LaSalle College
Felix Soumpholphakdy

January 20, 2019
Advanced Data Structure: 420-J13-AS

Practice Set 2

A. The Fibonacci sequence is composed of numbers where each number is the sum of the two previous numbers:

   $F_n = F_{n-1} + F_{n-2}$, where $F_0 = 0$, $F_1 = 1$

   Using recursion, implement the procedure FIBONACCI(*n*) which computes the n^{th} Fibonacci number.

B. Given a sorted array, we can use binary search to find an element in O(lg *n*) time. Using recursion, implement the procedure BINARY-SEARCH(*A*, *value*, *start*, *end*) which searches for the element *value* in the sorted array *A* from index *start* to *end*. The result is the index of given element. For extra practice, implement it again without recursion.

C. Implement MERGE(*A*, *p*, *q*, *r*) which merges the sorted subarrays A[*p*..*q*] and A[(*q*+1)..*r*] back into the array *A*.

D. Implement MERGE-SORT(*A*, *p*, *r*) which sorts the elements from *p* to *r* in an array *A*.

E. Implement MAX-HEAPIFY(*A*, *i*, *h*) which processes the nodes in array *A* of the subtree at index *i* with heap size *h*, to maintain the max-heap property at *i*.

F. Implement BUILD-MAX-HEAP(*A*) which converts an entire array *A* into a heap.

G. Implement HEAPSORT(A) using the previous procedures MAX-HEAPIFY and BUILD-MAX-HEAP to sort the array *A*.

H. ***HARD*** (CLRS 6.5-9) Give an O(*n* lg *k*)-time algorithm to merge *k* sorted lists into one sorted list, where *n* is the total number of elements in all the input lists. (Hint: Use a min-heap for *k*-way merging.)