Data Structures and Algorithms
Week 6 problem sheet

## A. ADTs

1. Which of the following are ADTs and which are concrete data types?
	* array
	* linked listy
	* graph
	* binary search tree
	* queue
	* priority queue
	* hash table
*
1. Suppose you are asked to create an ADT intended to represent a *lock*. A lock can be locked and unlocked, and should have accessor methods so the lock’s state can be observed. Show the Java code you would use for this ADT.

## B. Stacks and queues

1. Suppose we have a some integers, and perform the following operations with them on a stack or queue object:
	* add 3, add 2, add 5, remove, remove, add 9
* What would the object contain, after each operation, if we used a …
	+ … array-based stack, with capacity 5?
	+ … linked list–based queue?
	+ … priority queue?
*

## C. Linked lists

1. Suppose we want to write a Node class for use in a linked list that stores Strings.
* Write code for this class.
1. How would the class in question 1 change if we wanted to use a generic class?

## D. Big “O” notation

1. Suppose we have one algorithm *A* that runs in $O(logn)$ time, and another algorithm *B* that runs in $O(n)$ time.
* We write a program in which we run algorithm *A*, then algorithm *B*. What is the big “O” complexity of our program?

# E. Sorting and search

1. Suppose we want to sort an array, but it turns out (unbeknownst to us) that the array is already sorted.
* Of the sorting algorithms we have seen (insertion sort, merge sort and quick sort), which is likely to perform *worst*? Why?
1. Suppose we want to perform binary search on an un-sorted list of numbers. Can we do so? If so, how? If not, why not?

# F. Trees



1. Is the tree shown above a binary tree?
* Is it also a binary search tree?
* Is it also a *balanced* binary search tree?
* Explain why or why not.
1. Suppose we wanted to use binary search to see if the number 8 was in the tree. What steps would be involved?
2. Can we use a binary search tree to implement the Set ADT? How would we implement the “add” operation?

# G. Graphs

1. Suppose we have a very dense weighted graph, and will be using Prim’s algorithm to obtain a minimum spanning tree from it.
* Would it be better to use an adjacency list implementation, or an adjacency matrix implementation? Explain why.
1. Are all trees also graphs? Explain why or why not.