Tutorial (week 4)

All the answers can be given with pseudocode or with Python. Remember that most of the time, many solutions are possible.

1. Write a function `sorted_insert(h,t,e)` which given the head sentinel `h` and the tail sentinel `t` of an already sorted list (in increasing order) and a single node `e`, inserts the node into the correct sorted position in the list.

2. Write a function `insertion_sorted(h,t)` which given the head sentinel `h` and the tail sentinel `t` of a list, will rearrange the nodes so that they are sorted in increasing order. You can use the function `sorted_insert(h,t,e)` from previous exercise.

3. Describe, using Python or pseudocode, an implementation of the function `insertBefore(p, e)`, for a linked list, assuming the list is implemented using a doubly linked list.

4. A double-ended queue, or *deque*, is a list that allows for insertions and removals at either its head or its tail. Describe a way to implement a deque using a doubly linked list, so that every operation runs in $O(1)$ time.

5. Describe, in Python or pseudocode, a link-hopping method `middle_node(h,t)` for finding the middle node of a doubly linked list with header and trailer sentinels (`h` and `t`), and an odd number of real nodes between them (note that this method must only use link hopping, it cannot use a counter). What is the running time of this method ?

6. Describe the structure and Python code/pseudocode for an array-based implementation of an index-based list that achieves $O(1)$ time for insertions and removals at index 0, as well as insertions and removals at the end of the list. Your implementation should also provide for a constant-time `get` method.

7. Using an array-based list, describe an efficient way of putting a sequence representing a deck of n cards into random order. Use the function `random.randint(a,b)` from the `random` module, which returns a random integer between a and b inclusive. Your method should guarantee that every possible ordering is equally likely. What is the running time of your method?

8. In the children's game "hot potato", a group of n children sit in a circle passing an object, called the "potato", around the circle (say in a clockwise direction). The children continue passing the potato until a leader rings a bell, at which point the child holding the potato must leave the game, and the other children close up the circle. This process is then continued until there is only one child remaining, who is declared the winner. Using a list, describe an efficient method for implementing this game. Suppose the leader always rings the bell immediately after the potato has been passed k times. What is the running time of your method in terms of n and k, assuming the list is implemented with a doubly linked list ? What if the list is implemented with an array ?

Hints

- **Question 3:** review the code for `insertAfter(p, e) in the textbook`.
- **Question 5:** consider a combined search from both ends.

- **Question 6:** think about how to extend the circular array implementation of a queue (given in the textbook).
- **Question 7:** consider randomly shuffling the deck one card at a time.