**IT for Erhvervsøkonomer**

UK d. 1.5.2010

**Obligatorisk opgave**

**Vejledende løsning**

Med ”vejledende løsning” menes her ikke, hvordan en aflevering fra jer kunne se ud – den beskæftiger sig kun med selve Matlab-løsningen, men ikke med rapporten og dokumentationen i øvrigt.

Der er i det følgende to bud på, hvordan opgaven kunne løses. Det første baserer sig på benyttelse af *funktioner*, det andet på benyttelse af *klasser*. Løsningerne tilstræber udelukkende at basere sig på det gennemgåede Matlab-pensum. Det er endvidere tilstræbt at vælge de enklest mulige løsninger af de enkelte problemer. For eksempel er der helt afstået fra at teste validiteten af parametre, der overføres – i stedet er der med kommentarer angivet, hvilke forudsætninger de enkelte parametre skal opfylde, hvis der er sådanne.

**1. Løsning baseret på funktioner**

Løsningen tager udgangspunkt i datastrukturer (vektorer) til aktiebeholdningen – disse vektorer må så for hver funktion parameteroverføres og returneres, hvis de ændres.

***datastrukturer***

Der er valgt flg. datastrukturer, der samtidigt initialiseres til at indeholde 3 aktier med tilknyttede værdier.

%aktiernes navne

vNavn=['ABC';'XYZ';'C10'];

%aktiernes pålydende værdier og antal

vVaerdi\_antal=[100,3;50,10;200,8];

%de seneste 10 dages kurser

vKurs=[85,86,85.5,85,85,84.5,83,82,83,83;

312.5,310,301,298.5,299,300,297.5,295,296,291;

192,195,201.5,201,202,204.5,208,207.5,207,208];

%til en given aktie svarer data med samme rækkenummer

Som det ses, er der her foretaget 2 forenklinger: 1) der er kun medtaget 10 kurser for hver aktie og 2) navnene på aktierne er på 3 tegn, og de er lige lange. I begge tilfælde er der dog intet i den følgende kode, der forhindrer at vælge flere kurser og længere navne. Dog skal navnene stadig være lige lange, hvis de ikke er det, skal de ”pastes” med blanke. Dette er et typisk eksempel på forudsætninger, man kan tillade sig at gøre sig ved løsningen af en opgave – de går jo ikke ud over principperne for opgavens løsning.

Disse sætninger kan enten placeres i en selvstændig m-fil, der så udføres hver gang en funktion skal testes fra kommandovinduet – eller de kan placeres først i en testfunktion. Den sidste mulighed er benyttet her – testfunktionen vises sidst i dette afsnit.

***hjælpefunktion***

En del af opgaverne kræver, at en bestemt aktie findes i datastrukturen. Det er derfor hensigtsmæssigt, at definere en funktion, der kan udføre denne opgave:

%hjælpefunktion:

%findes aktien med navnet "pNavn" i vektoren "vNavn"?

%hvis ja, returneres dets indeks i vektoren, ellers returneres -1

function indeks=findes(pNavn,vNavn)

indeks=-1;

[n,x]=size(vNavn); %find længden af navnene (x)

for i=1:n

if strcmp(pNavn,vNavn(i,1:x))

indeks=i;

end

end

end

***tilgang af aktier***

Det fremgår ikke eksplicit af opgaveteksten, om dette kun omfatter tilgang af nye aktier, eller om det også kan være tilkøb af flere stk. af en aktie, der allerede er i beholdningen. Det sidste er jo nok det mest naturlige, så det vælges her.

Man kunne lade én funktion udføre begge opgaver, men her er det valgt at splitte de 2 former for tilgang op i 2 funktioner. Dette kan gøres, fordi brugeren jo har mulighed for først at checke om en aktie findes med funktionen *findes()*.

