12. september 2019

MAT-INF 1100

Obligatorisk oppgave 1 av 2

Innleveringsfrist

Torsdag 26. september 2019, klokken 14:30 i Canvas (canvas.uio.no).

Instruksjoner

Du velger selv om du skriver besvarelsen for hånd og scanner besvarelsen eller om du skriver løsningen direkte inn på datamaskin (for eksempel ved bruk av LaTeX). Besvarelsen skal leveres som én PDF-fil. Scannede ark må være godt lesbare. Besvarelsen skal inneholde navn, emne og oblignummer.

Det forventes at man har en klar og ryddig besvarelse med tydelige begrunnelser. Husk å inkludere alle relevante plott og figurer. Studenter som ikke får sin opprinnelige besvarelse godkjent, men som har gjort et reelt forsøk på å løse oppgavene, vil få én mulighet til å levere en revidert besvarelse. Samarbeid og alle slags hjelpemidler er tillatt, men den innleverte besvarelsen skal være skrevet av deg og reflektere din forståelse av stoffet. Er vi i tvil om du virkelig har forstått det du har levert inn, kan vi be deg om en muntlig redegjørelse.

I oppgaver der du blir bedt om å programmere må du legge ved programkoden og levere den sammen med resten av besvarelsen. Det er viktig at programkoden du leverer inneholder et kjøreeksempel, slik at det er lett å se hvilket resultat programmet gir.

Søknad om utsettelse av innleveringsfrist

Hvis du blir syk eller av andre grunner trenger å søke om utsettelse av innleveringsfristen, må du ta kontakt med studieadministrasjonen ved Matematisk institutt (e-post: studieinfo@math.uio.no) i god tid før innleveringsfristen.

For å få adgang til avsluttende eksamen i dette emnet, må man bestå alle obligatoriske oppgaver i ett og samme semester.

For fullstendige retningslinjer for innlevering av obligatoriske oppgaver, se her:

www.uio.no/studier/admin/obligatoriske-aktiviteter/mn-math-oblig.html

LYKKE TIL!

Tilleggstekst spesielt for emnet.

- Kravet for å få bestått er at omtrent 70 % av oppgavene skal være godkjent. Alle deloppgaver vektes likt.
- Det er også viktig at du skriver slik at det er lett for andre å forstå hva du mener og forklarer hvorfor du mener resultatene av kjøringer, plott og lignende er rimelige eller urimelige.

Oppgaver

Oppgave 1. Vi skal se på differensligningen

$$x_{n+2} - 2x_{n+1} - 2x_n = 0, \quad \text{med } x_0 = 1 \text{ og } x_1 = 2. \tag{1}$$

a) Lag et dataprogram som simulerer denne ligningen og skriver ut følgen $x_2, x_3, \ldots, x_{100}$.

b) Simuler ligningen og skriv ut følgen $x_2, x_3, \ldots, x_{100}$ når startverdien x_1 endres til $x_1 = 1 - \sqrt{3}$

c) Vis at den generelle løsningen av ligningen $x_{n+2} - 2x_{n+1} - 2x_n = 0$ er på formen

$$x_n = C(1 - \sqrt{3})^n + D(1 + \sqrt{3})^n$$

og at initialverdiene $x_0 = 1$ og $x_1 = 1 - \sqrt{3}$ bestemmer den endelige løsningen til å være $x_n = (1 - \sqrt{3})^n$.

d) Sjekk om den analytiske løsningen i (c) stemmer med dine beregninger i (b) og forklar eventuelle avvik.

e) (Om du har lyst og tid). Bruk simuleringene i (b) til å estimere avrundingsenheten (the round-off unit) på maskinen din.

Oppgave 2. Binomialkoeffisienten $\binom{n}{i}$ er definert som

$$\binom{n}{i} = \frac{n!}{i!\,(n-i)!} \tag{2}$$

der $n \geq 0$ er et ikke-negativt heltall og i er et heltall i intervallet $0 \leq i \leq n$.

Binomialkoeffisientene dukker opp i mange ulike sammenhenger og må ofte beregnes på datamaskin. Siden alle binomialkoeffisientene er heltall (divisjonen i (2) gir aldri noen rest) så er det rimelig å bruke heltallige variable i slike beregninger. For små verdier av n og i går det bra, men for større verdier får vi fort problemer fordi både teller og nevner i (2) lett blir større enn det største heltallet som kan representeres med 32-

eller 64-bits heltall selv om binomialkoeffisienten i seg selv ikke er så stor. I mange språk vil dette føre til en form for «overflow», men selv i et språk som Python som unngår dette ved at innebygget programvare kommer til unnsetning, vil beregningene gå langt saktere enn ellers. Vi kan bruke flyttall i stedet, men selv da vil vi lett få «overflow» underveis i beregningene. I denne oppgaven skal vi se hvordan vi kan unngå slike problemer.

Hvis vi ser nærmere på definisjonen (2) legger vi merke til at vi kan forkorte i stor skala,

$$\binom{n}{i} = \frac{1 \cdot 2 \cdots i \cdot (i+1) \cdots n}{1 \cdot 2 \cdots i \cdot 1 \cdot 2 \cdots (n-i)} = \frac{i+1}{1} \cdot \frac{i+2}{2} \cdots \frac{n}{n-i}.$$

Ved hjelp av produktnotasjon kan vi derfor skrive $\binom{n}{i}$ som

$$\binom{n}{i} = \prod_{j=1}^{n-i} \frac{i+j}{j}. \tag{3}$$

a) Skriv et program som beregner binomialkoeffisienter ved hjelp av formelen (3). Test metoden på eksemplene

$$\binom{5000}{4} = 26010428123750,$$

$$\binom{100000}{60} = 1.18069197996257 \cdot 10^{218},$$

$$\binom{1000}{500} = 2.702882409454366 \cdot 10^{299}.$$

Hvorfor må du bruke flyttall og hvilke resultater får du?

b) Er det nå mulig at du underveis får «overflow» om binomialkoeffisienten du skal beregne er mindre enn det største flyttallet som kan representeres på maskinen din?

c) I vår utledning av (3) forkortet vi $i!$ mot $n!$ i (2). En alternativ metode kan utledes ved å forkorte $(n-i)!$ mot $n!$ isteden. Utled denne alternative metoden på samme måte som over og diskuter når de to metodene bør brukes (du trenger ikke programmere denne metoden, det holder å argumentere matematisk).

Oppgave 3. Følgende Python-program er gitt:

```
1 from random import random
2
3 antfeil = 0; N = 100000
4
```

```
5  for i in range(N):
6      x = 500*random(); y = random(); z = random()
7      res1 = (x + y) + z
8      res2 = x + (y + z)
9
10     if res1 != res2:
11         antfeil += 1
12         x0 = x; y0 = y; z0 = z
13         ikkeass1 = res1
14         ikkeass2 = res2
15
16 print (100. * antfeil/N)
17 print (x0, y0, z0, ikkeass1 - ikkeass2)
```

En kjøring av programmet ga utskriften

```
24.933
308.8701703294632 0.5322944857397117 0.06539200494708597 5.684341886080802e-14
```

a) Forklar hva programmet gjør og hva utskriften forteller oss.

b) Bytt ut line 6 i programmet ovenfor, med

```
1      x = random(); y = 500*random(); z = random()
```

og kjør det på nytt. Du skal nå se at det første tallet som blir skrevet ut blir ulikt det første tallet som ble skrevet ut over (d.v.s. 24.933). Prøv å forklare resultatene.

Lykke til!