Algorithms & Data Structures 2 (1DL231) **Exam of 15 January 2020**

Instructions: This is a multiple-choice exam, in order to save you the time of tidying up the presentation of your answers. There is exactly ***one*** correct answer per ***assessed*** question. You can keep this questionnaire and should ***hand in only the answer sheet***: you are ***not*** expected to explain your answers. Unfortunately, the head teacher cannot attend this exam. ***Also read the instructions on the answer sheet before starting***.

0 Information About You

Question -1: (not assessed) What is your study programme?

A DVK B IT C STS D master E exchange

Question 0: (not assessed) How many of the 13 lectures did you attend?

A $0..1$ B $2..4$ C $5..7$ D $8..10$ E $11..13$

1 String Matching

Question 1: On which of the following length-m patterns P does the naïve string matching algorithm reach its ***worst-case*** runtime when looking for ***all*** occurrences of P in the text $T = 0^n$ (that is, a string of n occurrences of the character '0'), with $n \geq m \geq 3$?

A 0^m B $0(1^{m-1})$ C $1(0^{m-1})$ D $1^{m-1}0$ E 1^m

Question 2: For the Rabin-Karp string matching algorithm, let p denote the fingerprint of the length-m pattern P, and let t_s denote the fingerprint of the length-m substring T_s for shift s in text T (of length at least m). On which assumption does the algorithm rely?

A $p = t_s \Leftarrow \forall k \in 1..m : P[k] = T_s[k]$

B $p = t_s \Leftrightarrow \forall k \in 1..m : P[k] = T_s[k]$

C $p = t_s \Rightarrow \forall k \in 1..m : P[k] = T_s[k]$

D $p \neq t_s \Leftarrow \exists k \in 1..m : P[k] \neq T_s[k]$

E $p \neq t_s \Rightarrow \forall k \in 1..m : P[k] \neq T_s[k]$

Question 3: How many ***spurious*** hits does the Rabin-Karp string matching algorithm encounter in the text $T =$ "3141512659849792" when looking for ***all*** occurrences of the pattern $P =$ "26", working modulo $q = 11$ and over the alphabet $\Sigma = \{$'0', '1', '2', ..., '9'$\}$?

A 0 B 1 C 2 D 3 E 4

Question 4: On which of the following patterns P does the Rabin-Karp string matching algorithm reach its ***worst-case*** runtime when looking for ***all*** occurrences of P in the text $T = 0^n$ (that is, a string of n occurrences of the character '0'), with $n \geq 3$, working modulo $q = 3$ and over the alphabet $\Sigma = \{$'0', '1', '2', ..., '9'$\}$?

A "660" B "300" C "099" D "007" E "006"

2 Greedy Algorithms

Consider n lectures $\ell_1, \ell_2, \ldots, \ell_n$. Each lecture ℓ_i has a start time s_i and a finish time f_i, where $0 \le s_i < f_i < \infty$, and happens during the half-open time interval $[s_i, f_i)$. We wish to find the minimum number of classrooms to schedule all lectures so that no two lectures overlap in time in the same room. For ***example***, consider the following set of lectures:

i	1	2	3	4	5	6	7	8	9	10
s_i	1	1	1	5	5	9	9	11	13	13
f_i	4	8	4	8	11	12	12	16	16	16

The partition $\{\{\ell_1, \ell_6, \ell_9\}, \{\ell_2, \ell_8\}, \{\ell_3, \ell_4, \ell_7\}, \{\ell_5, \ell_{10}\}\}$ uses 4 classrooms, but the partition $\{\{\ell_1, \ell_5, \ell_8\}, \{\ell_2, \ell_7, \ell_9\}, \{\ell_3, \ell_4, \ell_6, \ell_{10}\}\}$ uses only 3 classrooms. The latter partition uses the minimum number of classrooms; other such partitions swap its symmetric lectures ℓ_1 and ℓ_3, or ℓ_6 and ℓ_7, or ℓ_9 and ℓ_{10}. Consider the following greedy-algorithm template:

Greedy-Lecture-Partition($n, [s_1, \ldots, s_n], [f_1, \ldots, f_n]$)

1 in-place sort the lectures by some criterion
2 $c := 0$ // c is the current number of allocated classrooms
3 **for** $i := 1$ **to** n
4 **if** ℓ_i does not overlap with any lecture in some already allocated room, say $k \in 1\,..\,c$
5 schedule lecture ℓ_i in classroom k
6 **else**
7 $c := c + 1$; schedule lecture ℓ_i in classroom c
8 **return** the schedule

Question 5: Assume we in-place sort in line 1 the lectures by monotonically increasing ***start time***, giving $s_1 \le s_2 \le \cdots \le s_n$: on which inputs (such as the example above) does the greedy algorithm above return a minimum partition?

