Danmarks Tekniske Universitet

Danmarks Tekniske Universitet

Skriftlig prøve, 9. december 2020

Kursusnavn:	Programmering af matematisk software
Kursusnummer:	02635
Hjælpemidler:	Alle hjælpemidler
Varighed:	4 timer
Vægtning:	80/100

Skriftlig eksamen
Programmering af matematisk software

Denne eksamen indeholder 18 opgaver: 14 multiple-choice-opgaver (opgave 1–14) og 4 programmeringsspørgsmål (opgave 15–18). ZIP-filen som er vedhæftet eksamensopgaven indeholder et tekstdokument til din svar på multiple choice-spørgsmålene samt skabeloner til programmeringsspørgsmålene. Du skal bruge disse filer til din besvarelse.

---

1. (4 point) Egenskaber ved flydende kommatal.
   (a) Kan tallet $1/3$ repræsenteres som et flydende kommatal med endelig præcision i base 10?
      A. Ja
      B. Nej

   (b) Kan tallet $1/3$ repræsenteres som et flydende kommatal med endelig præcision i base 2?
      A. Ja
      B. Nej

2. (2 point) Betragt følgende kodestump:

```
double x = 33.0/395.0, y = 1.0/12.0;
double z = x-y;
```

   Hvor mange betydende binære cifre (dvs. bits) mistes i forbindelse med subtraktionen $z = x - y$?
      A. Mindst 6 bits og højst 7 bits.
      B. Mindst 7 bits og højst 8 bits.
      C. Mindst 8 bits og højst 9 bits.
      D. Mindst 9 bits og højst 10 bits.

3. (2 point) Genkald at binary64 er et binært flydende kommatalsformat med præcision 53. Hvor mange forskellige binary64 flydende kommatal er der i intervallet $[1, 2)$?
      A. 2^{51}
      B. 2^{52}
      C. 2^{53}
      D. 2^{54}

4. (2 point) Betragt følgende repræsentation af et flydende kommatal i base b

$$(d_0.d_1d_2\ldots d_{p-1})_b \cdot b^E.$$

Hvornår kaldes denne repræsentation *normal*?

A. Når præcisionen p er et lige tal.

B. Når eksponenten E er positiv.

C. Når eksponenten E er nul.

D. Når det forreste ciffer, d_0, er forskellige fra nul.

E. Når basen b er lig 2.

5. (4 point) Den teoretiske forbedring i eksekveringshastighed for en opgave som eksekveres på p processorer kan udtrykkes ved

$$S(p) = \frac{T(1)}{T(p)} = \frac{fT(1) + (1-f)T(1)}{(f/p)T(1) + (1-f)T(1)}$$

hvor $T(p)$ er eksekveringstiden på p processorer (realtid), og f er den såkaldte parallelle brøkdel af opgaven. Hvis for eksempel 60% af en opgave kan paralleliseres, så er $f = 0.6$.

(a) Vi antager nu, at vi har en computer med $p = 8$ processorer og en opgave af hvilken den parallelle brøkdel er $f = 0.95$. Er det muligt at opnå en faktor 6 forbedring i eksekveringshastigheden ved at udnytte alle 8 processorer?

A. Ja

B. Nej

(b) Hvor mange af de 8 processorer er som minimum nødvendige for at opnå en forbedring med en faktor 4?

A. 4

B. 5

C. 6

D. 7

E. 8

6. (2 point) Hvad er *stride* af et array?

A. Antallet af elementer i arrayet.

B. Størrelsen af arrayet i bytes.

C. Afstanden i hukommelsen mellem på hinanden følgende elementer i arrayet.

D. Datatypen af elementerne i arrayet.

7. (4 point) Antag at matricen

$$C = \begin{bmatrix} C_{1,1} & C_{1,2} & C_{1,3} & C_{1,4} & C_{1,5} \\ C_{2,1} & C_{2,2} & C_{2,3} & C_{2,4} & C_{2,5} \\ C_{3,1} & C_{3,2} & C_{3,3} & C_{3,4} & C_{3,5} \end{bmatrix}$$

er gemt som et én-dimensionelt rækkevist `double` array af længde 15, og lad variablen `pC` være en peger til det første element af C (dvs. variablen `pC` indeholder adressen på $C_{1,1}$).

(a) Hvilket af følgende udtryk er en peger til elementet $C_{2,2}$?

A. `pC+4`
B. `pC+5`
C. `pC+6`
D. `pC+7`

(b) Antag nu at vi ønsker at udføre operationen

$$\begin{bmatrix} C_{2,1} \\ C_{3,1} \end{bmatrix} \leftarrow \begin{bmatrix} C_{2,2} & C_{2,3} & C_{2,4} \\ C_{3,2} & C_{3,3} & C_{3,4} \end{bmatrix} \begin{bmatrix} C_{1,5} \\ C_{2,5} \\ C_{3,5} \end{bmatrix}$$

ved hjælp af BLAS-funktionen DGEMV, som udfører matrix-vektor-operationer af formen $y \leftarrow \alpha Ax + \beta y$ og $y \leftarrow \alpha A^T x + \beta y$. Hvilket at følgende udtryk skal bruges som input til DGEMV som en peger til første element af vektoren y?

