CSC263H1: Problem Set 2

Due Tuesday September 24 before 10pm

General instructions

Please read the following instructions carefully before starting the problem set. They contain important information about general problem set expectations, problem set submission instructions, and reminders of course policies.

- Your problem sets are graded on both correctness and clarity of communication. Solutions that are technically correct but poorly written will not receive full marks. Please read over your solutions carefully before submitting them.
- Each problem set may be completed in groups of up to three. If you are working in a group for this problem set, please consult the articles on collaboration and plagiarism on posted on quercus.
- Solutions must be typeset electronically, and submitted as a PDF with the correct filename. **Hand-written submissions will receive a grade of ZERO.**

  The required filename for this problem set is **problem_set2.pdf**.
- Problem sets must be submitted online through CrowdMark. If you haven't used CrowdMark before, give yourself plenty of time to figure it out, and ask for help if you need it! If you are working with a partner, you must form a group on CrowdMark, and make one submission per group. "I didn't know how to use CrowdMark" is not a valid excuse for submitting late work.
- Your submitted file(s) should not be larger than 9MB. You might exceed this limit if you use a word processor like Microsoft Word to create a PDF; if it does, you should look into PDF compression tools to make your PDF smaller, although please make sure that your PDF is still legible before submitting!
- Submissions must be made *before* the due date on CrowdMark.
- The work you submit must be that of your group; you may not use or copy from the work of other groups, or external sources like websites or textbooks.

Additional instructions

- Lorem ipsum.

1. **[10 marks] Heaps.** Consider a max-priority queue Q implemented using a binary max-heap. We would like to design an ExtractSecondLargest(Q) operation, which returns the second largest key in Q and deletes it from Q. The worst-case running time of this operation must be in $\mathcal{O}(\log n)$. We assume that all keys in Q are distinct integers.

   (a) Write the pseudo-code of your ExtractSecondLargest(Q) algorithm. Let Q be an array whose indices start from 1. You can use operations from the textbook and lectures as helpers without describing their details.

   (b) Explain why your pseudo-code works correctly. In particular, explain why the element you extract is the second largest one, and why this operation maintains the heap property.

   (c) Explain why the worst-case running time of your algorithm is in $\mathcal{O}(\log n)$.

2. **[20 marks] Expected cost.** Consider a min-priority queue Q implemented using a binary min-heap. Let $k \in \mathbb{N}$ be a given natural number. Suppose that Q contains $n = 2^k - 1$ elements (or "nodes"), with indices $1 \ldots n$. Let a_j be the value stored in the element with index 2^{j-1}.

   Let $x \in \mathbb{Z}$ be a given integer. Let m be a random variable corresponding to the number of swaps performed by Q.insert(x).

   Let p be a given real number, $0 < p < 1$. Suppose that $P(a_j \leq x) = p^j$.

   (a) Consider the tree node with index 2^j. What is the index of its parent?

   (b) Use a diagram to illustrate the shape of the binary tree, indicate the number of levels on of the tree, and indicate the position of the indices 2^{j-1}, $1 \leq j \leq k$ on the tree.

   (c) Express $P(x < a_j)$ in terms of p and j.

   (d) Express $P(\, x < a_j \cap x < a_{j+1}\,)$ in terms of p and j. Justify your response.

   (e) Express $P(x < a_{j+1} \,|\, x < a_j)$ in terms of p and j. Justify your response.

   (f) Express $P(x < a_j \,|\, x < a_{j+1})$ in terms of p and j. Justify your response.

   (g) Express $P(a_j \leq x \,|\, x < a_{j+1})$ in terms of p and j. Justify your response.

   (h) Under what condition would Q.insert(x) perform exactly one *swap*? Express this condition in terms of x, a_{k-1}, and a_k.

   (i) What is $P(m = 1)$, i.e., the probablity that Q.insert(x) performs *exactly one* swap operation? Express your answer in terms of k and p. Justify your response.

   (j) Under what condition would Q.insert(x) perform exactly two *swaps*? Express this condition in terms of x, a_{k-2}, and a_{k-1}.

   (k) What is $P(m = 2)$, i.e., the probablity that Q.insert(x) performs *exactly two* swap operations? Express your answer in terms of k and p. Justify your response.

   (l) Under what condition would Q.insert(x) perform exactly j *swaps*? Express this condition in terms of x, a_{k-j}, and a_{k-j+1}. You may assume $0 < j < k$.

   (m) What is $P(m = j)$, i.e., the probablity that Q.insert(x) performs *exactly* j swap operations? Express your answer in terms of j, k and p. Justify your response.

   (n) Express $\langle m \rangle$ (also denoted $E[m]$), i.e., the expected number of *swaps* for Q.insert(x) in terms of k and p. Your answer may be left in summation form and may include dummy (summation index) variables if necessary.