Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

TDT4110 IT Grunnkurs
Høst 2015

Løsningsforlag
Auditorieøving 2

1 **a)** Hva brukes en *enkelt* transistor til?

1. Lagre data permanent.
2. Regne ut summen av to tall
3. **Åpne og lukke en strøm-port**
4. Regne ut produktet av to tall

b) Hvilken av følgende er software?

1. Prosessor
2. **Operativsystem**
3. Kosebamse
4. Tastatur

c) Hva står RGB for?

1. Reliable Group Broadcast
2. Random Generated Bit
3. Red Green Black
4. **Red Green Blue**

d) Hvilket tall får man om man konverter tallet 12 fra desimaltall til binærtall?

1. 1001
2. 0101
3. **1100**
4. 1010

e) Hvilken oppgave har TCP-protokollen som brukes på Internett?

1. Tildeling av IP adresse, nettmaske og default gateway
2. **Tilby logiske forbindelser og multipleksing av disse**
3. Feilkorrigerende koding
4. Paritet, CRC eller Hash-funksjoner

Deloppgave	a	b	c	d	e
Svar	3	2	4	3	2

2 Strenger

a) Lag en funksjon `sok` som tar inn fire argumenter. Et av argumentene er en liste, `liste`, bestående av en mengde med ord, og de andre tre argumentene er forskjellige ord; `ord1, ord2, ord3`. Funksjonen skal returnere det største antall forekomster av `ord2` som finnes mellom `ord1` og `ord3`. Det vil si at du bare teller `ord2` etter at `ord1` har forekommet, og før `ord3` forekommer. Du må resette tellingen din hver gang `ord1` forekommer, og avslutte tellingen (til du igjen finner `ord1`) hver gang du finner `ord3`.

Løsning:

```
def sok(list, ord1, ord2, ord3):
    tmp, cPos,count = 0,0,0
    while (cPos < len(list)):
        if (list[cPos] == ord1):
            while (cPos<len(list) and list[cPos] != ord3):
                if (list[cPos] == ord2):
                    tmp+=1

                cPos+=1

        if (tmp>count):
            tmp, count = 0, tmp

        cPos+=1
    return count
```

3 Lister

a) Lag en funksjon som summerer sammen annenhvert element i en liste; altså elementnummer $1, 3, 5, \cdots$. Den skal deretter returnere summen.

Løsning:

```
def jumpSum(list):
    sum = 0
    for x in range(0,len(list), 2):
        sum+=list[x]

    return sum
```

b) Lag en funksjon som summerer sammen annenhvert element i en matrise, slik som de svarte rutene i figur 1 vist under. Funksjonen skal returnere summen. Denne funksjonen skal fungere for generelle matriser med `n` rader og `m` kolonner. Hint: Bruk gjerne funksjonen fra a) i funksjonen du skal lage.

Løsning:

```
def sumJumpMat(matrix):
    sum = 0
        for x in range(0,len(matrix)):
            if (x%2):
                sum+= jumpSum(matrix[x][0:len(matrix[0])]
            else:
                sum+= jumpSum(matrix[x][1:len(matrix[0])]

    return sum

% Eller:

def sumJumpMat(matrix):
    sum = 0
    for x in range(0,len(matrix)):
        sum+=jumpSum(matrix[x][x\%2:len(a[0])])

return sum
```

c) Lag en funksjon som summerer sammen alle hvite elementene i lister gitt av formen vist i Figur 2. Du kan anta at mønsteret fortsetter, altså at de svarte rutene havner på index $0, 2, 5, 9, 14, 20, \cdots$, funksjonen din må derfor kunne gjøre dette for lister av alle lengder. Du kan anta at listen er større enn 0. Funksjonen skal til slutt returnere summen.

Løsning:

```
def incSum(list):
    position = 1
    length = 1
    total = 0
    while (position <= len(list)):
        total += sum(list[position:position+length])
        position += 1 + length
        length +=1

    return total
```

4 Kodeforståelse

a) Hva blir skrevet ut om man kjører Kodesnutt 1?

Kodesnutt 1

```
liste = [4,3,3,6,6,6,3,4,3,3]
ant = 0
tempAnt = 0
index = 0
tempIndex = 0
for i in range(len(liste)):
        if (liste[i] % 2 == 0):
                if tempAnt == 0:
                        tempIndex = i
                tempAnt +=1
        else:
                if tempAnt > ant:
                        ant = tempAnt
                        index = tempIndex
                tempAnt = 0

print(liste[index:index+ant])
```

Løsning:
Listen `[6,6,6]` blir skrevet ut. Koden finner det lengste intervallet med partall i en liste og printer denne.

b) Hva blir skrevet ut om man kjører Kodesnutt 2?

Kodesnutt 2

```
def scram(a,b,c):
        a,b,c = c,b,a
        if b > c:
                return c,b,a
        else:
                return b,a,c

a,b,c = 1,2,3
a,c,b = scram(a,b,c)

print(a,b,c)
```

Løsning: 1 3 2

5 Fagplanlegging

a) Lag en funksjon `rmvDup` som tar inn en liste, fjerner alle duplikater fra listen og returnerer listen, uten duplikater.Eksempel:

$$[5,2,1,3,4,6] = \texttt{rmvDup}([5,2,2,1,3,4,3,1,4,5,6,6])$$

Løsning:

```
def rmvDup(list):
    newList = []
    for x in range(0,len(list)):
        if not (newList.count(list[x])):
            newList.append(list[x])

    return newList

print(rmvDup([5,2,2,1,3,4,3,1,3,6,6]))
```

Du har fått i ansvar å holde styr på informasjon om diverse fag. Relevant informasjon er: fagkode, antall personer som tar faget, og om det er obligatorisk øvingsopplegg (1 eller 0). I Kodesnutt 3 ser du hvordan listen kan se ut med 3 fag.

Kodesnutt 3

```
fag1 = ['TMA4100',315,1]
fag2 = ['TDT4110',402,1]
fag3 = ['TMA4170',16,0]

alleFag = [fag1,fag2,fag3]
```

b) Lag en funksjon `leggTilFag` som legger til et fag i listen. All relevant informasjon skal kunne skrives inn og lagres i listen `alleFag`. Om faget allerede er registrert skal bruker gjøres oppmerksom på dette. Funksjonen skal returnere listen `alleFag` til slutt.

Løsning:

```
def leggTilFag(alleFag):
    fagKode = input('Fagkode:')
    oppmeldte = int(input('Antall oppmeldte:'))
    obligatorisk = input('Obligatorisk ovingsopplegg:')
    obligatorisk = (obligatorisk == 'ja')

    for x in range(0,len(alleFag)):
        if (alleFag[x][0] == fagKode):
            print('Faget finnes fra for av!')
            return alleFag

    alleFag.append([fagKode, oppmeldte, obligatorisk])

    return alleFag
```

c) Lag en funksjon, `enkeltFag`, som skriver ut fagkoden på alle fag som ikke har et øvingsopplegg.

Løsning:

```
def enkeltFag(alleFag):
    for x in range(0,len(alleFag)):
        if not (alleFag[x][2]):
            print(alleFag[x][0])
```