Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

TDT4105 IT Grunnkurs
Høst 2014

Øving 7

1 Teori

a) Konverter tallet 69 fra det desimale til det binære og det heksadesimale tallsystemet.

b) Nevn en måte for å gjøre feildeteksjon i digitale signaler.

c) Hva er en algoritme?

2 Kodeforståelse

a) Hva gjør følgende funksjon?

```
function table = unknown(table)
    a = table(1);

    for i = 1:length(table)-1
      table(i) = table(i+1);
    end

    table(length(table)) = a;
end
```

b) Hva gjør følgende funksjon?

```
function res = unknown2( str )

  n = size(str,2);

  for i = 1:n
    res(n-(i-1)) = str(i);
  end

end
```

3 Massemidtpunkt

Tenk deg en lang stang hvor massen er ulikt fordelt. Den første meteren veier 3 kg, den neste meteren veier 5 kg, den neste 2 kg, osv. Vi kan representere dette som en vektor `stang = [3, 5, 2, ...]`. Stangens massemidtpunkt er det punktet hvor det er like mye vekt på hver side.

a) Lag en funksjon som tar inn en tabellrepresentasjon av stangen og returnerer massemidtpunktet

Test funksjonen med følgende verdier:

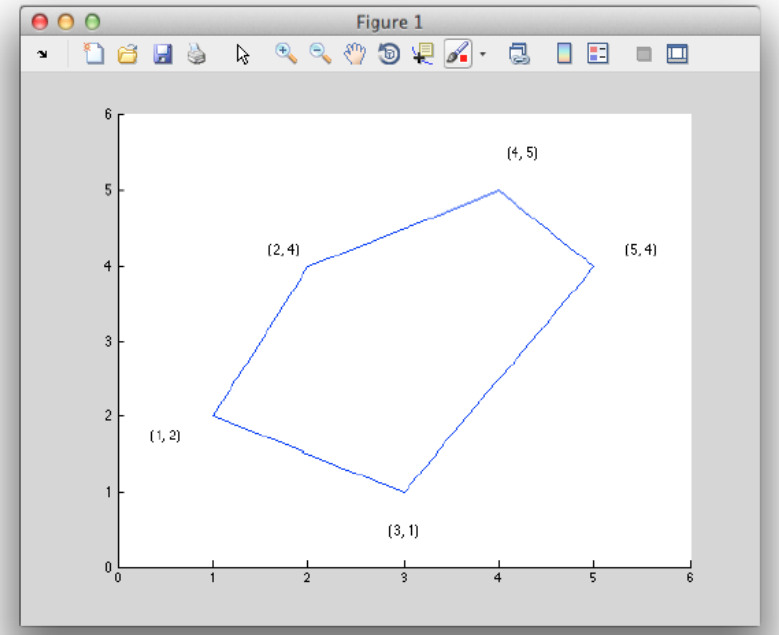
```
center_of_mass([1]) % 0.5
center_of_mass([1 1]) % 1
center_of_mass([1 1 1]) % 1.5
```

```
center_of_mass([3 1 3]) % 1.5
center_of_mass([1 2 3 4]) % 2.6667
```

b) Lag et skript som genererer en liste med tilfeldige tall og skriver ut massemidtpunktet.

4 Omkrets

Et polygon kan representeres som to vektorer med henholdsvis x- og y-koordinatene i xy-planet. Femkanten på figuren kan representeres slik: x = [1 2 4 5 3], y = [2 4 5 4 1].



a) Lag funksjonen `perimeter`. Funksjonen skal ta to vektorer, `x` og `y`, som angir punkter i xy-planet. Den skal returnere lengden langs kanten (omkretsen) av polygonet gitt av de to vektorene.

Lengden av en kant i polygonet er gitt av pytagoras' formel slik:

$$\sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}$$

For å regne ut lengden på den første kanten i eksempelet over får man:

$$x_i = 1, x_{i+1} = 2, y_i = 2, y_{i+1} = 4$$

Dette gir formelen $\sqrt{(1-2)^2 + (2-4)^2}$.

TIPS: Implementer pytagoras som en funksjon og bruk en for-løkke i `perimeter` funksjonen.

