Programmering og Modellering (PoM)
Ugeseddel 3 — Uge 38 — Deadline 19/9

Kim Steenstrup Pedersen, Katrine Hommelhoff Jensen, Knud Henriksen,
Mossa Merhi og Hans Jacob T. Stephensen

11. september 2014

3 Plan for ugen

I denne uge fortsætter vi med brugen af funktioner som element i vores programmer og vi vil fokusere på såkaldte *rekursive* funktioner og hvorledes konceptet rekursion kan anvendes som modelleringsværktøj. Et andet modelleringsaspekt vi berører er *konvergens* i numeriske beregninger, Newtons metode og bisektionsmetoden.

Vi skal også se hvorledes vi i Python kan skrive vores egne moduler og pakker som indeholder samlinger af funktioner og datatyper og hvad vi kan bruge disse til.

Endelige så fortsætter vi gennemgangen af talrepræsentation i computeren og vil i denne uge fokusere på flydende tal og faldgrupper ved regning med flydende tal.

I nogle af øvelsesopgaverne arbejder vi med "kædebrøker".

Forberedelse til forelæsningerne:

Til tirsdag:
Læs Guttag kap. 3.3 - 3.5, 4.3 - 4.5 samt noterne om Talrepræsentation af Klaus Grue.

Til torsdag:
Læs Guttag kap. 3.3 - 3.5, 4.3 - 4.5 samt noterne om excess N repræsentation at tal.

Gennemlæs også noterne om "kædebrøker" (beviserne kan overspringes).

3.1 Individuel opgave

Den 19/9 senest klokken 15:00 skal besvarelse af følgende opgave afleveres elektronisk via Absalon. Opgaven skal besvares individuelt og skal godkendes, for at du kan kvalificere dig til den afsluttende tag-hjem eksamen. Opgavebesvarelsen skal oploades via kursushjemmesiden på Absalon (find underpunktet `ugeseddel3` under punktet `Ugesedler og opgaver`). Kildekodefiler ("script"-filer) skal afleveres som "ren tekst", sådan som den dannes af `emacs`, `gedit`, `Notepad`, `TextEdit` eller hvilket redigeringsprogram man nu bruger (*ikke* PDF eller HTML eller RTF eller MS Word-format). Filen skal navngives *fornavn.efternavn*.38.py, mens andre filer skal afleveres som en PDF-fil med navnet *fornavn.efternavn*.38.pdf.

3i1 **Definition 1 (TL 6.1.1)** *Lad* $f : (a, b) \to \mathbb{R}$ *og lad* $x \in (a, b)$*. Da er* f *differentiabel i* x*, såfremt*

$$\lim_{h\to 0} \frac{f(x+h) - f(x)}{h}$$

eksisterer. I såfald er den afledte i x *givet ved*

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}$$

Vi udnytter dette og skriver

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} \quad \text{for } h \text{ tæt på } 0 \tag{1}$$

Antag $c \in (a, b)$ så $f(c) = 0$. Sæt $h = x - c$, så hvis h er lille, kan vi bruge (1) til at approksimere $f(c)$.

$$0 = f(c) = f(x + h) \approx hf'(x) + f(x) \implies h \approx -\frac{f(x)}{f'(x)}$$

Ideen er nu at nærme sig c iterativt ved at udnytte ovenstående. Vi begynder dermed med et startgæt $c_0 \in (a, b)$ og beregner

$$c_{n+1} = c_n - \frac{f(c_n)}{f'(c_n)} \tag{2}$$

a) Brug (1) og (2) til at skrive et program, der finder nulpunkterne af en kontinuert funktion. Bemærk pga. afrundingsfejl, som medfører at vi aldrig når 0, bør vi indføre terminationskriterier såsom

1. Givet et $\epsilon > 0$, $|f(c_{n+1}) - f(c_n)| < \epsilon$,
2. Givet et $\delta > 0$, $|c_{n+1} - c_n| < \delta$,

og man kan derudover indfører et maksimalt antal skridt $n_{\max}$, således at hvis $n > n_{\max}$, så terminerer beregningen.