%tilkøb af "antal" stk af en aktie, der ikke findes i vektoren "vNavn"

%de nye data indsættes sidst i de respektive vektorer

function [n,v,k]=tilgangNy(pNavn,pVaerdi,pAntal,pKurs,vNavn,vVaerdi\_antal,vKurs)

n=[vNavn;pNavn];

v=[vVaerdi\_antal;pVaerdi,pAntal];

k=[vKurs;pKurs];

end

%tilkøb af "antal" stk af en aktie, der allerede findes i vektoren "vNavn"

%forudsætning: aktien med "pNavn" findes i beholdningen

function v=tilgangFindes(antal,pNavn,vNavn,vVaerdi\_antal)

v=vVaerdi\_antal; %initialiser returværdien

indeks=findes(pNavn,vNavn); %find aktiens indeks

v(indeks,2)=vVaerdi\_antal(indeks,2)+antal; %læg det tilkøbte antal til

end

***afgang af aktier***

Her er de tilsvarende 2 muligheder, der også splittes i 2 funktioner:

%afgang af aktien med navn "pNavn" - alle i beholdningen

%det hermed opståede "hul" i vektoren lukkes ved at de efterfølgende

%objekter rykkes en plads "op"

%forudsætning: aktien med "pNavn" findes i beholdningen

function [n,v,k]=afgangAlle(pNavn,vNavn,vVaerdi\_antal,vKurs)

indeks=findes(pNavn,vNavn); %find aktiens indeks

[r,s]=size(vNavn); %find antal rækker i vektorerne

%initialisering af n,v og k

n=vNavn;

v=vVaerdi\_antal;

k=vKurs;

for i=indeks:r-1 %flyt alle rækker efter den, der skal slettes

n(i,:)=n(i+1,:); %en række op

v(i,:)=v(i+1,:);

k(i,:)=k(i+1,:);

end

n=n(1:r-1,:); %fjern sidste række

v=v(1:r-1,:);

k=k(1:r-1,:);

end

%afgang af "pAntal" stk af aktien med navn "pNavn"

%hvis "antal" er større eller lig med antallet af aktien i beholdningen,

%ændres den ikke

%forudsætning: aktien med "pNavn" findes i beholdningen

%forudsætning: "pAntal" < antallet i beholdningen

function v=afgangNogle(pAntal,pNavn,vNavn,vVaerdi\_antal)

indeks=findes(pNavn,vNavn); %find aktiens indeks

if pAntal<vVaerdi\_antal(indeks,2)

vVaerdi\_antal(indeks,2)=vVaerdi\_antal(indeks,2)-pAntal;

v=vVaerdi\_antal;

end

end

***opdatering af aktiekurser***

%opdatering af kursen for aktien med navnet "pNavn",

%den nye kurs er "pKurs"

%den nye kurs indsættes forrest i vektoren "vKurs" (indeks 1) -

%efter at alle kurser er rykket 1 række "ned" og den sidste droppet

%forudsætning: aktien med "pNavn" findes i beholdningen

function k=kursOpdatering(pNavn,pKurs,vNavn,vKurs)

k=vKurs; %initialiser returværdien

r=length(vKurs); %find antal rækker i vektorerne

indeks=findes(pNavn,vNavn); %find aktiens indeks

k(indeks,:)=[pKurs,vKurs(indeks,1:r-1)]; %indsæt den nye kurs

%som den første, ryk resten

end

***aktiebeholdningens samlede værdi***

%udregner porteføljens samlede værdi ved for hver aktie at gange

%kursværdien med antallet og så summere

function sv=samletVaerdi(vVaerdi\_antal,vKurs)

sv=0; %initialiser returværdien

[r,s]=size(vVaerdi\_antal); %find antal rækker i vektorerne

for i=1:r

sv=sv+vVaerdi\_antal(i,1)\*vVaerdi\_antal(i,2)\*vKurs(i,1)/100;

end

end

***liste over aktier, hvis kurs er steget de sidste 3 dage***

%udskriver liste over navnene på de aktier, hvis kurser er steget hver

%dag de sidste 3 dage, listen udskrives på filen "pFil"

function liste(pFil,vNavn,vKurs)