☐ A none ☐ B some, but not all ☐ C all

Question 6: Assume we in-place sort in line 1 the lectures by monotonically increasing ***finish time***, giving $f_1 \le f_2 \le \cdots \le f_n$: on which inputs (such as the example above) does the greedy algorithm above return a minimum partition?

☐ A none ☐ B some, but not all ☐ C all

Question 7: Assume we in-place sort in line 1 the lectures by monotonically increasing ***duration***, giving $f_1 - s_1 \le f_2 - s_2 \le \cdots \le f_n - s_n$: on which inputs (such as the example above) does the greedy algorithm above return a minimum partition?

☐ A none ☐ B some, but not all ☐ C all

Question 8: Assume it is lecture ℓ_j that finishes last in a classroom k considered in the loop over $1\,..\,c$ in line 4 of the algorithm in ***Question 6***: how can one test the condition in line 4 in ***constant*** time for that classroom k?

☐ A $f_j < s_i$ ☐ B $f_i < s_j$ ☐ C $f_j \le s_i$ ☐ D $f_i \le s_j$ ☐ E impossible

Question 9: What is the ***tightest*** time complexity in which the **for** loop in lines 3 to 7 (without the sorting) of the greedy algorithm in ***Question 5*** can be implemented to run?

☐ A $\mathcal{O}(n^3)$ ☐ B $\mathcal{O}(n^2 \cdot \lg n)$ ☐ C $\mathcal{O}(n^2)$ ☐ D $\mathcal{O}(n \cdot \lg n)$ ☐ E $\mathcal{O}(n)$

3 Dynamic Programming

Consider the problem of finding the *length of a longest common subsequence* (LLCS) of two given sequences $\langle x_1, \ldots, x_m \rangle$ and $\langle y_1, \ldots, y_n \rangle$, whose lengths m and n may differ. For ***example***, the sequences $X = \langle A, B, C, B, D, A, B \rangle$ and $Y = \langle B, D, C, A, B, A \rangle$ have common subsequences of length 4, such as $\langle B, C, B, A \rangle$ and $\langle B, D, A, B \rangle$, but no longer ones. Define the kth *prefix* of a sequence $Z = \langle z_1, \ldots, z_\ell \rangle$ as $Z_k = \langle z_1, \ldots, z_k \rangle$, for $k \in 0\,..\,\ell$. In our example, X_3 is $\langle A, B, C \rangle$ and X_0 is the empty sequence. Consider the following recurrence — with placeholders α_1, α_2, β_1, and β_2 — on a numeric quantity $L(i, j)$:

$$L(i,j) = \begin{cases} 0 & \text{if } \neg\beta_1 \\ L(\alpha_1, \alpha_2) + 1 & \text{if } \beta_1 \text{ and } \beta_2 \\ \max\{L(\alpha_1, j),\ L(i, \alpha_2)\} & \text{otherwise} \end{cases}$$

Question 10: If $L(m, n)$ is returned by a correct algorithm for computing the LLCS of two given sequences $X = \langle x_1, \ldots, x_m \rangle$ and $Y = \langle y_1, \ldots, y_n \rangle$, then what is $L(i, j)$, for $i \in 0\,..\,m$ and $j \in 0\,..\,n$?

- [A] the LLCS of X and Y
- [B] the LLCS of $\langle x_0, \ldots, x_i \rangle$ and $\langle y_0, \ldots, y_j \rangle$
- [C] the LLCS of X_{i-1} and Y_{j-1}
- [D] the LLCS of X_{i+1} and Y_{j+1}
- [E] the LLCS of X_i and Y_j

Hint: For the remaining questions, think about how $L(2, 2)$ and $L(3, 3)$ can be computed.

Question 11: What is the Boolean placeholder β_1?

[A] $i \in 1\,..\,m$ [B] $i > m$ [C] $i + j > 0$ [D] $i \cdot j > 0$ [E] $i \leq j$

Question 12: What is the Boolean placeholder β_2?

[A] $x_i \neq y_j$ [B] $x_i < y_j$ [C] $x_i \leq y_j$ [D] $x_i \geq y_j$ [E] $x_i = y_j$

Question 13: What is the index placeholder α_1?

[A] $i - 1$ [B] i [C] $i + 1$ [D] $j - 1$ [E] j

Question 14: What is the index placeholder α_2?

[A] $i - 1$ [B] i [C] $j - 1$ [D] j [E] $j + 1$

Question 15: What order of computing the $L(i, j)$ by a bottom-up method only refers to already computed values?