b) Afprøv følgende eksempler:

$$\begin{aligned} f(x) &= x^2 - 3, \quad x \in [-2, 2] \\ f(x) &= x \sin\left(\frac{1}{x}\right), \quad x \in [0.01, 0.1] \\ f(x) &= \exp(x) - 1, \quad x \in [-1, 1] \end{aligned}$$

Kør dit program på disse funktioner og for nulpunktet $\hat{c}$ evaluerer funktionerne på føgende måde $f(\hat{c})$, $f(\hat{c} + \epsilon)$ og $f(\hat{c} - \epsilon)$, $\epsilon > 0$ og udskriv værdierne. Kommentér dine resultater i et dokument du aflevere i PDF format.

Hvis du har løst tirsdagsøvelsesopgave 3ti2, så prøve at sammenligne de to løsninger. Hvordan fungere metoden ift. løsningen til tirsdagsøvelsesopgave 3ti2?

c) Kan vi også bruge metoden til at bestemme maksimum og minimum af en funktion?

3.2 Tirsdagsøvelser

Besvarelser af disse opgaver skal ikke afleveres, men opgaverne forventes løst inden torsdag.

3ti1 Denne opgave er hentet fra kryptografi, læren om at skrive hemmelige meddelelser. *Klarteksten*, som er den meddelelse, der skal transmitteres, omskrives til *cifferteksten* på en sådan måde, at kun den rette modtager, som er i besiddelse af den nødvendige hemmelige *nøgle*, er i stand til at gendanne klartekst ud fra ciffertekst. Klartekst og ciffertekst opfattes som sekvenser af *tegn* fra et vist *kodealfabet*. I moderne kryptografi benyttes meget store alfabeter (typisk med mere end 10^{19} tegn), men i opgaverne her vil vi omtale klassiske metoder (fra før computernes tid)-

Dengang var kodealfabetet de almindelige bogstaver, idet man først fjernede alle mellemrum, kommaer og andre tegn fra klarteksten, så der udelukkende stod bogstaver tilbage, og man i øvrigt ved indkodning ikke skelnede mellem små og store bogstaver.

For nemheds skyld vil vi i første omgang løse opgaven for tekster, der kun bruger de 26 latinske bogstaver a–z.

Affin kodning er bestemt af to konstanter, vi kan kalde k_0 og k_1. Hvert bogstav i klarteksten omsættes (i uændret rækkefølge) til et bogstav i cifferteksten på følgende måde: Idet hvert bogstav svarer til et tal ($a \sim 0$, $b \sim 1$, $c \sim 2$, ..., $z \sim 25$), ganges dette tal med konstanten k_1, konstanten k_0 lægges til, og resultatet skal være bogstavet med det fremkomne nummer.

Hvis nummeret bliver for stort, bruges i stedet den rest, som fremkommer ved heltalsdivision med 26.

For $k_0 = 5$ og $k_1 = 3$ bliver klarteksten "code" (svarende til $[2, 14, 3, 4]$) for eksempel til cifferteksten "lvor" (svarende til $[2 \cdot 3 + 5, (14 \cdot 3 + 5)\%26, 3 \cdot 3 + 5, 4 \cdot 3 + 5] = [11, 21, 14, 17]$).

Skriv en funktion af tre argumenter, som for to konstanter k_0 og k_1, der skal bestemme den affine kode, samt en klartekst, danner den tilsvarende ciffertekst.

Løs bagefter, hvis tiden tillader det, den tilsvarende opgave for det danske alfabets 29 bogstaver, hvor altså $a \sim 0$, $b \sim 1$, $c \sim 2$, $\ldots$, $å \sim 28$.

Man kan vise, at klarteksten i almindelighed netop kan gendannes ud fra cifferteksten, hvis k_1 og antallet af bogstaver i alfabetet er indbyrdes primiske.

