## 16.0 Arrays

Ligesom det er nemt at skrive store arrays til skærmen, er det enkelt at skrive til harddisken. Data kan så indlæses i et andet eller det samme program. Hvis et filnavn genbruges overskrives data.

Hvis du har brug for at tilføje data til din fil, bliver du nødt til først at indlæse de gamle data i et array, tilføje de nye data til arrayet og til sidst skrive arrayet i en fil igen.

I de viste eksempler ligger filerne i \workspace\mitProjekt\minPakke

Du kan hente og gemme hvor du vil. Angiv blot stien

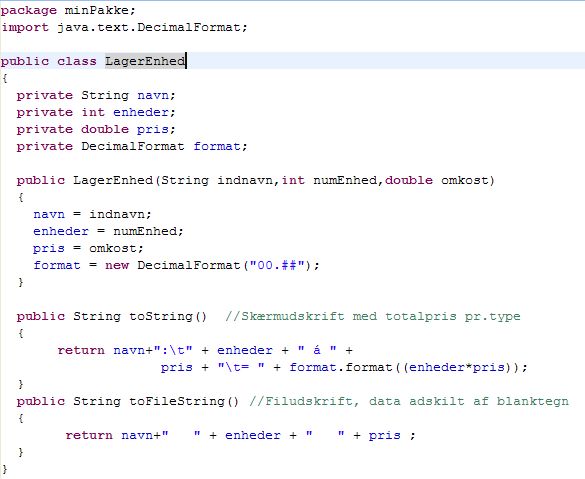
*String filnavn = d:\Documents and Settings\Dit Navn\Dokumenter\navnPaFil\_version2.txt*

### 16.1 Lager-enhed, enkelt objekt:

Eksemplerne vi skal se på opretter en lagerliste. Hver enhed på listen indeholder

* en type, f.eks. ”bord”, ”stol”, ”lampe” eller andre varer
* antal af varen (i heltal)
* pris på varen (pr. styk, decimaltal ok)

Til at holde styr på hver enhed, bruges et LagerEnheds-objekt. Denne programdel kan ikke køres, da den laver objekter og skal bruges af en programdriver (main-metode på næste side):



### 16.2 Skriv array til fil på disk:

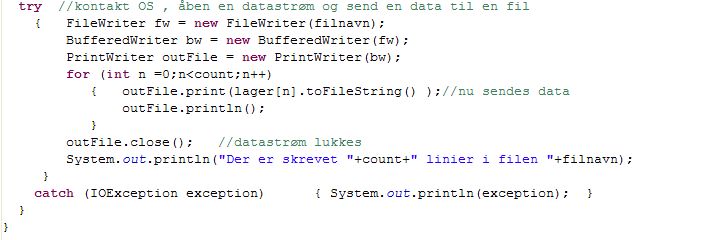
Her er et program, der kan køres. Det skriver nogle lagerenheder til en fil på harddisken: Detaljer på den følgende side.



12

20

26

30

31

39

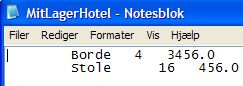
42

Linie 12: Her reserveres plads til et array på 100 elementer, Hver element er et objekt af typen ”LagerEnhed”.

Linie 20-30 indlæses et datasæt (navn,antal,pris) ad gangen.  
I linie 26 instantieres (skabes) hvert nyt objekt af typen LagerEnhed. Altså bliver der først lagt noget ind på den første plads nu.  
Bemærk at datasættet (navn,antal,pris) sendes med konstruktør-kaldet i denne linie.

Linie 31-42: Bemærk at operativsystemet forlanger, at der fanges fejl, når vi skal kommunikere ind (in) og ud (out) af programafviklingen og ned på harddisken. Derfor er linierne lagt ind i en **try-catch-struktur**.  
Linie 32-34: Her oprettes buffere til at rumme datastrømme, så det bliver muligt at gemme data i en fil på harddisken. Det er lidt omstændigt, at skulle bruge alle disse linier, men det er nødvendigt.  
I linie 36-37 sendes data af sted og i linie 39 lukkes datastrømmen pænt af, så filen bliver brugbar.

Filen bliver gemt i arkivet \*\\workspace\ditprojekt med mindre du angiver en fuld sti: c:\Documents and Settings\ Dit Navn\ programmering\jbproject\ditprojekt\navnPaaFilen.type

Filen kunne f.eks. se sådan ud, hvis den er gemt som \*.doc og åbnes i Notesblok:

### 16.2 Indlæs fil, overordnet virkemåde:

På næste side er et program vist, der læser linier i en fil. Data læses linie for linie. Hver linie skilles i tokens – dvs. stumper – for hver gang der er blanktegn.   
 Første stump anbringes i String-variablen *nav*n, næste stump i *enheder* og sidste lægges i *pris*.

