University of Waterloo
CS240 Spring 2022
Assignment 3

Due Date: Wednesday, June 15 at 5:00pm

The integrity of the grade you receive in this course is very important to you and the University of Waterloo. As part of every assessment in this course you must read and sign an Academic Integrity Declaration before you start working on the assessment and submit it **before the deadline of June 15** along with your answers to the assignment; i.e. **read, sign and submit A03-AID.txt now or as soon as possible**. The agreement will indicate what you must do to ensure the integrity of your grade. If you are having difficulties with the assignment, course staff are there to help (provided it isn't last minute).

The Academic Integrity Declaration must be signed and submitted on time or the assessment will not be marked.

Please read `https://student.cs.uwaterloo.ca/~cs240/s22/assignments.phtml#guidelines` for guidelines on submission. **Each question must be submitted individually to MarkUs as a PDF** with the corresponding file names: a3q1.pdf, a3q2.pdf, a3q3.pdf, a3q4.pdf, a3q5.pdf . It is a good idea to submit questions as you go so you aren't trying to create several PDF files at the last minute.

Late Policy: Assignments are due at 5:00pm on Wednesday. Students are allowed to submit **one** late assignment, 2 days after the due date on Friday by 5:00pm. Assignments submitted after Friday at 5:00pm or Wednesday at 5:00pm (if you have already used your one late submission) will not be accepted for grading but may be reviewed (by request) for feedback purposes only. If you want to use your one time late assignment allowance, please send an email to "cs240@uwaterloo.ca" with the title "Using one time late assignment allowance".

Problem 1 [5 marks]

Derive the average-case run time of the algorithm $HalfSort(A)$. Your derivation must be based on the definition of the average-case running time (not on an equivalence of running time to some randomized version of $HalfSort(A)$). You can assume n is divisible by 2 in your analysis. Express running time asymptotically. You can use the fact that $\sum_{n=0}^{\infty} \frac{n}{2^n} = c$ for some constant c.

```
HalfSort(A)
A: an array of size $n \geq 2$ storing distinct numbers
1.  $i \leftarrow 1$
2.  while $i < n$
3.      if $A[i-1] > A[i]$
4.          return false
5.      $i \leftarrow i + 2$
6.  return true
```

Problem 2 [2+2+4=8 marks]

Consider the algorithm below, where *random(k)* returns an integer from the set of $\{0, 1, 2, \ldots, k-1\}$ uniformly and at random.

```
ArrayAlg(A)
A: an array storing numbers
1.  if $A.size == 1$
2.      return
3.  $i \leftarrow random(A.size)$
4.  for $j = 0$ to $i$ do
5.      print($A[j]$)
6.  $ArrayAlg(A[0, ..., A.size - 2])$
```

a) What is the best-case (i.e. the best luck) running time of *ArrayAlg* on array A of size n? Justify.

b) What is the worst-case (i.e. the worst luck) running time of *ArrayAlg* on array A of size n? Justify.

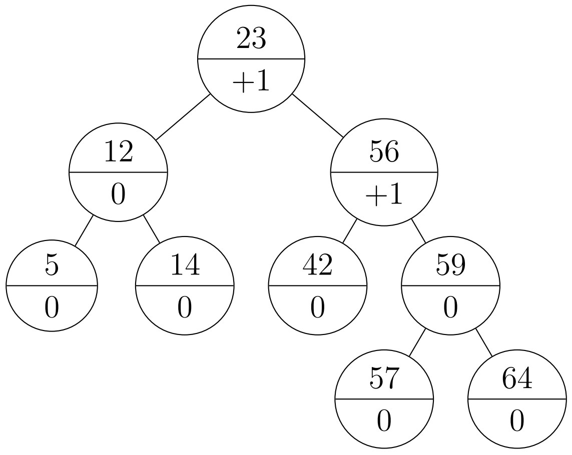
c) Let $T^{exp}(n)$ be the expected running time of *ArrayAlg* of size n. Show how to derive a recurrence relation for $T^{exp}(n)$ and then solve it. Express $T^{exp}(n)$ using big-O asymptotic notation. Your bound must be asymptotically tight, but you need not prove that it is tight.