[r,s]=size(vKurs); %find antal rækker i vektorerne

navneliste=[]; %matrix til aktienavnene, hvis kurser er steget

for i=1:r

if vKurs(i,1)>vKurs(i,2) && vKurs(i,2)>vKurs(i,3) &&

vKurs(i,3)>vKurs(i,4)

navneliste=[navneliste,vNavn(i,:)];

end

end

fil=fopen(pFil,'w'); %åbn filen "pFil" for skrivning

if length(navneliste)>0

fprintf(fil,'%c%c%c\n',navneliste);

else

fprintf(fil,'ingen aktier med stigende kurser de sidste 3 dage');

end

fclose(fil);

end

***plotning af graf over kursudviklingen***

%plotter en graf over kursudviklingen de sidste 10 dage for aktien med

%navnet "pNavn"

%forudsætning: aktien med "pNavn" findes i beholdningen

function kursPlot(pNavn,vNavn,vKurs)

indeks=findes(pNavn,vNavn); %find aktiens indeks

[r,s]=size(vKurs); %find antal rækker i vektorerne

plot(1:s,fliplr(vKurs(indeks,1:s))); %"fliplr" vender vektoren om, så

%den svarer til fremadskridende

%tid

end

***testfunktion***

Funktionerne kan testes med flg. testfunktion. Bemærk at datastrukturerne hele tiden må parameteroverføres, og modtages som returværdier hvis de ændres.

function AktieTestFunktioner()

%aktiernes navne

vNavn=['ABC';'XYZ';'C10'];

%aktiernes pålydende værdier og antal

vVaerdi\_antal=[100,3;50,10;200,8];

%de seneste 10 dages kurser

vKurs=[85,86,85.5,85,85,84.5,83,82,83,83;

312.5,310,301,298.5,299,300,297.5,295,296,291;

192,195,201.5,201,202,204.5,208,207.5,207,208];

%til en given aktie svarer data med samme rækkenummer

'tilgangNy, DEF,80,5,98'

[vNavn,vVaerdi\_antal,vKurs]=tilgangNy('DEF',80,5,...

[98,97,97,98,96,95,94,95,96,93],vNavn,vVaerdi\_antal,vKurs);

vNavn

vVaerdi\_antal

vKurs

'tilgangFindes, DEF,8'

vVaerdi\_antal=tilgangFindes(8,'DEF',vNavn,vVaerdi\_antal)

'afgangAlle, DEF'

[vNavn,vVaerdi\_antal,vKurs]=afgangAlle('DEF',vNavn,vVaerdi\_antal,vKurs)

vNavn

'afgangNogle, ABC, 1'

vVaerdi\_antal= afgangNogle(1,'ABC',vNavn,vVaerdi\_antal)

'kursOpdatering, XYZ, 314'

vKurs=kursOpdatering('XYZ',314,vNavn,vKurs)

'samletVaerdi'

sv=samletVaerdi(vVaerdi\_antal,vKurs)

'liste'

liste('steget.data',vNavn,vKurs)

'kursPlot, XYZ'

kursPlot('XYZ',vNavn,vKurs)

end

**2. Løsning baseret på klasser**

Fordelen ved at benytte klasser er, at datastrukturerne ikke hele tiden skal parameteroverføres, for de ligger jo inde i objekterne. Desuden er fordelen jo som bekendt også, at vi får struktureret programmet i nogle logisk sammenhængende enheder. Det vil være naturligt i denne opgave at vælge 2 klasser: 1) *Aktie*, der repræsenterer en enkelt aktie og 2) *Aktiebeholdning*, der repræsenterer hele beholdningen.

Med denne opdeling kan de forskellige opgaver fordeles mellem de 2 klasser efter hvor de naturligt hører hjemme – og nogle opgaver kan splittes mellem de 2 klasser, så de hver udfører en del. Et eksempel på det sidste er listen over aktier, hvis kurser er steget de sidste 3 dage – den enkelte aktie ”ved jo” om dens egne kurser er steget, men den ”ved” jo ikke, hvilke andre aktiers kurser er steget, det må vi niveauet op til aktiebeholdningen for at finde ud af.

Selve den ”algoritmiske” problemstilling er jo nøjagtig samme som uden brug af klasser – så metoderne i klasserne fungerer principielt på samme måde som funktionerne i det foregående afsnit.

