Question 1: Grådige algoritmer

Hvilke(n) påstand(er) er korrekt(e)?

- [x] Grådige algoritmer tar globalt optimale valg.
- [x] Grådige algoritmer trenger ikke å vite løsningen på alle mulige delproblemer før den kan gjøre det grådige valget.
- [ ] Dersom man kan løse et problem med dynamisk programmering kan man også løse det med en grådig algoritme.
- [ ] Grådige algoritmer pleier å ombestemme seg senere når de har funnet ut mer om løsningene på delproblemene.
- [ ] Grådige algoritmer finner alltid den globalt optimale løsningen.
- [x] Grådige algoritmer brukes til å løse optimaliseringsproblemer.

Question 2: Grådige algoritmer

Hvilke to egenskaper må et problem ha for at vi kan bruke en grådig algoritme?

- ○ Syklisk og optimal substruktur
- ○ Polynomisk kjøretid og problemet lar seg redusere til bare ett delproblem
- ◉ Grådighetsegenskapen og optimal substruktur
- ○ Ingen av alternativene er korrekt
- ○ Grådighetsegenskapen og polynomisk kjøretid

Question 3: Grådige algoritmer

Hvorfor kan det være ønskelig å bruke en grådig algoritme istedenfor dynamisk programmering?

- ○ For å utnytte overlappende delproblemet på en bedre måte
- ○ Fordi vi ønsker å løse problemet rekursivt - noe vi ikke får til med dynamisk programmering.
- ◉ Algoritmen kan være enklere å implementere og ha bedre kjøretid

Question 4: Grådige algoritmer

Hva har grådige algoritmer og dynamisk programmering til felles?

- ( ) Begge løser problemer nedenfra og opp
- ( ) Begge garanterer O(n) kjøretid
- (•) Begge utnytter optimal delstruktur
- ( ) Begge utnytter overlappende delproblemer

Question 5: Grådige algoritmer

Hvilke(n) påstand(er) er korrekt(e)?

- [ ] En grådig algoritme kan enkelt løse både 0-1 og fractional knapsack-problemet
- [ ] En grådig algoritme kan ikke løse fractional knapsack-problemet
- [x] En grådig algoritme kan ikke løse 0-1 knapsack-problemet

Question 6: Aktivitetsutvalg 1

Du ønsker å velge ut så mange aktiviteter som mulig fra en mengde av åtte aktivitet uten at de overlapper.

Information

Author(s)	Marius
Deadline	12/10/2018 16:00:00
Status	Succeeded
Grade	93.33%
Grading weight	1.0
Attempts	2
Submission limit	2 submissions

Submitting as

Sander Lindberg

Classroom : Default classroom

For evaluation

Best submission

08/10/2018 20:47:31 - 93.33%

Submission history

08/10/2018 20:47:31 - 93.33%

08/10/2018 20:43:01 - 86.67%

Question 4: Grådige algoritmer

Hva har grådige algoritmer og dynamisk programmering til felles?

- ○ Begge løser problemer nedenfra og opp
- ○ Begge garanterer O(n) kjøretid
- ◉ Begge utnytter optimal delstruktur
- ○ Begge utnytter overlappende delproblemer

Question 5: Grådige algoritmer

Hvilke(n) påstand(er) er korrekt(e)?

- ☐ En grådig algoritme kan enkelt løse både 0-1 og fractional knapsack-problemet
- ☐ En grådig algoritme kan ikke løse fractional knapsack-problemet
- ☑ En grådig algoritme kan ikke løse 0-1 knapsack-problemet

Question 6: Aktivitetsutvalg 1

Du ønsker å velge ut så mange aktiviteter som mulig fra en mengde av åtte aktivitet uten at de overlapper. Aktivitetene har følgende start og sluttidspunkter.

