

Tentamen Inleiding Programmeren (IN1608WI), 1 november 2011, 14.00-16.00
Technische Universiteit Delft, Faculteit EWI, Afdeling 2
Gesloten boek tentamen. Je mag geen gebruik maken van een tekstboek of slides.

Opgave 1 (15 punten)

Voor een financieel programma zijn de klassen `Overschrijving` en `Betalings` geschreven. Klasse `Overschrijving` bevat het bedrag en rekening(nummer) van 1 overschrijving. Klasse `Betaling` bevat een lijst met 0 of meer overschrijvingen. In Bijlage A is een deel van de implementatie van deze klassen gegeven.

Klasse `Betalings` bevat een methode `test()`. Wat is na de uitvoering van deze methode de waarde van onderstaande expressies? Licht je antwoord toe.

Mocht een van de expressies aanleiding geven tot een compileer- of runtime-fout, dan dient je dit aan te geven samen met de reden van de fout.

- a. `betalingen[0] == betalingen[2];`
- b. `betalingen[1] == betalingen[2];`
- c. `betalingen[0].getRekening();`
- d. `betalingen[1].getRekening();`
- e. `betalingen[2].getRekening() == betalingen[1].getRekening();`

Opgave 2(30) punten)

Aan klasse `Betalings` moeten 3 methoden worden toegevoegd. De specificatie van de methoden is hieronder gegeven.

- a. Geef een implementatie van methode `somBedragen()` die de som van de bedragen berekent van de overschrijvingen die in `betalingen` zijn opgeslagen.

`somBedragen() : double`

post: retourneert de som van de bedragen van de overschrijvingen die zijn opgeslagen in `betalingen`.

- b. Geef een implementatie van methode `betalingenAan()` die alle overschrijvingen naar een opgegeven rekening geeft.

`betalingenAan(rek: Rekening) : Betalings`

post: retourneert een `Betalings`-object dat alle overschrijvingen naar rekening `rek` bevat.

- c. Geef een implementatie van methode `bevatRekeningMetNOverschrijvingen()` die vaststelt of `betalingen` een rekening bevat waarnaar `n` overschrijvingen worden gedaan.

`bevatRekeningMetNOverschrijvingen(n : int) : boolean`

post: retourneert `true` dsd er in `betalingen` een rekening voorkomt met `n` overschrijvingen.

Opgave 3 (35 punten)

Gegeven de specificatie van een klasse Verzameling die getallen kan bevatten uit het interval $[og, bg]$, met $0 \leq og \leq bg$, Geef een implementatie van deze klasse.

Verzameling
<pre>-elementen: boolean[] -onderGrens : int -bovenGrens : int</pre>
<pre>+Verzameling(og : int, bg : int) pre: 0 <= og <= bg post: indien pre geldt is er een lege verzameling gecreëerd van getallen uit het interval [og,bg], anders is het interval leeg, dwz []. +bevat(e : int): boolean pre: onderGrens <= e <= bovenGrens post: retourneert true indien pre geldt en e in de verzameling voorkomt, retourneert anders false +voegToe(e : int) pre: onderGrens <= e <= bovenGrens post: indien pre geldt en !bevat(e), is e aan de verzameling toegevoegd +doorsnijding(that: Verzameling) : Verzameling post: retourneert de verzameling van alle getallen die zowel in this als in that voorkomen</pre>

Toelichting van het attribuut elementen: de indices van elementen corresponderen met de getallen uit het interval $[og, bg]$. Als elementen[i] true is, betekent dat i in de verzameling voorkomt, is elementen[i] false, dan komt i niet in de verzameling voor.

Opgave 4 (10 punten)

In de schakeltechniek wordt de werking van netwerken van schakelaars beschreven mbv logische operatoren. Naast de bekende logische operatoren AND en OR wordt daarbij de operator NOR gebruikt.

De waarheidstabel van NOR ziet er als volgt uit:

a	b	a NOR b
true	true	false
true	false	false
false	true	false
false	false	true

Geef een implementatie van methode NOR die aan de waarheidstabel voldoet.

```
public static boolean NOR(boolean a, boolean b)
post: retourneert de waarde die voldoet aan de waarheidstabel van NOR
```

Bijlage A.

```
public class Overschrijving{
    private double bedrag;
    private String rekening;

    public Overschrijving(double b, String r){
        setBedrag(b);
        setRekening(r);
    }

    public void setBedrag(double b){
        bedrag = b;
    }

    public double getBedrag(){
        return bedrag;
    }

    public void setRekening(String r){
        rekening = r;
    }

    public String getRekening(){
        return rekening;
    }
}

public class Betalingen{
    private Overschrijving[] betalingen;
    private int aantal;

    public Betalingen(){
        betalingen = new Overschrijving[200];
        aantal = 0;
    }

    public void voegToe(Overschrijving ov){
        betalingen[aantal] = ov;
        aantal = aantal + 1;
    }

    public void test(){

        Overschrijving ov = new Overschrijving(100.00,"A15");
        voegToe(ov);

        ov = new Overschrijving(200.00,"B33");
        voegToe(ov);

        ov.setBedrag(betalingen[0].getBedrag());
        ov.setRekening(betalingen[0].getRekening());
        voegToe(ov);
    }
}
```