[Vink: Eftersom de latinske bogstaver a–z ligger samlet efter hinanden i tegnsættet ASCII, kan man med fordel anvende type konverteringsfunktionerne `ord` og `chr` til konvertering mellem tegn og tal (anvend `help` til at få information om disse funktioner). Eksempelvis ved

```
basis = ord('a')
chr(basis + n)
```

]

3ti2 (Hint: Denne opgave er god at løse inden du løser den obligatoriske opgave):

Theorem 1 (Skæringssætningen, TL 5.2.1) *Lad $f : [a, b] \to \mathbb{R}$ være en kontinuert funktion med $f(a)f(b) < 0$, dvs. $f(a)$ og $f(b)$ har modsat fortegn. Da eksisterer (mindst) et $c \in [a, b]$ så $f(c) = 0$.*

Lad $f : [a, b] \to \mathbb{R}$ være kontinuert og lad $c = \frac{a+b}{2}$.

a) Hvis $f(a)f(c) < 0$, hvad kan vi så sige om f's nulpunkter? Hvad sker der hvis $f(b)f(c) < 0$? Kan $f(a)f(c)$ og $f(b)f(c)$ begge være skarpt negative?

b) Skriv et rekursivt program, der finder et nulpunkt af en kontinuert funktion på et lukket interval, som udnytter resultaterne i opgave (a). Bemærk pga. afrundingsfejl, som medfører at vi aldrig når 0, bør vi indføre terminationskriterier såsom

1. Givet et $\epsilon > 0$, $|f(a) - f(c)| < \epsilon$,
2. Givet et $\delta > 0$, $|c - a| < \delta$,

Har du andre forslag til terminationskriterier?

c) Afprøv følgende eksempler:

$$f(x) = x^2 - 3, \quad x \in [-2, 2]$$
$$f(x) = x \sin\left(\frac{1}{x}\right), \quad x \in [0.01, 0.1]$$
$$f(x) = \exp(x) - 1, \quad x \in [-1, 1]$$

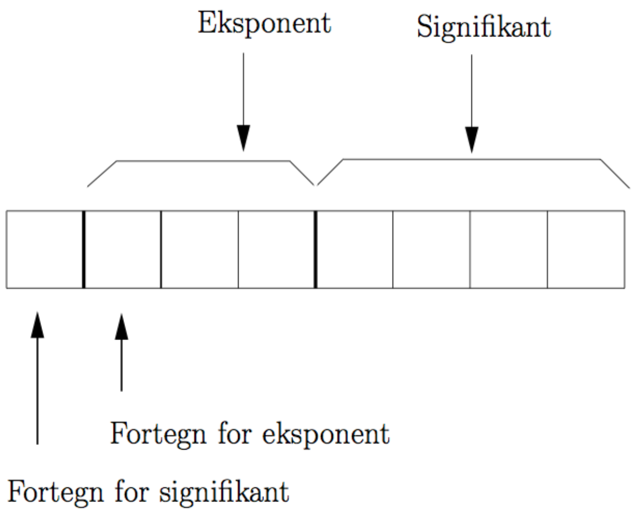
og kommentér dine resultater.

3.3 Torsdagsøvelser

Besvarelser af disse opgaver skal ikke afleveres, men opgaverne forventes løst inden tirsdag i efterfølgende uge.

3to1 Denne opgave handler om binære tal. Udregn i hånden og tjek bagefter dit resultat ved hjælp af Python.

1. Beregn $1101 - 11$.



Figur 1: Formatet for et 8 bit flydende tal.

2. Beregn $111 - 10000$.
3. Beregn $10000/11$.
4. Beregn $1001/11$.

3to2 Denne opgave omhandler 2-komplement repræsentation af binære heltal. Udregn i hånden og tjek bagefter dit resultat ved hjælp af Python.