***klassen Aktie***

classdef Aktie

properties (Access=private)

navn

vaerdi

antal

kurser

end

methods (Access=public)

function obj=Aktie(n,v,a,k) %initialiserer aktien med navn, værdi,

%antal og kurser

obj.navn=n;

obj.vaerdi=v;

obj.antal=a;

obj.kurser=k; %"kurser" er en vektor med de sidste

%x dages kurser

end

%hjælpemetode:

%viser aktiens data på skærmen, dog kun med den aktuelle kurs

function display(obj)

disp([obj.navn,',',int2str(obj.vaerdi),...

',',int2str(obj.antal),',',int2str(obj.kurser(1))]);

end

%returnerer aktiens navn

function n=getNavn(obj)

n=obj.navn;

end

%tilgang af "antal" stk af aktien

function obj=tilgang(obj,antal)

obj.antal=obj.antal+antal;

end

%afgang af "antal" stk af aktien

%hvis "antal" er større eller lig med antallet af aktien i beholdningen,

%ændres den ikke

%forudsætning: "pAntal" < antallet i beholdningen

function obj=afgang(obj,antal)

if antal<obj.antal

obj.antal=obj.antal-antal;

end

end

%opdatering af kursen, den nye kurs er "pKurs"

%den nye kurs indsættes forrest i vektoren "kurser" (indeks 1) -

%efter at alle kurser er rykket 1 række "ned" og den sidste droppet

function obj=kursOpdatering(obj,pKurs)

obj.kurser=[pKurs,obj.kurser(1:length(obj.kurser)-1)]; %indsæt den

%nye kurs

end

%udregner værdien af aktieposten ved at gange

%kursværdien med antallet

function sv=aktieVaerdi(obj)

sv=obj.vaerdi\*obj.kurser(1)/100\*obj.antal;

end

%er kursen steget hver dag de sidste 3 dage?

%hvis ja returneres 1, hvis nej returneres 0

function b=kursSteget(obj)

if obj.kurser(1)>obj.kurser(2)...

&&obj.kurser(2)>obj.kurser(3)...

&&obj.kurser(3)>obj.kurser(4)

b=1;

else

b=0;

end

end

%plot en graf over kursudviklingen de sidste 10 dage

function kursPlot(obj)

plot(1:length(obj.kurser),...

fliplr(obj.kurser(1:length(obj.kurser))));

%"fliplr" vender vektoren om,

%så den svarer til fremadskridende tid

end

end

end

***klassen AktieBeholdning***

classdef AktieBeholdning

properties (Access=private)

beholdning %vektor med objekter af klassen Aktie

end

methods (Access=public)

function obj=AktieBeholdning()

obj.beholdning=[]; %initialiserer beholdningen til at være

%tom

end

%hjælpefunktion:

%findes aktien med navnet "pNavn" i beholdningen?

%hvis ja, returneres dets indeks i vektoren, ellers returneres -1

function indeks=findes(obj,pNavn)

indeks=-1;

for i=1:length(obj.beholdning)

if strcmp(pNavn,obj.beholdning(i).getNavn())

indeks=i;

end

end

end

%hjælpefunktion:

%viser beholdningen på skærmen, dog kun med dagens kurs

function display(obj)

for i=1:length(obj.beholdning)

obj.beholdning(i).display()

end

end

%tilføjer aktien "aktie" til beholdningen, den indsættes sidst

%i vektoren

function obj=tilgangNy(obj,aktie)

obj.beholdning=[obj.beholdning,aktie];

end

%tilføjer "pAntal" stk af en aktie med navnet "pNavn", der allerede

%findes i beholdningen

%forudsætning: aktien med "pNavn" findes i beholdningen

function obj=tilgangFindes(obj,antal,pNavn)

indeks=obj.findes(pNavn); %find aktiens indeks

obj.beholdning(indeks)=obj.beholdning(indeks).tilgang(antal);

end

%afgang af aktien med navn "pNavn" - alle i beholdningen

%det hermed opståede "hul" i vektoren lukkes ved at de efterfølgende

%objekter rykkes en plads "op"

%forudsætning: aktien med "pNavn" findes i beholdningen

function obj=afgangAlle(obj,pNavn)

indeks=obj.findes(pNavn); %find aktiens indeks

for i=indeks:length(obj.beholdning)-1 %flyt alle rækker efter

%den, der skal slettes, ét element op

obj.beholdning(i)=obj.beholdning(i+1);

end

obj.beholdning=obj.beholdning(1:length(obj.beholdning)-1);

