Homework 2

1: Extendible Arrays

Consider a data structure supporting the following operations:

- Init – create an array A of length zero.
- Extend(A) – increase the length of A by one and let the new entry contain nil.
- Shrink(A) – decrease the length of A by one.
- Write(A, i, v) – set the ith entry of A to v.
- Read(A, i) – return the ith entry of A.

The operations may use a procedure alloc(n) that in worst case constant time returns an uninitialized array of length n, *i.e.* all entries of the allocated array contain random information, and a procedure dealloc(A) that deallocates an already allocated array A.

a) Give an implementation that supports all operations in amortized $O(1)$ time and where the space required for an array is linear in the length of the array.

b) Show that it also can be done with "amortized" changed to "worst case".

2: Binomial Heaps

In the section on binomial heaps in [Kozen, Lecture 8], it is shown that the "lazy" version supports the operations Insert, Meld, and DeleteMin in amortized time $O(1)$, $O(1)$, and $O(\log n)$, respectively. Show that the result is also true for the original "eager" version.

3 Dijstra's Algorithm

Prove optimality (in a comparison based model) of the implementation of Dijkstra's algorithm by Fibonacci heaps as follows:

a) Show that any algorithms for finding a shortest path between two vertices in a graph must make $\Omega(m)$ queries to edge weights. Your proof should work for all values of m between $\Theta(n)$ and $\Theta(n^2)$.

b) Show that Dijkstra's algorithm can be used to sort n numbers.

4: Prim's Algorithm

In a connected undirected graph, a *spanning tree* is a subgraph which is a tree, and which contains all nodes. If there are weights on the edges of the graph, the weight of a spanning tree is the sum of the weights of its edges. The *Minimum Spanning Tree* (MST) problem is to find a spanning tree of minimum weight among all spanning trees of the graph.

The following algorithm, known as *Prim's algorithm*, produces a minimum spanning tree T. Initially, we choose an arbitrary vertex and let T be the tree consisting of that vertex and no edges. We then repeat the following step $n-1$ times: find an edge of minimum weight with exactly one endpoint in T and include that edge in T.

a) Prove that Prim's algorithm solves the MST problem.

b) Give an implementation that runs in time $O(m + n \log n)$.

5: The Stable Marriage Problem *

In a group of n boys and n girls, each girl ranks the n boys according to her preference, and each boy ranks the n girls according to his preference. A *marriage* is a matching between the boys and the girls such that everybody is married (a *perfect* matching in graph terminology). A marriage is *unstable* if there is a pair who are not married to each other but who like each other more than they like their respective spouses, otherwise it is *stable*. Prove that a stable marriage always exists and give an efficient (i.e. polynomial time) algorithm to find one.