1. Omregn tallet 247_{10} til et 16-bit 2-komplementtal.
2. Omregn tallet -247_{10} til et 16-bit 2-komplementtal.
3. Udvid tallet -247_{10} (fra 2) fra et 16-bit 2-komplementtal til et 32-bit 2-komplementtal.

3to3 I denne opgave betragtes binær 'floating point' notation med 8 bit. Notationen er som følger: Længst til venstre er en fortegnsbit efterfulgt af 3 bit til eksponenten, hvoraf første er eksponentens fortegn. De sidste fire bit er signifikanten, se Figur 1.

1. Tallet 2.75 kan skrives som $0.275 \cdot 10^1$. Opskriv 2.75 i binær 'floating point' notation med 8 bit.
2. Hvad er det nærmeste, man kan komme på tallet 2.80 med denne repræsentation? D.v.s. hvor stor vil afrundingsfejlen blive i titalssystemet?

3to4 Vi returnere til denne opgave fra Ugeseddel 2: Antallet af delmængder med netop k elementer udtaget fra en mængde med n forskellige elementer er $\binom{n}{k}$ (læses: "binomialkoefficienten n over k" eller blot "n over k") og kan beregnes af

$$\binom{n}{k} = \begin{cases} 0 & \text{hvis } k < 0 \text{ eller } k > n \\ 1 & \text{hvis } k = 0 \text{ eller } k = n \\ \binom{n-1}{k-1} + \binom{n-1}{k} & \text{for } 0 < k < n \end{cases} \tag{3}$$

Definer en rekursiv funktion `binomcoeff(n,k)`, som ud fra sammenhængen i (3) beregner og returnerer binomialkoefficienten n over k.

3to5 Ackermanns funktion $A(m, n)$ er defineret således for to heltal parametre m, n:

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1, 1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m, n-1)) & \text{if } m > 0 \text{ and } n > 0\ . \end{cases}$$

Skriv en funktion `ack` som evaluerer Ackermanns funktion. Anvend funktionen med `ack(3, 4)` som skal give resultatet 125. Hvad sker der for større værdier af m og n?

3to6 Den største fælles divisor (GCD) for a og b er det største tal som kan dividere begge tal uden en rest.

En måde at finde GCD for to tal er Euclids algoritme, som er baseret på at hvis r er resten efter division af a med b, så gælder der at $gcd(a,b) = gcd(b,r)$. Som base tilfælde kan vi anvende $gcd(a,0) = a$.

Skriv en funktion `gcd` som tager to parametre a og b og returnere den største fælles divisor. For mere hjælp se [`http://en.wikipedia.org/wiki/Euclidean_algorithm`].

3to7 Et palindrom er et ord som staves ens forfra og bagfra, såsom "mellem" og "rotor". En rekursiv definition af et palindrom er at første og sidste bogstav er ens, og ordet i midten mellem disse to bogstaver er et palindrom.

Følgende funktioner tager en streng som parameter og returnere henholdvis første og sidste bogstav samt midten:

```
def first(word):
    return word[0]

def last(word):
    return word[-1]

def middle(word):
    return word[1:-1]
```

1. Skriv ovenstående funktioner ind i et Python script og afprøv dem. Hvad sker der hvis du kalder `middle` med et ord på to bogstaver? Hvad med et bogstav? Hvad med den tomme streng som skrives som enten `''` eller `""`?
2. Skriv en funktion `er_palindrom` som tager en streng som parameter og returnere værdien `True` hvis ordet er et palindrom, ellers returneres værdien `False`.

3to8 I noternes afsnit om "Rationale tilnærmelser til en brøk" anføres det, at "Sideløbende med bestemmelsen af kvotienterne q_i [delnævnerne] fra(3) kan man beregne A'erne og B'erne [konvergenternes tællere og nævnere] med (1) ...".

