Practical Task 4.1

(Pass Task)

Submission deadline: 10:00am Monday, August 19
Discussion deadline: 10:00am Saturday, August 31

General Instructions

A skip-list is a probabilistic linked-list-like data structure that expects, for an ordered sequence of n elements, a $O(\log n)$ runtime complexity for basic operations such as searching, insertion and removal. Importantly, it supports a quick insertion that is not possible for a simple array. Fast operations are made possible by maintaining a linked hierarchy of layered linked-list subsequences, with each successive subsequence skipping over fewer elements than the previous.

1. This task asks you to conduct a small research about this data structure. You can find its detailed explanation in chapter 10.4 of the course book "Data Structures and Algorithms in Java". You, of course, may explore and refer to any other resources covering this topic. We expect you to grasp the idea and structure of a skip-list along with standard operations on it. As the results of your study, you must be able to answer the following questions and discuss the related issues:
   - How exactly is a skip-list constructed from a series of linked-lists? What does each of the linked-lists constituting a skip-list store?
   - What is a height of a skip-list? How can we determine the height when adding new elements?
   - What is the runtime complexity for searching, insertion, and removal operations on a skip-list? What does it mean when we say that the complexity of these operations is '*expected*'? What is the difference between an *expected* bound on runtime complexity and a *worst-case* bound?
   - Compare a skip-list to other data-structures that you should already know, e.g. sorted arrays, singly and doubly linked lists. What are the advantages and disadvantages of a skip-list? Compare the complexities of the basic operations offered by a skip-list to those of the mentioned data structures.
   - How exactly does each of the standard operations work? Here, you should be ready to explain the sequence of actions required to perform searching, insertion, and removal from a skip-list.
2. Solve the following numeric example. Given the following sequence of *coin tosses*, where H stands for *head* and T stands for *tails*:

   T H T T H H T T H T H T H H T H H T T H H T H T T H T T T H T H T H T T H T H H T T.

   Build a skip-list containing the following values:

   1 40 11 85 86 5 0 8.

   Show the full state of the skip-list after each insertion. Note that *you must explicitly state the meaning you attach* to *heads* and *tails* at the start of your answer. Remember that you must start tossing the coin from left side of the sequence and may not need to use all the tosses to build the skip-list. You may assume as a height resolution policy that we allow a tower to grow as long as heads (or tails, as both can be used interchangeably) keep getting returned from the given sequence, which emulates a random number generator.

Further Notes

– You will find the answer to this task by reading chapter 10.4 of the course book "Data Structures and Algorithms in Java" by Michael T. Goodrich, Irvine Roberto Tamassia, and Michael H. Goldwasser (2014). You may access the book on-line for free from the reading list application in CloudDeakin available in

Resources → Additional Course Resources → Resources on Algorithms and Data Structures → Course Book: Data structures and algorithms in Java.

Marking Process and Discussion

To get this task completed, you must finish the following steps strictly on time:

- Submit your answers to the task via OnTrack submission system. You may submit a hand-written and then scanned document, but ensure that the text is very clear to read. Note that this is a theoretical task, thus we do not expect you to write any program code.
- Meet with your marking tutor to explain your solutions. When the solution is hand-written, do not forget to bring it with you.
- Show to your tutor how the standard operations on a skip-list are performed. Cloud students must record a short video explaining their work on the numeric instance along with execution of searching, insertion and removal operations based on the example.
- Answer all additional (theoretical) questions that your tutor may ask you. Please, come prepared so that the class time is used efficiently and fairly for all the students in it. You should start your interview as soon as possible as if your answers are wrong, you may have to pass another interview, still before the deadline. Use available attempts properly.

Note that we will not check your solution after the submission deadline and will not discuss it after the discussion deadline. If you fail one of the deadlines, you fail the task and this reduces the chance to pass the unit. Unless extended for all students, the deadlines are strict to guarantee smooth and on-time work through the unit.

Remember that this is your responsibility to keep track of your progress in the unit that includes checking which tasks have been marked as completed in the OnTrack system by your marking tutor, and which are still to be finalised. When marking you at the end of the unit, we will solely rely on the records of the OnTrack system and feedback provided by your tutor about your overall progress and quality of your solutions.