A. `pC+4`
B. `pC+5`
C. `pC+6`
D. `pC+7`

8. (4 point) Betragt funktionen $f(x) = \int_0^1 e^{tx}\,\mathrm{d}t$ som også kan udtrykkes som

$$f(x) = \begin{cases} 1, & x = 0, \\ \frac{e^x - 1}{x}, & x \neq 0. \end{cases}$$

(a) Problemet at evaluere f når x er tæt på 0 er

A. velkonditioneret.

B. dårligt konditioneret.

C. hverken velkonditioneret eller dårligt konditioneret; det afhænger af, hvordan f evalueres og/eller den numeriske præcision.

(b) Betragt følgende implementering af en funktion, som evaluerer f numerisk:

```
double feval(double x) {
    if (x == 0)
        return 1.0;
    else
        return (exp(x)-1.0)/x;
}
```

Er dette en numerisk stabil algoritme til at evaluere f?

A. Ja

B. Nej

9. (4 point) Betragt følgende implementering af sekventiel summation:

```
double sequential_sum(double * arr, int n) {
    double sum = 0.0;
    for (int i=0;i<n;i++)
        sum += arr[i];
    return sum;
}
```

(a) Referencer til elementerne af arrayet `arr` er

A. rummeligt lokale.

B. tidsligt lokale.

(b) Referencer til variablen `sum` er

A. rummeligt lokale.

B. tidsligt lokale.

10. (2 point) En reference i C++ er det samme som en peger i C.
    A. Sandt
    B. Falsk

11. (2 point) En klasse i C++ er
    A. en variabel.
    B. en abstrakt datatype.
    C. en matematisk operator.
    D. en adresse i hukommelsen.

12. (4 point) Betragt følgende rekursive definition af en klasse af polynomier:

$$
\begin{aligned}
H_0(x) &= 1 \\
H_1(x) &= 2x \\
H_n(x) &= 2xH_{n-1}(x) - 2nH_{n-2}(x), \qquad n = 2, 3, \ldots
\end{aligned}
$$

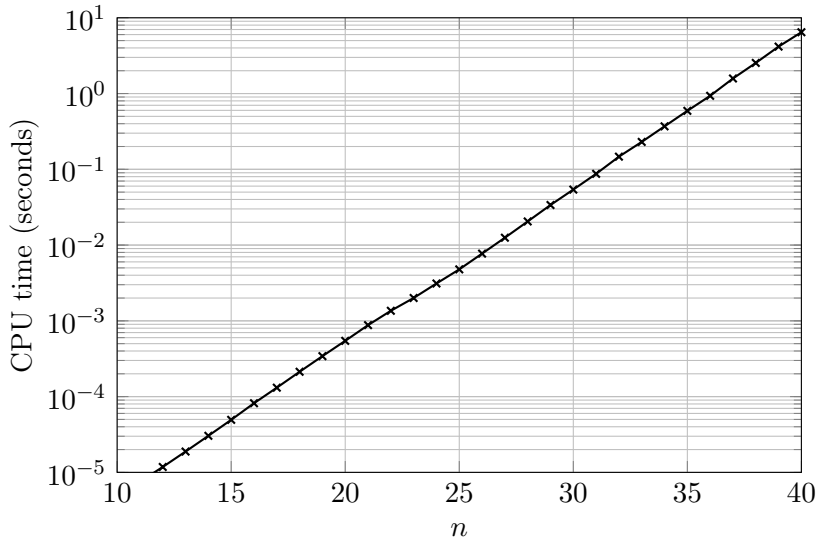
    (a) Hvilken form for rekursion er dette?
        A. Enkel rekursion
        B. Multipel rekursion

    (b) Hvad er kompleksiteten af følgende algorithme til at beregne $H_n(x)$?

```
double H(double x, unsigned int n) {
   if (n==0) return 1.0;
   if (n==1) return 2*x;
   return 2*x*H(x,n-1)-2*n*H(x,n-2);
}
```

        A. $O(n)$ plads og $O(n)$ tid
        B. $O(2^n)$ plads og $O(n)$ tid
        C. $O(n)$ plads og $O(2^n)$ tid
        D. $O(2^n)$ plads og $O(2^n)$ tid

13. (4 point) Følgende plot viser CPU-tiden, som kræves af en algoritme for at løse et givent problem som en funktion af problemets dimension n.



Hvad er algoritmens tidskompleksitet?

A. $O(\log n)$

B. $O(n)$

C. $O(n^\alpha), \alpha > 1$

D. $O(2^n)$

14. (4 point) Hukommelseshåndtering.

(a) Et *cache miss* er resultatet af et hukommelseslæk.

