Technical University of Denmark

Written exam, December 9, 2016.

Course name: Algorithms and data structures II.

Course number: 02110.

Aids allowed: All written materials are permitted.

Exam duration: 4 hours

Weighting: Question 1: 14% - Question 2: 16% - Question 3: 15% - Question 4: 20% - Question 5: 15% - Question 6: 20%.

The weighting is only an approximative weighting.

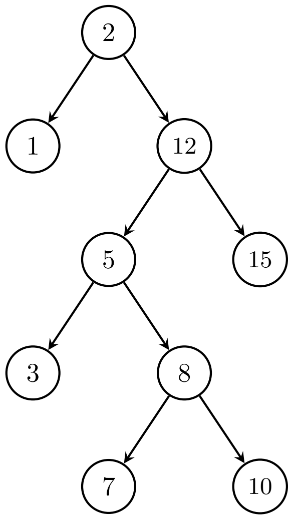
You can answer the exam in either Danish or English.

All questions should be answered by filling out the blank space below the question. As exam paper just hand in this and the following pages filled out. If you need more space you can use extra paper that you hand in together with the exam paper.

The exam set continues on the next page

Question 1

Question 1.1 (8%) Show how the splay tree below looks after inserting the element 6.



1.2 (6%) A sequence of n operations is performed on a data structure. The ith operation costs i if i is an exact power of 2, and 1 otherwise.

The worst case cost of an operation is:

$\boxed{\text{A}}$ $\Theta(1)$ $\boxed{\text{B}}$ $\Theta(\log n)$ $\boxed{\text{C}}$ $\Theta(n)$ $\boxed{\text{D}}$ $\Theta(2^n)$

The amortized cost of an operation is:

$\boxed{\text{A}}$ $\Theta(1)$ $\boxed{\text{B}}$ $\Theta(\log n)$ $\boxed{\text{C}}$ $\Theta(n)$ $\boxed{\text{D}}$ $\Theta(2^n)$

The exam set continues on the next page

Question 2

Question 2.1 (8%) Draw the suffix tree for the string `ABABBAA$` (don't replace the labels by indexes into the string, just write the labels in the vertices):

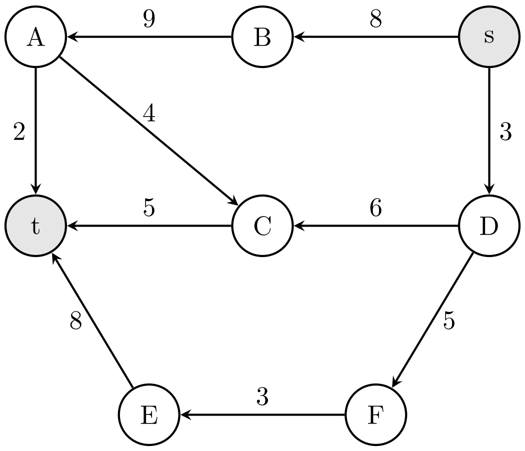
Question 2.2 (8%) Compute the prefix function as used in the Knuth-Morris-Pratt algorithm for the string `AABCAAAB`:

i	1	2	3	4	5	6	7	8
$P[i]$	A	A	B	C	A	A	A	B
$\pi[i]$								

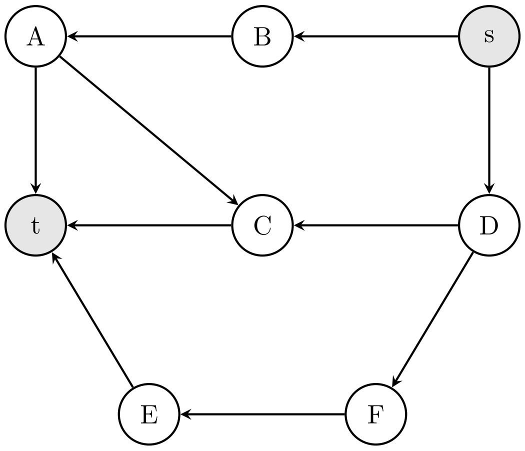
The exam set continues on the next page

Question 3

Consider the network N below with capacities on the edges.



Question 3.1 (8%) Give a maximum flow from s to t in the network N (write the flow for each edge along the edges on the graph below), give the value of the flow, and give a minimum $s-t$ cut (give the partition of the vertices).



value of flow: ____________________

minimum cut: ____________________

Question 3.2 (7%) Use Edmonds-Karp's algorithm to compute a maximum flow in the network N. For each augmenting path write the nodes on the path and the value you augment the path with in the table below.

augmenting path	value

Question 4

You live in a cooperative apartment with n people. The co-op needs to schedule cooks for the next n days, so that each person cooks 3 days and each day there are 3 cooks. In addition, each member of the co-op has a list of days they are available to cook (and are unavailable to cook on the other days). You are asked to come up with a schedule for cooking, so that everyone cooks on days they are available.

