# Assignment 1

Theodore S. Norvell

6892 Due 2017 Oct 17th.

Submission will be by D2L. Please put your work for question 0 and question 1 in separate files. Make sure your name is included in comments at the top of the file.

**Q0 [20] Square Roots.**

(a)[5] Write a contract for a method to compute the integer part (floor) of the square root of a positive integer.

(b)[5] Design a verified Dafny method to compute the integer part (floor) of the cube root of a positive integer. For this part your algorithm should use a linear search. I.e., it should try 0, then 1, then 2, and so on.

(c)[5] Improve your algorithm as follows. Instead of incrementing by one in each iteration, increment by the largest power of two that works.

(d)[5] By introducing tracking variables, eliminate all multiplications from the executable part of your algorithm. (Multiplications by small constants you can leave if you like.)

**Q1 [20]** Insertion Sort. See file insertionSort.dfy on the course web site.

We will talk about Dafny and arrays on Oct 12th. In the mean time, here is some useful information. You may also wish to read ahead in the notes.

- If arrays are modified in a method, they must be declared in the method's modifies clause.
- If `a` is an array, then `a[..]` is a sequence (seq) of all the values of the items in the array from left to right. Also `a[..k]` is a sequence of the values of the first `k` items, `a[i..k]` is the sequence of the values of the `k-i` items starting with `a[i]`.
- If `s` and `t` are sequences, `s+t` is the concatenation of the two sequences.
- Universal quantification in Dafny looks like this

  ```
  forall j:int ::  i<=j<k ==> a[j]==0
  ```

  This example says all items of the sequence `a[i..k]` are equal to 0. The `==>` operator is implication. Note that Dafny uses left-to-right 3-valued logic (short-circuiting), so it doesn't matter if `a[j]` is undefined for some values of `j` outside the range of interest.
- Existential quantification is similar. E.g.

  ```
  exists j:int ::  i<=j<k && a[j]==0
  ```

- Dafny allows one to define predicates to be used in assertions and contracts. See the insertionSort.dfy file for examples of predicates.

(a) [10] Design a body for the `insertionSort` method. Ensure that your code verifies. (Hint: Use the supplied `insert` method.)

(b) [10] Add a body for the `insert` method. Ensure the code verifies.

Advice: Draw a picture of your invariants.

Advice: I found that algorithms that only modify the array by swapping two elements to be the easiest for the verifier to check.

**Bonus:[10]** Implement and verify one of the following using Dafny: Heap sort, merge sort, quick sort.