%slet sidste element

end

%afgang af "pAntal" stk af aktien med navn "pNavn"

%hvis "antal" er større eller lig med antallet af aktien i beholdningen,

%ændres den ikke

%forudsætning: aktien med "pNavn" findes i beholdningen

%forudsætning: "pAntal" < antallet i beholdningen

function obj=afgangNogle(obj,pAntal,pNavn)

indeks=obj.findes(pNavn); %find aktiens indeks

obj.beholdning(indeks)=obj.beholdning(indeks).afgang(pAntal);

end

%opdatering af kursen for aktien med navn "pNavn", den nye kurs er

%"pKurs"

%forudsætning: aktien med navn "pNavn" findes

function obj=kursOpdatering(obj,pKurs,pNavn)

indeks=obj.findes(pNavn); %find aktiens indeks

obj.beholdning(indeks)=obj.beholdning(indeks).kursOpdatering(pKurs);

end

%udregning af aktiebeholdningens samlede værdi

function vaerdi=samletVaerdi(obj)

vaerdi=0;

for i=1:length(obj.beholdning)

vaerdi=vaerdi+obj.beholdning(i).aktieVaerdi();

end

end

%navnene på de aktier, hvis kurser er steget hver af de sidste 3

%dage, printes ud på en tekstfil med navn "pFilNavn"

function kursStegetListe(obj,pFilNavn)

navneliste=[];

for i=1:length(obj.beholdning)

if obj.beholdning(i).kursSteget()

navneliste=[navneliste;obj.beholdning(i).getNavn()];

end

end

fil=fopen('kurserSteget.data','w');

[r,c]=size(navneliste);

if r==0

fprintf(fil,'ingen aktier med stigende kurser');

else

fprintf(fil,'%c%c%c\n',navneliste');

end

fclose(fil);

end

%plot en graf over kursudviklingen de sidste 10 dage for

%aktien med navn "pNavn"

%forudsætning: aktien med "pNavn" findes i beholdningen

function kursPlot(obj,pNavn)

indeks=obj.findes(pNavn); %find aktiens indeks

obj.beholdning(indeks).kursPlot();

end

end

end

***testfunktion***

Klasserne kan testes med flg. testfunktion:

function AktieTest()

a1=Aktie('ABC',100,3,[85,86,85.5,85,85,84.5,83,82,83,83]);

a2=Aktie('XYZ',50,10,[312.5,310,301,298.5,299,300,297.5,295,296,291]);

a3=Aktie('C10',200,8,[192,195,201.5,201,202,204.5,208,207.5,207,208]);

'display: '

a1

['getNavn: ',a1.getNavn()]

'tilgang: '

a1=a1.tilgang(1)

'afgang:'

a1=a1.afgang(3)

'afgang, for mange'

a1=a1.afgang(10)

'kursOpdatering: '

a1=a1.kursOpdatering(84)

['aktieVaerdi: ',num2str(a1.aktieVaerdi())]

['kursSteget, a1:',num2str(a1.kursSteget()),'

,a2:',num2str(a2.kursSteget())]

'kursPlot'

a2.kursPlot();

ab=AktieBeholdning();

'tilgangNy, a1, a2 og a3'

ab=ab.tilgangNy(a1);

ab=ab.tilgangNy(a2);

ab=ab.tilgangNy(a3);

ab

'tilgangFindes, XYZ,1000'

ab=ab.tilgangFindes(1000,'XYZ')

'afgangAlle, C10'

ab=ab.afgangAlle('C10')

'afgangNogle, XYZ,200'

ab=ab.afgangNogle(200,'XYZ')

'kursopdatering, ABC,88'

ab=ab.kursOpdatering(88,'ABC')

ab=ab.kursOpdatering(89,'ABC')

ab=ab.kursOpdatering(90,'ABC')

'samletVaerdi'

ab.samletVaerdi()

'kursStegetListe'

ab.kursStegetListe()

'kursPlot ABC'

ab.kursPlot('ABC');

end