# Problem Set 4

Ekkapot Charoenwanit
Efficient Algorithms

August 27, 2020

**Problem 4.1. Direct Addressing**

1) Suppose that a dynamic set S is represented by a direct-address table $T$ of length $m$. Describe a procedure that finds the maximum element of $S$. What is the worst-case performance of your procedure?

**Problem 4.2. Hashing**

1) Demonstrate what happens when we insert the keys $< 14, 37, 10, 33, 47, 6, 21, 8, 55 >$ into a hash table with collisions resolved by chaining. Let the table have 9 slots, and let the hash function be $h(k) = k \mod 9$.

2) Explain why the delete operation for hash tables does not work in $\mathcal{O}(1)$ time when separate chaining is implemented using singly-linked lists instead of doubly-linked lists.

3) Suppose we come up with a hash function for computing the hash values of names using the following scheme.

$h(S) = \sum_{i=0}^{S.length-1} ASCII(S[i])$, where $S$ is a string of characters of length $S.length$ and the function $ASCII$ returns the ASCII value of a character.

What is wrong with this hash function?

As a hint, consider the following situation when we use it to hash these three names,"Lee Chin Tan","Chen Le Tian" and "Chan Tin Lee".

**Problem 4.3. Open Addressing**

1) Suppose we want to delete an element with key $k$. What is wrong with deleting the element by simply marking the slot as empty? You may provide a counter example of situations when doing this will cause searching to behave incorrectly.

2) Write pseudocode for HASH-DELETE and modify HASH-INSERT to handle the special value $DELETED$.