Vrije Universiteit 19 december 2002

Studenten die het tweede deeltentamen Datastructuren maken krijgen 2:00 uur de tijd en maken de opgaven 2 t/m 3.

Studenten die het gehele tentamen Datastructuren maken krijgen 2:45 uur de tijd en maken de opgaven 1 t/m 3.

Opgave 1

Bij deze opgave geldt dat deelproblemen in aparte methodes in de juiste class geprogrammeerd dienen te worden. Verder mogen gevraagde methodes, in latere onderdelen, indien nodig, gebruikt worden alsof ze gegeven waren.

Gebruik in deze opgave steeds een copy-constructor om objecten te kopieren.

Gegeven zijn de onderstaande interfaces en classes. De puntjes geven weggelaten onderdelen aan die niet nodig zijn voor deze opgave.

```
class Kleur {
    ...

    Kleur () {
        ... // de default constructor zet de kleur op 'kleurloos'
    }

    Kleur (Kleur k) {
        ...  // copy-constructor
    }

    public boolean equals (Object obj) {
        ...
    }

    ...
}

interface Kleurbaar {

    static final Kleur KLEURLOOS = new Kleur();

    void brengKleurAan (Kleur k);
    /* PRE  -
       POST - het object heeft de kleur k
    */

    void verwijderKleur ();
    /* PRE  -
       POST - het object heeft de kleur KLEURLOOS
    */

    Kleur kleur ();
    /* PRE  -
       POST - de kleur van het object is geretourneerd
    */

    ...
}
```

```
class Papiersoort implements Kleurbaar {
    pri  : int gewicht;   // in gram/vierkante meter

    Papiersoort (int g) {
        gewicht = g;
        brengKleurAan(KLEURLOOS);
    }

    ...
}

class VelPapier implements Kleurbaar {
    private Papiersoort gebruiktePapier;
    private double lengte, breedte;
    ...
}

class Pakpapier extends Papiersoort {
    ...
    Pakpapier () {
        ...
    }
    ...
}
```

a) Programmeer voor de class Papiersoort de methodes die vanwege het implementeren van de interface Kleurbaar noodzakelijk aanwezig moeten zijn. Voeg ook eventueel benodigde constanten, variabelen en hulpmethodes toe. Maak alle variabelen die eventueel worden toegevoegd private.

b) Geef een implementatie van de constructor

```
VelPapier (Papiersoort papier, Kleur k, double lengte, double breedte)
```

voor de class VelPapier die een vel papier aanmaakt van de gegeven papiersoort met de eveneens gegeven kleur, lengte en breedte.

c) Override in de class VelPapier de methode equals() uit de class Object met een public methode equals() waarmee gecontroleert kan worden of twee verschillende VelPapier-objecten een identieke inhoud hebben.

d) Programmeer voor de class VelPapier de methodes die vanwege het implementeren van de interface Kleurbaar noodzakelijk aanwezig moeten zijn. Voeg ook eventueel benodigde constanten, variabelen en hulpmethodes toe.

e) Als de volgende declaraties gegeven zijn:

```
static final int MAX_AANTAL= 10;
Papiersoort[] rij = new Papiersoort[MAX_AANTAL];
Kleurbaar gekleurdPapier;
Object object;
Kleur kleur = new Kleur();
Papiersoort papier = new Papiersoort(80);
VelPapier a4 = new VelPapier(papier, kleur, 21.0, 29.7);
Pakpapier pakpapier = new Pakpapier();
```

welke van de onderstaande statements zijn dan niet correct. Geef bij ieder statement dat niet correct is aan waarom het niet correct is.

```
1) rij[0] = kleur;          6) papier = a4;
2) rij[1] = papier;         7) papier = pakpapier;
3) rij[2] = a4;             8) object = (Papiersoort) a4;
4) rij[3] = pakpapier;      9) object = (Papiersoort) pakpapier;
5) gekleurdPapier = a4;    10) gekleurdPapier = pakpapier;
```

Opgave 2.

Gegeven is de onderstaande class voor de implementatie van knopen in een boom:

```
class Knoop {
    int data;
    Knoop links, rechts;

    Knoop (int data, Knoop links, Knoop rechts) {
        this.data = data;
        this.links = links;
        this.rechts = rechts;
    }

    Knoop () {
        this(0, null, null);
    }
}
```

a) Wanneer is een boom (tree) een binaire boom (binary tree)?