- [A] **for** $j = 0$ **to** n **for** $i = m$ **downto** 0
- [B] **for** $j = 0$ **to** n **for** $i = 0$ **to** m
- [C] **for** $j = n$ **downto** 0 **for** $i = 0$ **to** m
- [D] **for** $i = m$ **downto** 0 **for** $j = 0$ **to** n
- [E] **for** $i = 0$ **to** m **for** $j = n$ **downto** 0

4 Complexity

Question 16: If the best-known solution checker for a decision problem D takes $\mathcal{O}(k^n)$ time on an instance of size n, for a constant $k > 1$, then what is the ***tightest*** time complexity class of D, according to this knowledge?

☐ A P ☐ B NP ☐ C NP-complete ☐ D NP-hard ☐ E none of the others

Question 17: Consider the following decision version of the LLCS problem of Section 3: is the LLCS of two given sequences $X = \langle x_1, \ldots, x_m \rangle$ and $Y = \langle y_1, \ldots, y_n \rangle$ at least a given natural number ℓ? Assuming $m \geq n \geq \ell$, a brute-force decision algorithm tests in $\mathcal{O}(m \cdot 2^n)$ time every subsequence of Y whether it has at least ℓ characters and is a subsequence of X. What is the ***tightest*** time complexity class of this ***problem***, according to current knowledge?

☐ A P ☐ B NP ☐ C NP-complete ☐ D NP-hard ☐ E none of the others

Question 18: There exists an algorithm that generates the list $[0, 1, 2, \ldots, n-1]$ for a given natural number n in $\Theta(n)$ time: what is the most accurate description of this time complexity?

☐ A logarithmic ☐ B linear ☐ C pseudo-polynomial ☐ D super-exponential ☐ E none of the others

Question 19: There exists an algorithm that computes the power a^n of a given number a for a given natural number n in $\Theta(\log_2 n)$ time: what is the most accurate description of this time complexity?

☐ A logarithmic ☐ B linear ☐ C pseudo-polynomial ☐ D super-exponential ☐ E none of the others

Question 20: In order to prove that a decision problem D is NP-complete, one can:

☐ A prove that D reduces to (often denoted by $\leq_{\mathrm{P}}$) some known problem in P

☐ B prove that D reduces to some known NP-complete problem

☐ C prove that D reduces to some known NP-complete problem and that D is in NP

☐ D prove that some known NP-complete problem reduces to D and that D is in NP

☐ E prove that some known NP-complete problem reduces to D

Answer Sheet — AD2 Exam (1DL231) of 15 January 2020

Instructions: Do ***not*** alter the drawing above. Using a ***very dark*** colour, ***fill in*** entirely (only like this: ■) ***at most one*** answer box (A to E) per question: we will use an optical character recognition (OCR) system that ignores circles, crosses, ticks, etc. Transfer your answers from the questionnaire to this answer sheet ***just before handing in***; if an answer becomes ambiguous to an OCR system, then request another answer sheet. Every correct answer to an ***assessed*** question gives 1.5 points. Every multiple answer or incorrect answer gives 0 points. Partial credit of 0.5 points or 1 point may be given in exceptional circumstances. If you think a question is unclear or faulty, then mark its number with a $\star$ on ***this sheet*** and explain ***on the backside of this sheet*** what your difficulty with the question is ***and*** what additional assumption underlies the candidate answer that you have chosen or the new answer that you indicate.

Grading: Your grade is as follows, when your mark is e points:

Grade	Condition
5	$25.0 \leq e \leq 30.0$
4	$20.0 \leq e \leq 24.5$
3	$15.0 \leq e \leq 19.5$
U	$00.0 \leq e \leq 14.5$

0 Information About You

Question -1: ■ ■ ■ ■ ■

Question 0: ■ ■ ■ ■ ■

1 String Matching

Question 1: ■ B C D E

Question 2: ■ B C D E

Question 3: A B C ■ E

Question 4: A B C D ■

2 Greedy Algorithms

Question 5: A B ■

Question 6: A ■ C

Question 7: A ■ C

Question 8: A B ■ D E

Question 9: A B C ■ E

3 Dynamic Programming

Question 10: A B C D ■

Question 11: A B C ■ E

Question 12: A B C D ■

Question 13: ■ B C D E

Question 14: A B ■ D E

Question 15: A ■ C D E

4 Complexity

Question 16: A B C D ■

Question 17: ■ B C D E

Question 18: A B ■ D E

Question 19: A ■ C D E

Question 20: A B C ■ E

Again: Use a ***very dark*** colour to ***fill in*** your chosen boxes ***entirely*** (like this: ■)!

Your anonymous exam code: | | | | | | | | |