NTNU

Kunnskap for en bedre verden

Institutt for datateknologi og informatikk

Eksamensoppgave i Algoritmer og datastrukturer, IDATT2101

Eksamensdato: 15. desember 2023

Eksamenstid (fra–til): 09:00–13:00

Tillatte hjelpemiddel: ett A4-ark med notater

Faglig kontakt under eksamen: Helge Hafting

Tlf.: 924 386 56

Annen informasjon:

Målform/språk: bokmål

Antall sider (uten forside): 4

Antall sider vedlegg: 0

Informasjon om trykking av eksamensoppgave			
Originalen er:			
1-sidig	☐	**2-sidig**	☒
sort/hvit	☐	**farger**	☒
		Flervalgskjema?	☐

Kontrollert av

....

Dato Sign

Merk! Studentene finner sensur i Studentweb. Har du spørsmål om sensuren må du kontakte instituttet ditt. Eksamenskontoret vil ikke kunne svare på slike spørsmål.

Oppgave 1 20%

Analyser disse programmene. Bruk Θ om mulig. Om ikke, bruk O og Ω. Alle parametre er positive.

```
int prog_a(int a, int b, int c) {
    int sum = 0;
    for (int i=1; i<a ; ++i) {
       for (int j=1; j<c; ++j) {
          for (int k=a; k>0; --k) {
            sum += i*j-k;
            if (sum > b) return;
          }
       }
    }
    return sum;
}
```

```
int prog_b(int q, int r, int p) {
    int sum = 0;
    for (int i=0; i<q ; i += 1) {
       sum += i*r;
       if (sum > p) sum -= p;
    }
}
```

```
int prog_c(int n, int [] tab) {
    int sum = 0;
    if (n > 0) {
       sum += 4 * prog_c(n/2, tab);
       for (int i=0; i<n; ++i) sum += tab[i];
       tab[0]--;
       sum += 4 * prog_c(n/2, tab);
    }
    return sum;
}
```

```
double prog_d(int n, float x) {
    if (n == 0) return 0.0;
    else return x + prog_d(n - 1, x);
}
```

```
int prog_e(int a, int b, int c) {

    int sum = 1;
    if (a < b) {
       for (int i = 1; i < a; ++i) sum *= i;
       for (int j = a; j < b; ++j) sum += j;
    }
    return sum;

}
```

Oppgave 2 — 20%

a) Jeg bruker en heap som prioritetskø. Den inneholder n elementer. Hva blir kjøretidene for disse tre operasjonene: å sette inn et element til, å ta ut elementet på toppen, å endre prioritet for et element.

b) Dijkstras algoritme trenger en prioritetskø. Algoritmen brukes på en graf med N noder og K kanter. Vi kan velge en heap som prioritetskø, eller bruke en usortert tabell.

Vil et av valgene alltid være best? Eller vil dette avhenge av grafen på noe vis? Forklar, bruk gjerne kjøretider som funksjon av N og K.

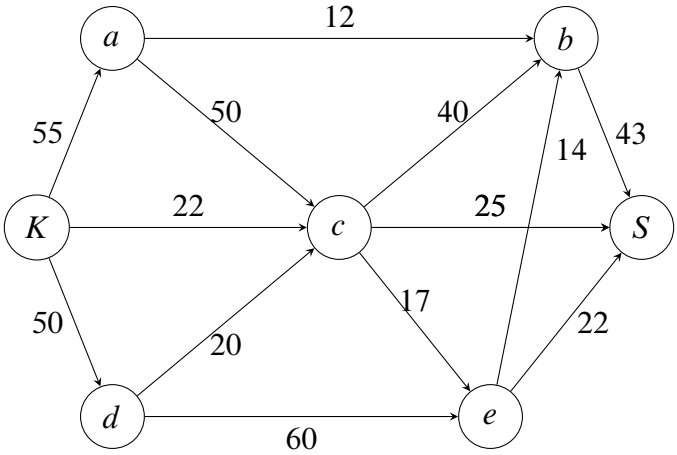
Oppgave 3 — 15%

a) Velg enten Lempel-Ziv eller Lempel-Ziv-Welsh, og forklar kort hvordan algoritmen komprimerer data.

b) A*-algoritmen er en videreutvikling av Dijkstras algoritme. Hva er det A* gjør anderledes? Forklar.

Oppgave 4 30%

Bruk denne grafen.



a) Finn maksimal flyt fra K til S. Nytt flytøkende veier, og skriv opp hver vei og hvor mye flyt du legger til langs veien.

b) Sorter grafen topologisk, eller forklar hvorfor det ikke er mulig.

c) Se bort fra retningen på kantene, og finn et minimalt spenntre for grafen. Skriv opp hvilke kanter som blir med, og total vekt på spenntreet.

d) Finn et annet minimalt spenntre i denne grafen, eller forklar hvorfor det ikke er mulig.

Oppgave 5 — 15%

a) Forklar kort hvordan quicksort virker.

b) Fortell om hvordan quicksort og innsettingssort kan kombineres til en bedre/raskere sorteringsalgoritme. Hvorfor blir den kombinerte algoritmen raskere?

c) Tellesortering egner seg ikke for desimaltall (float/double). Hvorfor ikke?