Problem Set 11

Question 1.

(a) In your own words, describe what a "Greedy" algorithm is.

(b) Give some examples of problems that a greedy algorithm would **not** be able to solve optimally.

Question 2. *Averaging down* (Levitin 9.1.4)
There are $n > 1$ identical vessels, one of them with W liters of water and the others are all empty. We can perform the following operation: Take two of the vessels, and split the total amount of water in them equally between them.

The goal is to achieve a minimum amount of water in the vessel that begins with W liters of water by a sequence of operations. Give an algorithm in pseudocode that achieves this and reason why it works.

Question 3. *Bachet's problem of weights*

Find an optimal (i.e. smallest) set of n weights $\{w_1, w_2, ..., w_n\}$ so that it would be possible to weigh on a balance scale any integer load in the largest possible range from 1 to 13, provided

(a) weights can be put only on the free cup of the scale,

(b) weight can be put on both cups of the scale.

(c) Provide pseudocode for solving the general case with for the range $1...W$. Justify why your algorithm works.

Question 4.

(a) Given a hash table with 9 slots and the hash function $h(k) = k \mod 9$.[1]

Demonstrate what happens when we insert the keys 82; 7; 18; 17; 19; 23; 21; 14; 10 into a hash table. If two values have the same hash, we say that they *collide*, put them in a **(singly) linked list**.

(b) Operation on a hash table are:

- `Insert(k)`, insert a key, k, into the hash table.
- `Delete(k)`, delete the instance of k (if it exists) from the hash table.
- `Search(k)`, If k exists in the table, return it.

For the hash map above, explain and provide the worst-case time complexities for these operations.

(c) It is known that in the average case these operations can be performed in $\Theta(1)$. What factors do you think are relevant for making a hash table fast?

[1] That is to say, every key, k, is sent to the location with index $h(k)$, we say that $h(k)$ is the *hash of k*