Denne bemærkning dækker mere konkret over, at man for en brøk $\frac{a}{b}$ kan opstille skemaet

$$\begin{array}{lllll}
 & r_{-2}=a & A_{-2}=0 & B_{-2}=1 & \\
 & r_{-1}=b & A_{-1}=1 & B_{-1}=0 & \\
q_0=r_{-2}//r_{-1} & r_0=r_{-2}\%r_{-1} & A_0=q_0\cdot A_{-1}+A_{-2} & B_0=q_0\cdot B_{-1}+B_{-2} & K_0=\frac{A_0}{B_0} \\
q_1=r_{-1}//r_{-2} & r_1=r_{-1}\%r_0 & A_1=q_1\cdot A_0+A_{-1} & B_1=q_1\cdot B_0+B_{-1} & K_1=\frac{A_1}{B_1} \\
\dots & \dots & \dots & \dots & \dots \\
q_n=r_{n-2}//r_{n-1} & r_n=r_{n-2}\%r_{n-1} & A_n=q_n\cdot A_{n-1}+A_{n-2} & B_n=q_n\cdot B_{n-1}+B_{n-2} & K_n=\frac{A_n}{B_n} \\
\dots & \dots & \dots & \dots & \dots
\end{array}$$

Opgaven går ud på at gennemføre de viste beregninger, idet man dog ikke skal bruge alle de mange variabelnavne i skemaet, men kun løbende opretholde et "vindue" af tre på hinanden følgende rækker.

Definer en funktion `convrgntsRat(a,b,n)` af tre formelle parametre, sådan at et kald af formen `convrgntsRat(`a`,`b`,`n`)`, hvor a er et helt tal, b et positivt helt tal og n et ikke-negativt helt tal, skal give anledning til, at konvergenterne $K_0, K_1, \dots, K_{n-1}$ i kædebrøksudviklingen af $\frac{a}{b}$ udskrives i n på hinanden følgende linjer. Hvis K_i bliver lig med $\frac{a}{b}$ allerede for et $i < n$ (hvilket vil vise sig ved, at r_i bliver lig med 0), skal udskrivning dog standse der.

Eksempel: Kaldet `convrgntsRat(5233,1000,5)` skal give anledning til udskriftslinjerne

```
convrgntsRat(5233,1000,5)
5/1
21/4
68/13
157/30
382/73
```

Udfør kaldet `convrgntsRat(314159265,100000000,5)`, og genfind derved nogle af de almindeligt benyttede rationale tilnærmelser til π.

[Vink: Programmer ligningerne (1) fra noternes afsnit 3.1: Beregn A_{n-2}, A_{n-1} og A_n i tre variable `a0`, `a1` og `a2` og tilsvarende B_{n-2}, B_{n-1} og B_n i tre variable `b0`, `b1` og `b2`. Ajourfør disse variable inde i en løkke, hvorfra `str(a2)+'/'+str(b2)` udskrives, og som gentages det ønskede antal gange. Pas på ikke at komme til at dividere med 0.]

3to9 Betragt den uendelige kædebrøk

$$\varphi = [[1; 1, 1, 1, \dots]] = 1 + \cfrac{1}{1 + \frac{1}{1+\frac{1}{1+\dots}}}$$

Beregn φ's første fem konvergenter. Opstil ud fra den måde, hvorpå kædebrøken φ er indeholdt som en del af sig selv, en andengradsligning til bestemmelse af φ, og løs ligningen. (φ kaldes også for "det gyldne snit". φ eller *phi* er det fjerdesidste bogstav i det græske alfabet og udtales på dansk [fi].)

3to10 (For de specielt matematisk interesserede:) Bevis, at for en brøk $\frac{a}{b}$ vil der for de i noternes afsnit 3.2 "Rationale tilnærmelser til en brøk" definerede størrelser gælde

$$\begin{cases} a = r_{i-1}A_i + r_i A_{i-1} \\ b = r_{i-1}B_i + r_i B_{i-1} \end{cases} \qquad \text{for alle } i = -1, 0, 1, \dots, m$$