Problem 3 [0 + 3 + 3 + 3 = 9 marks]

Work with an AVL tree:

a) **Practice** (not worth any marks): Starting with an empty AVL tree, insert the following keys in order: 42 5 12 14 23 56 57 59 64. You should obtain the AVL tree given in the next part.

b) Given the following AVL tree: Note: this tree shows balance factors instead of height.



Insert the following keys in order: 58⋆, 16, 15, 22⋆. Show the resulting AVL trees with balance factors (not height) for each node after the elements marked with star (⋆) are inserted.

Note: you should only show 2 trees.

c) Consider all the nodes in the tree we get at the end of part (b). Suppose we start with an empty tree. What is an ordering for inserting the nodes into the tree that does not cause any rotations? **Write** your example, then **propose** a general idea of creating such orderings and **justify** your answer.

d) Delete the following keys in order: 23⋆, 12, 14, 5⋆.
Show the resulting AVL trees with **balance factors** (not height) for each node after the elements marked with star (⋆) are deleted. If you have a choice of which element to move up, pick the inorder successor.

Note: you should only show 2 trees.

Problem 4 [10 marks]

Given two *AVL trees T1* and *T2*, where the largest key in *T1* is less than the smallest key in *T2*, *Join(T1,T2)* returns an *AVL tree* containing the union of the elements in *T1* and *T2*. Give an algorithm (in pseudocode) for *Join* that runs in time $O(\log n)$, where n is the size of the resulting *AVL tree*. Justify the correctness and efficiency of your algorithm.
Assume the height of *T1* is greater than or equal to the height of *T2*; the other case is symmetric.

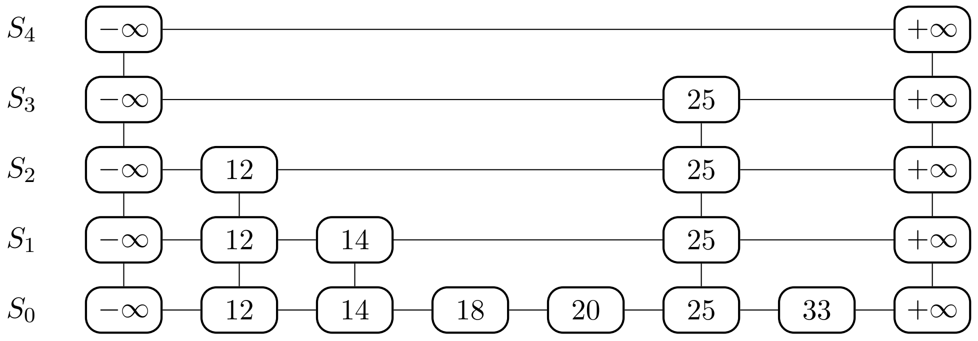
Problem 5 $[0 + 3 + 3 + 3 + 3 = 12$ marks$]$

a) Practice (not worth any marks): Starting with an empty skip list, insert the following keys in order: 18 12 33 25 14 20 using the following coin flip sequence:

THHTTHHHTHTTHT

You should obtain the skip list given in the next part.

b) Given the following skip list:



Show the skip list that is created by inserting the keys: 28, 42, 8, 1, 22, 99 into the given skip list. You must use the following coin flip sequence:

$$THTTHHHHTTTTHTTHT$$

Note that each coin flip in the sequence will only be used once, in order and there may be some unused coin flips.

After inserting all keys, determine the exact number of comparisons (between 2 keys) required to search for the keys inserted in this part (that is, indicate what the search cost would be for 28, and then the search cost for 42, and so on).

For example, a search for 18 in the given skip list above (from Part (a)), requires 6 comparisons.

Key	28	42	8	1	22	99
Comparisons						

For the remaining parts, assume that the probability of adding a level to a tower is p $(0 < p < 1)$, as opposed to $\frac{1}{2}$.

c) Explain why the probability that a tower in the skip list with height at least i is p^i.

d) Show that the expected number of extra nodes (i.e., the total number of nodes in skip list not including the nodes in S_0) is $O(n)$. Therefore, the space requirements for this skip list are linear in the number of keys being stored.

e) Show that the expected height of a skip list is at most $\log_{1/p}(n) + 1/(1-p)$.
Remember to also consider the left and right sentinels.
To make the math a bit easier, you may assume n is a power of $1/p$.