TASK	START	FINISH
1	12	14
2	12	17
3	6	10
4	15	18
5	16	17
6	0	5
7	4	7
8	6	9

Gitt at du hadde brukt RECURSIVE-ACTIVITY-SELECTOR (side 419) til å løse problemet. Hvilken aktivitet ville vært den 2. i løsningsmengden *A*?

- 1
- 4
- 5
- 7
- 6
- 3
- 8
- 2

Question 7: Aktivitetsutvalg 2

(Bruk tabellen i oppgave 'Aktivitetsutvalg 1')

Gitt at du hadde brukt GREEDY-ACTIVITY-SELECTOR (side 421) på tabellen. Hvilken aktivitet ville vært den 3. aktiviteten i løsningsmengden *A*?

MERK For at algoritmen skal fungere vil du måtte omorganisere elementene i tabellen slik at antagelsen til GREEDY-ACTIVITY-SELECTOR er oppfylt.

- 5
- 7
- 1
- 3
- 6
- 8
- 2
- 4

Question 8: Aktivitetsutvalg 3

(Bruk tabellen fra oppgave 'Aktivitetsutvalg 1')

Gitt at du hadde brukt RECURSIVE-ACTIVITY-SELECTOR (side 419) på tabellen. Hva blir løsningen (aktiviteter i kronologisk rekkefølge)?

- ○ 6, 3, 2, 5
- ◉ 6, 8, 1, 5
- ○ 6, 7, 1, 5
- ○ 6, 7, 3, 2

Question 9: Aktivitetsutvalg 4

Hva forteller teorem 16.1 i boka om aktivitetsutvalg-problemet?

- ○ Det har optimal substruktur
- ○ Det lar seg ikke løse
- ◉ At det har grådighetsegenskapen
- ○ Det har overlappende delproblemer

Question 10: Huffman-koder 1

Du ønsker å finne optimal prefix-kode for en streng. Strengens alfabet representeres ved bokstavene *a* til *g*. Frekvensene er som følger:

BOKSTAV	FREKVENS
a	50
b	2
c	20
d	25
e	200
f	80
g	60

Gitt at vi velger å kode alfabetet på følgende måte:

- *a*: 00001
- *b*: 001
- *c*: 1
- *d*: 00000
- *e*: 0001
- *f*: 010
- *g*: 011

Hvor mange bits må vi bruke for å representere strengen?

- (•) 1621
- ( ) 1537
- ( ) 1689
- ( ) 1546

Question 11: Huffman-koder 2

(Bruk tabellen fra oppgave Huffman-koder 1)

Du bruker Huffmans algoritme. Hvilke to bokstaver slår du sammen først?

- ○ *a* og *e*
- ○ *c* og *d*
- ○ *e* og *f*
- ◉ *b* og *c*
- ○ *a* og *b*

Question 12: Huffman-koder 3

(Bruk tabellen fra oppgave Huffman-koder 1)

Du bruker Huffmans algoritme. Hvor mange bits blir *b* kodet til?

- ○ 2
- ◉ 5
- ○ 1
- ○ 4
- ○ 6
- ○ 3

Question 13: Huffman-koder 4

(Bruk tabellen fra oppgave Huffman-koder 1)

Du bruker Huffmans algoritme. Hvor mange bits blir *d* kodet til?

- ○ 6
- ○ 1
- ○ 3
- ○ 5
- ◉ 4
- ○ 2

Question 14: Huffman-koder 5

(Bruk tabellen fra oppgaven Huffman-koder 1)

Du bruker Huffmans algoritme. Hvor mange bits blir *e* kodet til?

- ○ 5
- ○ 4
- ○ 6
- ◉ 1
- ○ 3
- ○ 2

Question 15: Huffman-koder 6

(Bruk tabellen fra oppgave Huffman-koder 1)

Du bruker Huffmans algoritme. Hvor mange bits trenger du for å kode strengen med løsningen du finner?

- ○ 1023
- ○ 734
- ○ 561
- ○ 452
- ○ 450
- ○ 958
- ◉ 980
- ○ 603
- ○ 789