- Lijst afdrukken → for lus gebruiken: for(int i = 0; …)
    - Collectieobject[i]

Heeft methoden hasMoreEntries() en hasNext() vergelijkbaar met hasNext() en next() van Iterator

VERWERKEN VAN EEN COLLECTIE

Lus: Voert een reeks statements herhaaldelijk uit

bewerkingen die op elk item in een collectie moeten worden uitgevoerd

For-each lus

Bepaalde iteratie: indien je een hele collectie wilt verwerken.

for(itemType item: collectieObject)

{

te herhalen statement

}

De items die in het collectieObject zijn opgeslagen worden een voor een aan de lusvariabele item toegekend. Op elke waarde van item wordt dan de te herhalen statement toegepast.

- Elk item in de collectie wordt één voor één benaderd
- Omvang van collectie kan niet worden gewijzigd
- Geen index
- Kan onderweg niet gestopt worden
- Geschikt voor te itereren over een collectie met veranderlijke omvang
- Niet nuttig in combinatie met een iterator
- Kan niet gebruikt worden op een HashMap

While Lus

Onbepaalde iteratie: we kunnen niet voorspellen op hoeveel plaatsen we moeten zoeken. Booleaanse voorwaarde of lus moet stoppen.

while(lusvoorwaarde)

{

te herhalen statements

}

- Kan werken met een indexvariabele: itereren met een indexvariabele
- Flexibeler dan for-each lus, kan flexibel aantal keren itereren
- Risico op oneindige lus

Iterator

Collecties hebben een iterator() methode, deze retourneert een Iterator-object. Iterator() is een methode van ArrayList en retourneert een iterator object van de klasse Iterator (generieke klasse).

Interactie met het collectieObject gebeurt via het Iterator-object, dit maar de Iterator zo handig, kan gebruikt worden op veel verschillende datatypen zonder te moeten weten hoe ze precies werken.

```
Iterator<itemType> it = collectieObject.iterator();

while(it.hasNext())

{

        haal item op met it.next()

        doe iets met item

}
```

- Als we onderweg misschien willen stoppen
- Als we elementen willen verwijderen tijdens een iteratie
- Werkt voor alle collecties van de Java-klassenbibliotheek

For Lus

Vooraf bepaald aantal keren de statements binnen een lus uitvoeren

Bepaalde iteratie: iets doen met elk item uit de array

- Niet geschikt om te itereren over een collectie met veranderlijke omvang (for-each)
- Binnen de lus is een variabele aanwezig waarvan de waarde bij elke iteratie met een vaste hoeveelheid verandert

```
For(initialisatie; voorwaarde; actie na de body)

{

        te herhalen statements;

}
```

Iterator in een for-lus: voor verwijderen items uit een collectie met veranderlijke omvang

Oefeningen

- Maak een lijst van items opgeslagen in een bestand, volgorde van afdrukken is niet van belang