Detaljerne:  
I linie 10 reserveres plads til et array med LagerEnheds-objekter.

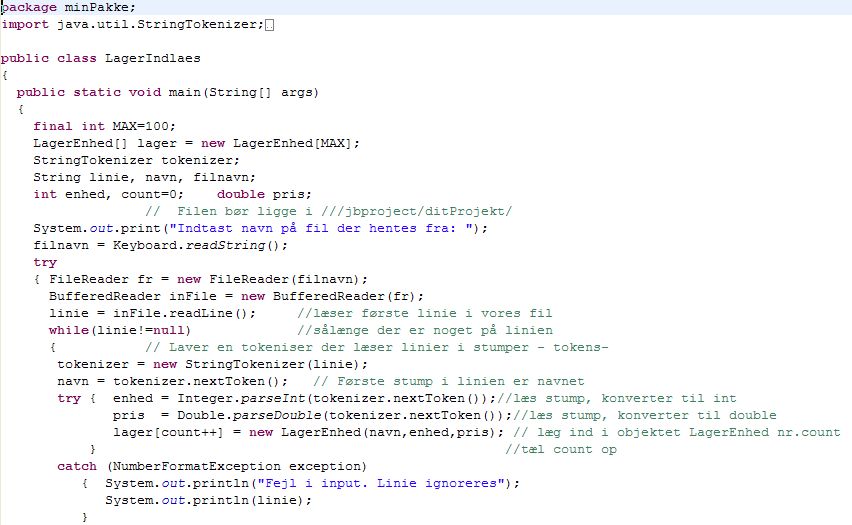
Blok linie 17-43: Da vi skal bruge in/out – faciliteter, skal eventuelle fejl fanges, så operativsystemet ikke belastes.

Linie 18-20 laves buffere og åbnes datastrøm til den ønskede fil.

Blok linie 21-34: While-løkken indlæser linie for linie indtil tom(*null*) linie.

Linie 24: Første token lægges i *navn*

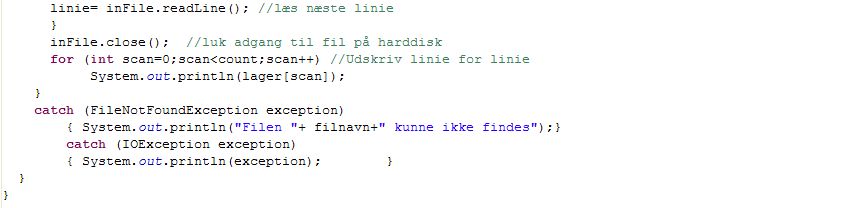
Linie 25-32: Her er indlagt en ekstra try-catch-struktur, som sikre at den indlæste token kan oversættes til tal. Linie 25-26 *enhed* og *pris* får værdi fra de næste to tokener.

Linie 35: Datastrømmen lukkes.

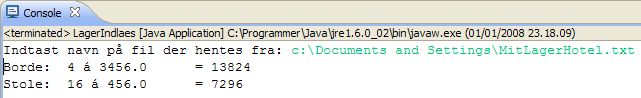
10

17  
18  
19  
20  
21

24  
25

32  
  
34  
35  
36

43

Linie 36-37: Arrayet udskrives til skærmen.  
Programafviklingen kunne se således ud:

Det er først nu, du kan begynde at gøre noget ved programmet, hvis du skal bruge data til noget spændende.

Hvis du har brug for at sortere i data enten før du gemmer dem eller når du indlæser dem, kan du

1. ændre i LagerEnhed:  
   Klassen skal implementere Comparable, hvilket betyder at du lover at lave en metode compareTo(Object other) som kan bruges til at sortere. Vil du sortere efter navn (alfabetisk) eller efter pris (talstørrelse)? Du bestemmer.
2. Lave en sorteringsmetode i din driverklasse. Den klasse hvor din main-metode også er.

Det er relevant at låne bøger om emnet Exceptions og Error-handling. Denne lektion introducerer kun et par eksempler, som kunne være nyttige for de fleste eksamensprojekt-programmer.

(Se lektion 17 og litteratur for sorteringsrutiner.)

------ slut på lektion JUJ ------