Een binaire boom is een hoop (heap) indien deze binaire boom voldoet aan de hoopvoorwaarden (heap storage rules) en een complete binaire boom (complete binary tree) is.

Welke zijn de hoopvoorwaarden?

Wanneer is een binaire boom een complete binaire boom?

Geef een recursieve implementatie van de onderstaande methode

```
boolean hoop (Knoop k);
/* PRE  - k is een referentie naar de wortel van een
          complete binaire boom
   POST - TRUE : de complete binaire boom met k als wortel
                 is een hoop.
          FALSE: de complete binaire boom met k als wortel
                 is geen hoop
*/
```

Kan een binaire zoekboom (binary search tree) een hoop zijn?

b) Wat is een queue?

Wat is een priority queue?

Beschrijf, voor het geval dat een priority queue geimplementeerd is met een hoop, hoe het toevoegen/verwijderen van een element aan/uit de priority queue uitgevoerd wordt.
N.B. er wordt dus niet gevraagd om iets te programmeren.

c) Gegeven is het volgende stuk van een interface van een rij

```
interface RijInterface {
    /* Elementen: getallen van het type int
       Structuur: lineair
       Domein   : alle rijen van ints
    */

    /*
    Rij ();
       PRE  -
       POST - Een nieuwe lege rij is aangemaakt.
    */

    void voegToe (int data);
    /* PRE  -
       POST - data is toegevoegd aan de rij als nieuwe laatste element
              achter het laatste element ten tijde van de PRE-conditie
    */
    // De rest van de RijInterface doet er voor deze opgave niet toe
}
```

en een class die deze interface implementeert

```
class Rij implements RijInterface, Iterator {
    // De implementatie doet er voor deze opgave niet toe.
}
```

Implementeer nu de onderstaande methode.

```
Iterator inOrder (Knoop k)
/* PRE  - k refereert naar de wortel van een binaire zoekboom
   POST - Een object van het type Iterator is geretourneerd.
          Dit object bevat de elementen in inorder volgorde.
*/
```

Opgave 3.

a) Bespreek hash tabellen (hash tables).

Geef in deze bespreking duidelijk aan welk het probleem is dat men met hash tabellen wil oplossen en hoe dat deze oplossing werkt. Gebruik in je bespreking de termen "hash table", key", "array index", "hash function", "perfect hash function", "collision", "collision resolution".

b) Geef een specificatie van een HashTabel class waarin entries worden opgeslagen die bestaan uit een key van het type Vector en data van het type Cloneable. Deze specificatie moet, buiten een beschrijving van de elementen, de stuctuur, het domein en de default constructor, operaties bevatten om data toe te voegen, om te testen of de hash tabel leeg is en om data op te vragen die onder een gegeven key is opgeslagen.
De operatie om data toe te voegen moet de PRE-conditie hebben dat er onder de key van de toe te voegen data nog geen data opgeslagen is.
De operatie die de onder een key opgeslagen data opvraagt, retourneert alleen een kopie van deze data. De geretourneerde data wordt niet verwijderd uit de hash tabel.

c) Geef een implementatie van de in het vorige onderdeel van deze vraag gespecificeerde hash tabel. Gebruik "chained hashing" voor het implementeren van de hash tabel.

Als hash function mag gebruik gemaakt worden van de formule "intwaarde key" % "aantal elementen tabel". De int waarde van de key mag bepaald worden met de methode hashCode() uit de class Object.

Het is voor deze vraag, in verband met de beschikbare tijd, niet nodig om, indien gebruik gemaakt wordt van hulpclasses voor de implementatie van de class HashTabel, specificaties voor deze hulpclasses te maken.

Beoordelingstabel

opgave	a	b	c	d	e	f	totaal
1	6	6	6	6	6		30
2	13	12	10				25
3	10	10	15				35
			TOTAAL				90

Het eindcijfer E van het gehele tentamen wordt als volgt uit het totaal T verkregen : E = T/10 + 1 .
Het eindcijfer [illegible] in het tweede deeltentamen wordt als volgt uit het totaal T verkregen : 9*T/60 + 1 .