
Example Final Exam, 2-AIN-205

Name and Surname: .....

- This exam has 8 pages. Write your solutions directly in the exam book. Write legibly, if we cannot read it you won't get points. If you need more space, use empty pages, but **clearly state that solution continues on a different page.**
- There is a bonus task which will be marked more strictly. In particular, in most tasks you get partial marks for partial solutions while this is unlikely in case of the bonus taks. We recommend that you attempt bonus tasks only at the end if you have spare time.
- If you think that the problem statement is ambiguous, document additional necessary assumptions and continue as if these assumptions were part of the problem statement.
- Do not attempt to cheat on the exam. Any violation of academic integrity will be submitted to the disciplinary committee.

1 (*25 bodov*) **Short questions**

Answer questions below and briefly justify your answer (typically one or two sentences are enough).

a) Assume that the expected running time of a Las Vegas algorithm is $O(n^2)$. Is there a possibility that sometimes this algorithm will run in an exponential (such as $\Omega(2^n)$) running time?

b) Consider a greedy algorithm for solving the knapsack problem that takes items in order from the largest price pre unit of weight to the smallest. What is the smallest number n such that for any number $d \geq 2$ we can find an input with n items where the solution created by this greedy algorithm is d times worse than the optimal solution?

c) Consider an algorithm that tests whether number n is a prime by trying whether it can be divided by numbers 2, 3, 5, 7, and 11. If n is not divisible by any of those, the algorithm claims that n is the prime number. Can this algorithm be considered a Monte Carlo algorithm?

d) Consider a random number generator that outputs zero with probability 70% and one with probability 30%. We generate random numbers until we see 99 ones. What is the expected number of the calls of the generator?

e) Let A be a 6-approximation algorithm for solving problem P and B be a 3-approximation algorithm for the same problem. Is it correct to say that algorithm B will always output a better solution than algorithm A?

2 (*25 bodov*) **Big data problem**

A hard drive stores n pages of data numbered $1 \dots n$. The internal memory of the computer can only hold $k < n$ pages of data. During the execution of a program, we sequentially request access to individual pages of data. If the requested page is already in the internal memory, no additional work is necessary. However, if the page is not present in the memory, so called *pagefault* occurs, and to continue the program, we need to load the page into the internal memory. In case there is no more space in the internal memory, one of the pages previously loaded must be evicted from the internal memory.

For example, let $k = 3$ and let the program request pages in the sequence $1, 2, 3, 2, 4, 3, 5, 1, 2, 5$. After the first three steps ($3\times$ pagefault), pages $1, 2, 3$ are stored in the memory. In the fourth step we do not need to do anything as page 2 is already in the memory. In the fifth step, we need to load page 4 (pagefault), however there is no more space. Thus we have to evict e.g. page 2 and load page 4 instead; thus the content of the memory will be $1, 3, 4$. The next pagefault will occur in step 7 at which point page 4 will be replaced with page 5. The last pagefault will occur at step 9 at which point page 3 is replaced with page 2. Altogehter there will be 6 pagefaults and this is the smallest number of pagefaults possible for this sequence.

In practice, we do not know the sequence of requests up front. Consequently, simple heuristics are typically used to determine which page to evict upon pagefault. For example, the *least-recently-used (LRU)* heuristics always evicts a page that was unused for the longest number of steps. In the above example, the LRU heuristics would in step 5 evict page 1 because it was last requested in step 1 (all the other pages were requested more recently).

a) Show that LRU is not the optimal algorithm.

b) Show that if in a subsequence of request there is only k distinct pages requested, the LRU will have at most k pagefaults, regardless of the length of such subsequence.

c) Is it possible that the optimal algorithm in a subsequence of requests containing $k+1$ distinct pages will not have any pagefault? Why?

d) Show that LRU is a k-approximation algorithm.

Hint: The whole sequence of requests can be divided into segments which end just before there would be a $(k+1)$-st distinct page in that particular segment. For example, for $k=3$, the sequence from above can be split into three segments $1,2,3,2|4,3,5|1,2,5$.

3 (*25 bodov*) **Almost minimum**

We are given array A that contains n distinct numbers; the array is already in the memory. Your task is to find a number that is *almost minimum* of array A, which means that only at most 1% of numbers from the array is smaller than the number that you found.

a) If we choose three numbers from array A at random, what is the probability that none of these numbers is almost minimum? Justify your answer.

b) Propose a Monte Carlo algorithm that will find almost minimum with probability of error $\leq 1/4$. Since algorithm with a running time of $O(n)$ is trivial, your algorithm must have a sublinear running time. Prove the probability bound for your algorithm.

c) What is the running time of your algorithm?

4 (*25 bodov*) **From Monte Carlo to Las Vegas**

We want to solve a **decision** problem P. Let n be the size of the input to the problem. We are given two Monte Carlo algorithms that can solve problem P with the following properties:

- **Algorithm A:** "yes" answer is always correct; the probability of an incorrect answer is upper bounded by 1/3. The running time is $3n$.
- **Algorithm B:** "no" answer is always correct; the probability of an incorrect answer is upper bounded by 1/4. The running time is $5n$.

Your task is to propose a Las Vegas algorithm which will solve P.

a) Write a pseudocode of Las Vegas algorithm for solving problem P and prove its correctness.

b) What is the running time of your algorithm? Give as tight bound as possible.
Hint: It is ok to analyze the case when the correct answer is "yes" and the case when the correct answer is "no" separately.

5 (*25 bodov*) **BONUS: Lorem ipsum**

There is usually bonus question that is more difficult and typically looks similar to one of the questions on homework assignments. Since this is a bonus question, it typically requires an almost perfect solution to actually get any points. In particular, a rough idea written in two-three sentences, even if correct, would not qualify for any bonus points.

This page is intentionally left empty. You can use it as an overflow space; if you want anything on this page to be marked please note use of the overflow space next to the corresponding question.