Question 4.1 (10%) Describe an algorithm that given the lists $L_1, L_2, \ldots, L_n$ of available days for each of the n people in the co-op returns a schedule, so that there are 3 people cooking every day and each person cooks on 3 days from their list. Or report (correctly) that no such schedule is possible.

Analyze the asymptotic running time of your algorithm. Remember to argue that your algorithm is correct.

Question 4.2 (10%) People tend to pick too few dates, e.g., people tend to say they can't cook on Friday. So it often happens that it is not possible to construct a feasible schedule. The co-op decides that people can get assigned to days not on their list, but only to at most 1 day not on their list. Modify your algorithm so that every person can get assigned to 3 days in total, and at most one of the days is not on their list.

Analyze the asymptotic running time of your algorithm. Remember to argue that your algorithm is correct.

Solution to this question requires knowledge of suffix trees, which is not part of the curriculum in 2023.

Question 5

Let S be a string of length n. A *repeat* of S is a string occurring at least two times in S. A *right-maximal repeat* is a repeat that cannot be extended to the right to a longer repeat, by adding characters to the right.

For example are `A` and `AB` repeats of the string $S =$ `A B C A C C A B C C A`, but not right-maximal repeats since `ABC` is also a repeat of S. The strings `ABC`, `BC`, `CA`, and `CCA` are all right-maximal repeats of S.

Given a string S of length n we want to find all the right-maximal repeats of S that are of length at least ℓ.

Question 5.1 (5%) Find all right-maximal repeats of length at least 3 in $T =$ `C C A B A B C C A B C C A`.

Question 5.2 (10%) Describe an algorithm that given a string S of length n returns all the right-maximal repeats of S that are of length at least ℓ.

The string is over an alphabet of size $O(1)$ and you can assume that the suffix tree of a string of length n over an alphabet of constant size can be constructed in $O(n)$ time.

Analyze the asymptotic running time of your algorithm. Remember to argue that your algorithm is correct.

The exam set continues on the next page

Question 6

Tomorrow is the big dance contest at DTU. You have a list of n songs that will be played during the contest, in chronological order. You know all the songs, all the judges, and your own dancing ability extremely well. For each integer k, you know that if you dance to the kth song on the schedule, you will be awarded exactly $\texttt{Score}[k]$ points, but some of the songs are very physically demanding. Thus if you want to dance to song k you will have to take a break for the $\texttt{Break}[k]$ songs before. E.g., if $\texttt{Break}[10] = 3$ and you choose to dance to song 10, then you cannot dance to song 7,8, and 9. You can assume that $\texttt{Break}[k] < k$.

Let $P(1, i)$ denote the maximal score you can have after the first i songs if you dance to song i. Similarly, let $P(0, i)$ denote the maximal score you can have after the first i songs if you do not dance to song i.

Question 6.1 (5%) Fill out the table below when $\texttt{Score} = [3, 4, 8, 1, 2, 1]$ and $\texttt{Break} = [0, 1, 2, 3, 1, 2]$.

$P(d,i)$	1	2	3	4	5	6
0						
1						

Question 6.2 (5%) Which of the following recurrences correctly computes $P(d, i)$:

$\boxed{\text{A}}$
$$P(d,i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\{P(1, i-1), P(0, i-1)\} & \text{if } d = 0 \text{ and } i \geq 1 \\ P(1, i - \texttt{Break}[i]) + \texttt{Score}[i] & \text{if } d = 1 \text{ and } i \geq 1 \end{cases}$$

$\boxed{\text{B}}$
$$P(d,i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\{P(1, i-1), P(0, i-1)\} & \text{if } d = 0 \text{ and } i \geq 1 \\ P(0, i - \texttt{Break}[i]) + \texttt{Score}[i] & \text{if } d = 1 \text{ and } i \geq 1 \end{cases}$$

$\boxed{\text{C}}$
$$P(d,i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\{P(1, i-1), P(0, i-1)\} & \text{if } d = 0 \text{ and } i \geq 1 \\ \max\{P(0, i - \texttt{Break}[i]), P(1, i - \texttt{Break}[i])\} + \texttt{Score}[i] & \text{if } d = 1 \text{ and } i \geq 1 \end{cases}$$

Question 6.3 (10%) Write pseudocode for an algorithm *based on dynamic programming and the recurrence from Question 6.2* that computes the maximum total score you can achieve. The input to your algorithm is the pair of arrays `Score`$[1 \ldots n]$ and `Break`$[1 \ldots n]$. Analyze the space usage and running time of your algorithm in terms of n.