Test funksjonen slik:

```
perimeter([1 1 2], [1 2 2]) % 3.4142
perimeter([1 1 2 2], [1 2 2 1]) % 4
perimeter([1 2 4 5 3], [2 4 5 4 1]) % 11.7280
```

5 Strenghåndtering

a) Et palindrom er et ord som staves likt begge veier (f.eks. 'abba'). Lag en funksjon som returnerer `true` om en streng er et palindrom; `false` ellers.

b) Lag en funksjon som tar inn to strenger og sjekker om den første strengen inneholder den andre. Dersom den gjør det, returner posisjonen den forekommer på, ellers returner -1.

6 Investeringsstrategier

Denne oppgaven er ment som en øvelse i å løse et litt stort og abstrakt problem. Det som er viktig å tenke på da er hva som i oppgaveteksten som er relevant for akkurat den deloppgaven du skal løse. Man må også passe på å dele problemer opp i mindre og løsbare problemer, og løse kun ett og ett om gangen. Disse delproblemene tilsvarer veldig ofte enten funksjoner eller kontrollstrukturer som `if-else`, `for-` eller `while-løkker`.

Når man skal investere penger i aksjer finnes det mange forskjellige investeringsstrategier. I denne oppgaven skal du implementere inntil tre forskjellige strategier.

Vi forenkler børsmarkedet og sier at en aksje har en fast pris per dag og kan enten kjøpes eller selges kl 12:00. Det tillates også å investere i brøkdeler av en aksje. Det vil si at om man har 53,17 kr kan man investere i 53,17 % av en aksje som koster 100 kr.

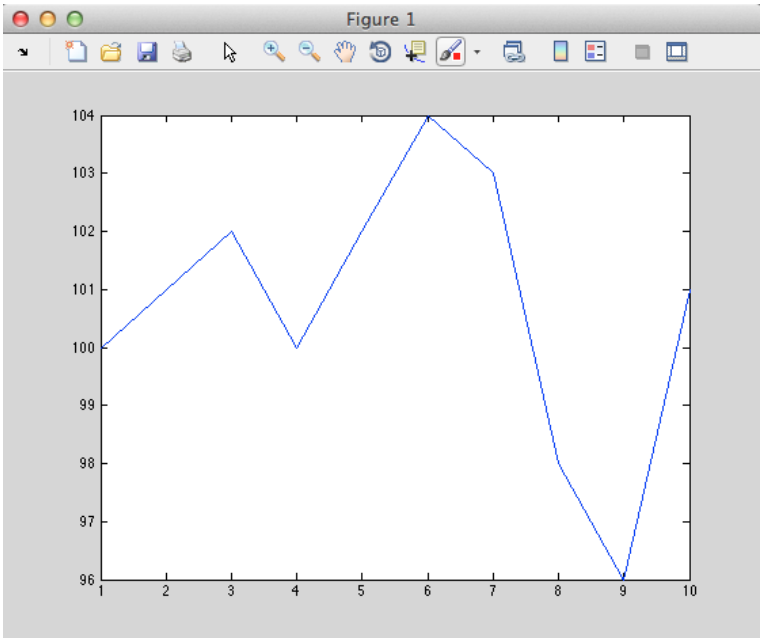
Aksjeprisene er gitt som en vektor av tall, hvor pris i tilsvarer prisen kl 12.00 på dag i. For eksempel:

Dag	1	2	3	4	5	6	7	8	9	10
Pris	100	101	102	100	102	104	103	98	96	101

Hvis du investerer 50 kr i denne aksjen på dag 1 og selger den på dag nummer 2 så har du tjent 50 * (101 / 100) = 0.50 kr.

a) Lag en vektor `closing_prices` med prisene over.

b) Plott aksjeprisene i en figur.
Resultatet skal se slik ut:



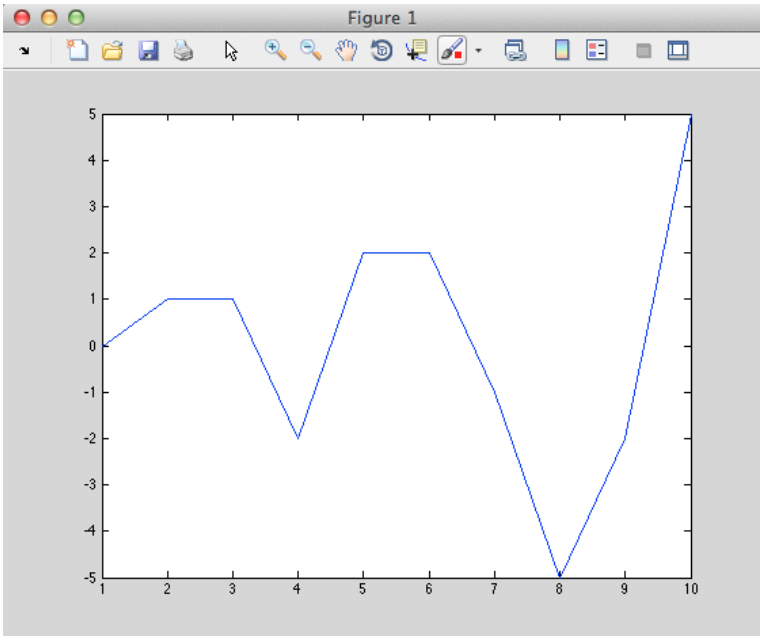
c) Lag funksjonen `daily_returns(closing_prices)` hvor `closing_prices` er aksjeprisene for et selskap i en gitt periode. Funksjonen skal returnere en vektor som viser dag for dag hvor mye en aksje har gått opp eller ned. Dvs.

$$daily\_returns_i = closing\_price_i - closing\_price_{i-1}$$

Daglig gevinst for dag nummer 1 skal være 0.
Gitt prisene over skal funksjonen returnere følgende vektor:

```
[0 1 1 -2 2 2 -1 -5 -2 5]
```

d) Plott daglige gevinster (`daily returns`) for aksjen over i en figur.
Resultatet skal se slik ut:



For å lage den første strategien må vi lage en funksjon som sjekker om prisen har gått opp de siste `n` dagene.

e) Lag funksjonen `is_going_up(n, i, daily_return)` som sjekker om alle elementene fra *i* og ned til *i-n+1* i `daily_return` er positive eller lik null. Hvis de er det skal funksjonen returnere `true`, hvis ikke skal funksjonen returnere `false`.
Test funksjonen slik:

```
is_going_up(2, 2, [0 1]) % skal skrive ut 1
is_going_up(2, 2, [0 -1]) % skal skrive ut 0
is_going_up(2, 3, [-1 0 1]) % skal skrive ut 1
is_going_up(3, 3, [1 0 1]) % skal skrive ut 1
```

f) Den første strategien du skal implementere heter *momentum*. Den baserer seg på at hvis en aksje er i ferd med å gå opp, så vil den fortsette å gå oppover. Det vil si kjøp aksjer som har hatt positiv eller 0 daglig gevinst de siste `n` dagene og selg aksjer som ikke har det.
Funksjonshodet for denne funksjonen skal se slik ut:

```
function returns = momentum( start_amount, n, closing_prices )
```

Implementer funksjonen etter denne pseudokoden:

```
dagligGevinst = kalkuler daglig gevinst
cash = start_amount
investert = 0
```

```
for dag fra n til lengde av closing_prices

Kalkuler kurs og oppdater investerte penger.
investert = investert * (dagen pris / gårsdagens pris)

hvis prisen har steget de siste n dagene
invester all cash i aksjen
ellers
ta ut alle pengene fra aksjen
slutt hvis

slutt for-løkke

svar = cash + investert
```

Testverdi:

```
% skal skrive ut 98.0579
momentum(100, 2, [100 101 102 100 102 104 103 98 96 101])
```

g) (Frivillig) Den andre strategien heter *contrarian* og mener at "what goes up must come down". I denne strategien vil du selge aksjer som er på vei opp, og kjøpe aksjer som er på vei nedover.

Hvorfor kan vi ikke bruke negasjonen av is_going_up(..) her?

Tips: Lag en funksjon som sjekker om prisen er på vei nedover.

Testverdi:

```
% skal skrive ut 103.0612
contrarian(100, 2, [100 101 102 100 102 104 103 98 96 101])
```