Opgave i Java-programmering

I nederste venstre hjørne af et åbent Word-dokument står der nogle oplysninger om dokumentets egenskaber, fx *Side 2 af 3* og *1112 ord*. Hvis man klikker på *ord,* kommer der en boks frem, som indeholder supplerende oplysninger om antal afsnit, tegn og linjer i dokumentet.

Denne opgave går ud på at skrive en simpel klasse med metoder, som kan vise egenskaber ved et simpelt dokument, som blot består af tekst, der kan repræsenteres i et `String`-objekt.

Klassens navn er *OrdOpdeler*.[1] Ideen er, at simple tekster består af *ord* og *skilletegn*.
Hvis vi ser på følgende tekst:

”Gid du var i Skanderborg og blev der, kære Peter.”

så består den af 10 ord, og der indgår desuden tre forskellige skilletegn: mellemrum (blank), komma og punktum.
Mængderne af tegn, der kan indgå i ord og skilletegn, er disjunkte, dvs. et skilletegn kan ikke indgå i et ord.

Klassen *OrdOpdeler* har følgende attributter:

```
String     tekst;        //tekst, der skal opdeles i ord
String     skilleTegn;   //streng som indeholder de aktuelle skilletegn
int        indeks;  _    //pegepind til aktuelt tegn i tekst
```

Opgave 1
Du skal starte med at implementere klassens constructors, som har følgende signaturer:

```
OrdOpdeler(String kilde);
OrdOpdeler(String kilde, String skilleT);
```

som kan initialisere attributterne således:

tekst : initialiseres med parameteren *kilde.*
skilleTegn : sættes til parameteren *skilleT* eller til strengen ” , ” (blank, komma), hvis parameteren mangler
indeks : sættes til 0 og udpeger dermed det første tegn i *tekst*

Opgave 2
Metoden

```
boolean     erSkilleTegn(char tegn);
```

skal returnere *true*, hvis parameteren *tegn* er et skilletegn, dvs. indeholdt i attributten *skilleTegn*, og ellers returneres *false*.

[1] Det tilsvarende engelske begreb er *StringTokenizer;* men af pædagogiske årsager anvendes danske ord.

Opgave 3

Der er flere ord i *tekst*, hvis *indeks* ikke har passeret sidste tegn i *tekst* og der findes mindst ét tegn efter *indeks*, som ikke er et skilletegn.

Skriv metoden

```
boolean erDerFlereOrd();
```

som gennem returværdien kan vise, om der er flere ord tilbage.

Opgave 4

Nu skal du implementere klassens to centrale metoder:

```
String naesteOrd();
String naesteOrd(String skilleT);
```

Den første returnerer næste ord ud fra de skilletegn, der findes i *skilleTegn*.

Den anden opdaterer først attributten *skilleTegn,* således at den indeholder parameteren *skilleT* og returnerer derefter næste ord med udgangspunkt i den nye værdi af *skilleTegn,* som forbliver uændret indtil næste kald af denne metode med en ny værdi til *skilleTegn*.

Næste ord findes ved at tage udgangspunkt i pegepinden *indeks*' placering, således at:

1. Hvis der ikke er flere ord, returneres en tom streng.
2. Alle skilletegn før næste ord overspringes.
3. Alle tegn herefter, der ikke er et skilletegn, opsamles i det ord, der skal returneres.
4. Når et skilletegn herefter mødes, eller sidste tegn er passeret, standser indeks ved dette tegn og ordet returneres

Første gang, metoden kaldes på eksemplet øverst i teksten, returneres *Gid*, anden gang *du*, tredje gang *var* etc., naturligvis under forudsætning af at blank/mellemrum er defineret som et skilletegn.

Opgave 5

```
int antalOrd();
```

returner antallet af ord i teksten.

Opgave 6

```
int antalTegn();
```

returner antallet af tegn (ord og skilletegn) i teksten.

Opgave 7

```
int antalTegnUdenSkilleTegnOrd();
```

returner antallet tegn i teksten bortset fra skilletegn, altså summen af tegn i ordene.