Oplossing. Eén lijn toe te voegen aan klasse *Kaart*. Een andere klassenstructuur is ook mogelijk.

```
    protected abstract int tarief (int minuut, int dag);

public class GewoneKaart extends Kaart {
    private int tarief;

    public GewoneKaart (int krediet, int tarief) {
        super(krediet);
        this.tarief = tarief;
    }

    public int tarief (int minuut, int dag) {
     return tarief;
    }
}

public class SuperKaart extends Kaart {
    private int gratis;
    private int tarief;

    public SuperKaart (int krediet, int tarief) {
        super (krediet);
        this.tarief = tarief;
        this.gratis = 200;
    }

    public int tarief (int minuut, int dag) {
        if (gratis >= 0) {
            gratis --;
            return 0;
        } else {
         return tarief;
        }
    }
}

public class StudentenKaart extends Kaart {
    private int dal;
    private int piek;

    public StudentenKaart (int krediet, int dal, int piek) {
        super (krediet);
        this.dal = dal;
        this.piek = piek;
    }

    public int tarief (int minuut, int dag) {
        if (dag < 5 && minuut >= 8*60 && minuut < 18*60) {
            return piek;
        } else {
            return dal;
        }
    }
}
```

7. (3 pt – 10 ln.) Schrijf een programma Kans (= een volledige klasse die we vanop de opdrachtlijn kunnen uitvoeren) die afdrukt hoe groot de kans is dat drie getallen van twee cijfers in stijgende volgorde staan.

Om deze kans te bepalen, genereer je 100 000 willekeurige drietallen (x,y,z) van getallen van twee cijfers en tel je hoeveel keer $x \leq y \leq z$ geldt.

De gevraagde kans is de verhouding tussen het aantal drietallen dat in juiste volgorde staat en het totaal aantal drietallen dat je geprobeerd hebt. Het resultaat is een kommagetal tussen 0.0 en 1.0.

Oplossing.

```
public class Kans {

    public static final Random RG = new Random ();

    public static void main (String[] args) {
        int aantal = 0;
        for (int i =0; i < 10000; i++) {
            int x = 10 + RG.nextInt (90);
            int y = 10 + RG.nextInt (90);
            int z = 10 + RG.nextInt (90);
            if (x <= y && y <= z) {
                aantal ++;
            }
        }
        System.out.println (aantal / 10000.0);
    }
}
```

---

8. (3 pt – 27 ln.) Een klasse *Bestand* wordt gebruikt voor het inlezen van tekstbestanden. Deze klasse bevat twee methoden (waarvan je de broncode niet nodig hebt):

- Een constructor met één enkele parameter van het type String. Deze parameter is de naam van het bestand dat door dit object wordt beheerd.
- Een methode *readLine* zonder parameters. Telkens wanneer je deze methode oproept, geeft ze de volgende lijn van het bestand terug, in de vorm van een string. Wanneer het einde van het bestand is bereikt, geeft de methode **null** terug.

Het volgende fragment toont hoe deze klasse doorgaans wordt gebruikt:

```
Bestand bestand = new Bestand ("mijnbestand.txt");
String lijn = bestand.readLine ();
while (lijn != null) {
    // doe iets met de variabele lijn
    lijn = bestand.readLine();
}
```

Opgave: Schrijf een klasse *BestandMetKorteLijnen* met dezelfde functionaliteit als *Bestand*, behalve dat lijnen die 'te lang' zijn, worden opgesplitst in meerdere lijnen.

M.a.w., wanneer een lijn langer is dan een vooropgegeven lengte *len* dan zal de *readLine* slechts *len* tekens teruggeven, en wordt de rest van de lijn pas geretourneerd bij de volgende *readLine*. Bij heel

n van twee

t en het
n 1.0.

lange lijnen kan het zelfs gebeuren dat er nie… readLines nodig zij…
lijn is verwerkt. (Elke *readLine* mag hoogste… s teruggeven.)

De maximale lijnlengte *len* die je wenst te gebruiken, geef je mee als extra pa…
structor van *BestandMetKorteLijnen*. Je hoeft niet te controleren of *len* positief

Het volgende fragment toont hoe we deze nieuwe klasse wensen te gebruiken — i…
len = 60. (Beide fragmenten verschillen slechts in de eerste opdracht.)

```
Bestand bestand
    = new BestandMetKorteLijnen ("mijnbestand.txt", 60);
String lijn = bestand.readLine ();
while (lijn != null) {
    // doe iets met de variabele lijn
    lijn = bestand.readLine();
}
```

Oplossing.

```
public class BestandMetKorteLijnen extends Bestand {
    private String todo;
    private int len;

    public BestandMetKorteLijnen (String naam, int len) {
        super (naam);
        this.len = len;
        this.todo = null;
    }

    public String readLine () {
        String str;
        if (todo == null) {
            str = super.readLine ();
            if (str == null) {
                return null;
            }
        } else {
            str = todo;
            todo = null;
        }
        if (str.length() <= len) {
            return str;
        } else {
            todo = str.substring(len);
            return str.substring(0, len);
        }
    }
}
```

omerkingen. Er zijn andere manieren om hetzelfde resultaat te bere…
ruik van 'super.readLine()', van 'super(..)' en de 'extends B…
sultaat te bereiken. Belangrijk is het correct
n de 'extends Bestand' in de hoofding van de klasse.