Topics covered by Test 1 (CSCE 221)

You should be familiar with the following topics.

1. Analysis of iterative algorithms.
   (a) Understand behavior of an algorithm using a small input data
   (b) Find a running time function for this algorithm
   (c) Use Big-O asymptotic notation to classify this algorithm
   (d) Find an input such that the algorithm takes the minimum number of operations (the best case)
   (e) Find an input such that the algorithm takes the maximum number of operations (the worst case)
   (f) Find the average case for this algorithm

2. Practice problems.
   (a) Homework questions
   (b) Assignment 1: `My_vec` implementation
   (c) Find a running time function and use the Big-O notation to classify the algorithms below

```
//-Algorithm 1--------------------------------------------------
   int i = n;
   while(i > 0) {
      i /= 10;
   }
//-Algorithm 2--------------------------------------------------
   int i = n; int s = 0;
   while(i > 0) {
      for(j = 1; j <= i; ++j) s++;
      i /= 2;
   }
//-Algorithm 3--------------------------------------------------
   int i = n;
   while(i > 0) {
      for(j = 1; j <= n; ++j) s++;
      i /= 2;
   }
//-Algorithm 4---------------------------------------------------
   int i = n;
   for(j = 1; j <= n; ++j)
      while(i > 0) i /= 2;
//-Algorithm 5---------------------------------------------------
   for(j = 1; j <= n; ++j) {
      int i = n;
      while(i > 0) i /= 2;
   }
```

3. Provide the best algorithm for all the problems below. Write their running time functions using Big-O notation.

   (a) Find the inner product of two vectors `v1` and `v2`, each of size `n`.
   (b) Find the number of occurrences of an element `t` in a vector/array `v`, of size `n`.
   (c) Concatenate two strings, each of `n` characters long.
   (d) Remove the `i-th` element from a vector/array
   (e) Find in a vector/array the sum of only those elements which are less than 43.
   (f) Add an element to the beginning/end of a vector/array `v`.

4. List the sorting algorithms implemented based on comparisons. Illustrate their behavior using a small input data. Provide their best, worst and average cases.

5. The lower bound theorem and its consequences.

6. List the sorting algorithms which are non-comparison based. Provide assumptions on the input data that ensure linear running time. Provide conditions on the input sequence that allow you to make decisions about the best choice of a non-comparison based algorithm to sort input data in linear time.

7. Stack ADT (it will covered on Monday).

   (a) Running time of its operations in Big-O notation using:
      i. the worst case analysis
      ii. the amortized analysis

8. Be familiar with material covered in class, on slides and in the course textbook.

   (a) Chapter 4: Analysis of algorithms. Sorting algorithms and their analysis
      i. insertion sort, pp. 109–111, 336
      ii. bubble sort, pp. 259–261
      iii. selection sort, p. 335
      iv. counting sort – see slides
      v. radix sort, pp. 529–530
      vi. bucket sort, p. 528
      vii. definition of the stable algorithm, p. 529
      viii. the lower bound theorem, pp. 526–527
   (b) Stack implemented based on a simple array, p. 194
   (c) Stack implemented based on expandable/resizable arrays. Amortized cost for push and enqueue operations, p. 231.