A. Sandt

B. Falsk

(b) Betragt følgende udtryk:

```
int myArray[64];
```

Hvilken form for hukommelsesallokering vil blive brugt til at allokere arrayet `myArray`?

A. Automatisk allokering

B. Dynamisk allokering

15. (8 point) Implementér en funktion i C, som evaluerer funktionen

$$f(x) = \frac{1 - \cos(x)}{\sin(x)}.$$

Krav

- Din funktion skal have følgende prototype:

```
double feval(double x);
```

- Funktionen må ikke skrive til `stdout`.
- Benyt skabelonen `exam_e20_question15.c` til din implementering. Skabelonen er inkluderet i ZIP-filen, som er vedhæftet eksamensopgaven.

16. (8 point) Implementér en funktion i C, som evaluerer

$$g(x) = \sqrt[3]{x^2 + 2} - \sqrt[3]{x^2}, \quad x \in \mathbb{R}.$$

Hint: Brug identiteten

$$(a - b)(a^2 + ab + b^2) = a^3 - b^3$$

til at omskrive $g(x)$ med henblik på at undgå tab af præcision for visse værdier af x.

Krav

- Din funktion skal have følgende prototype:

```
double geval(double x);
```

- Funktionen må ikke skrive til `stdout`.
- Benyt skabelonen `exam_e20_question16.c` til din implementering. Skabelonen er inkluderet i ZIP-filen, som er vedhæftet eksamensopgaven.

17. (8 point) Lad $x = (x_1, \ldots, x_n)$ og $w = (w_1, \ldots, w_n)$ være to vektorer af længe n med positive elementer. Funktionen

$$H(x; w) = \frac{\sum_{i=1}^{n} w_i}{\sum_{i=1}^{n} \frac{w_i}{x_i}}$$

er en vægtet harmonisk middelværdi af $x_1, \ldots, x_n$ med vægtene $w_1, \ldots, w_n$.

Implementér en funktion i C, som evaluerer $H(x; w)$.

Krav

- Din funktion skal have følgende prototype:

```c
double weighted_harmonic_mean(int n, double * x, double * w);
```

  Inputtet `n` er længden af vektorerne x and w, `x` er en peger til første element af vektoren x og `w` er en peger til første element af w.

- Funktionen skal returnere $H(x; w)$, hvis et eller flere input er gyldige.
- Funktionen må ikke skrive til `stdout`.
- Benyt skabelonen `exam_e20_question17.c` til din implementering. Skabelonen er inkluderet i ZIP-filen, som er vedhæftet eksamensopgaven.

18. (12 point) Lad L være en kvadratisk matrix af orden n på formen

$$L = \begin{bmatrix} a_1 & & & & \\ & a_2 & & & \\ & & \ddots & & \\ & & & a_{n-1} & \\ b_1 & b_2 & \cdots & b_{n-1} & a_n \end{bmatrix},$$

dvs. L er en nedre trekantsmatrix, hvor kun diagonale elementer og elementerne i sidste række kan være forskellige fra nul.

(a) Implementér en funktion i C, som udfører operationen $x \leftarrow Lx$, dvs. funktionen skal overskrive input-arrayet x med Lx.

Krav

- Din funktion skal have følgende prototype:

```
int darmv(int n, double * a, double * b, double * x);
```

  Inputtet `n` er ordnen af matricen L. Inputtene `a`, `b` og `x` er pegere til de første elementer af de respektive arrays som repræsenterer vektorerne $a = (a_1, \ldots, a_n)$, $b = (b_1, \ldots, b_{n-1})$ og $x = (x_1, \ldots, x_n)$.
- Funktionen skal returnere værdien -1, hvis et eller flere input er ugyldige, og ellers skal funktionen returnere værdien 0.
- Funktionen må ikke skrive til `stdout`.
- Benyt skabelonen `exam_e20_question18a.c` til din implementering. Skabelonen er inkluderet i ZIP-filen, som er vedhæftet eksamensopgaven.

(b) Implementér en funktion i C, som udfører operationen $x \leftarrow L^{-1}x$, dvs. funktionen skal overskrive input-arrayet x med $L^{-1}x$.

Krav

- Din funktion skal have følgende prototype:

```
int darsv(int n, double * a, double * b, double * x);
```

  Inputtet `n` er ordnen af matricen L. Inputtene `a`, `b` og `x` er pegere til de første elementer af de respektive arrays som repræsenterer vektorerne $a = (a_1, \ldots, a_n)$, $b = (b_1, \ldots, b_{n-1})$ og $x = (x_1, \ldots, x_n)$.
- Funktionen skal returnere værdien -1, hvis et eller flere input er ugyldige eller hvis L er singulær, og ellers skal funktionen returnere værdien 0.
- Funktionen må ikke skrive til `stdout`.
- Benyt skabelonen `exam_e20_question18b.c` til din implementering. Skabelonen er inkluderet i ZIP-filen, som er vedhæftet eksamensopgaven.