CS 173 (B), Spring 2015, Examlet 7, Part A

NAME:	**NETID:**

Discussion Section: BDA:1PM BDB:2PM BDC:3PM BDD:4PM BDE:5PM

1. Consider the following sorting algorithm, which you already encountered in a homework problem (rewritten here more clearly):

```
1:  function BubbleSort($A$ : array of numbers, $n$: size of array)
2:      $t := n$                                ▷ $t$ such that we need to sort $A[1]..A[t]$
3:      while $t > 1$ {
4:          $d := 0$
5:          for $i := 1$ to $t - 1$ {
6:              if $A[i] > A[i+1]$ {
7:                  SWAP$(A, i, i+1)$           ▷ swaps the entries $A[i]$ and $A[i+1]$ in constant time
8:                  $d := i$
9:              }
10:         }
11:         $t := d$                            ▷ $d = \max i$ for which swap occurred. $A[d+1]..A[n]$ finalized.
12:     }
13:     return $A$
```

   (a) Execute this algorithm on input ($A = [2, 1, 3], n = 3$). Write down all the values that get assigned to t in step 11. Write down the final array returned by the function. [3 points]

   Solution: Step 11 is encountered only once, and t is set to 1 then. Finally, $A = [1, 2, 3]$.

   (b) For each of the following arrays A, give a bound on the running time of the algorithm on input (A, n) (in the form $\Theta(f(n))$)? Briefly justify your answers. [10 points]

   i. $A = [1, 2, \ldots, n]$

   Solution: $\Theta(n)$. During the first iteration of the outerloop, no swaps take place and in Step 11 t is set to 0. Hence there is only one iteration of the outer-loop. During that iteration, the inner-loop iterates $n - 1$ times (each iteration taking $\Theta(1)$ time).

   ii. $A = [n, 1, 2, \ldots, n - 1]$

   Solution: $\Theta(n)$. During the first iteration of the outer-loop, the first entry n is bubbled all the way to the end of the array, to result in a sorted array; in Step 11, t is set to $n - 1$. In the second iteration of the outer-loop, no swaps take place, and in Step 11, t is set to 0. Hence there are only two iterations of the outer-loop, and the inner-loop iterates $n - 1$ and $n - 2$ times respectively in those two iterations.

   iii. $A = [2, 3, \ldots, n, 1]$

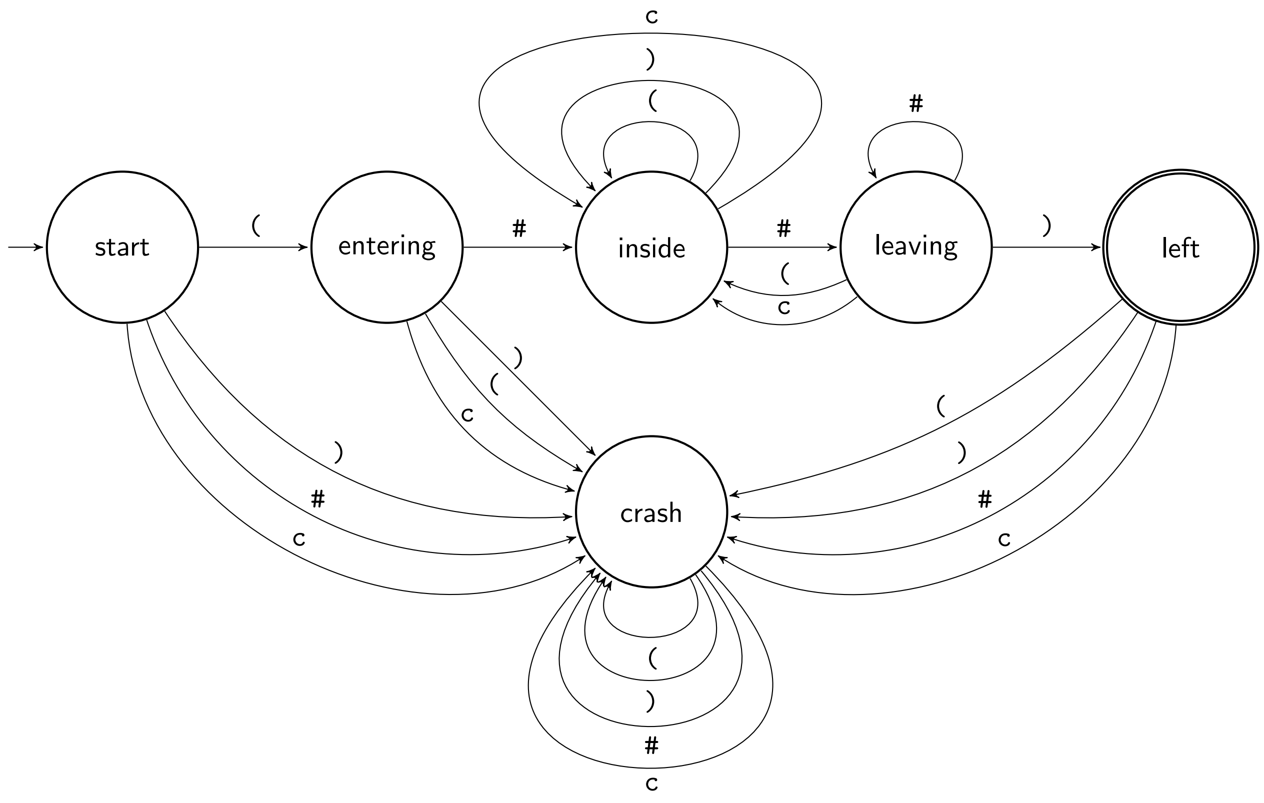
   Solution: $\Theta(n^2)$. In the first iteration of the outer-loop, no swaps take place except at the last iteration of the inner-loop, when $A[n-1] = n$ and $A[n] = 1$ are swapped; in Step 11, t is set to $n-1$. The array $A = [2, 3, \ldots, n-1, 1, n]$ at the end of this iteration. More generally, in every iteration of the outer-loop, t is decremented by 1 and the inner-loop is iterated $t - 1$ times. Hence the total number of iterations of the inner-loop is $(n - 1) + (n - 2) + \cdots + 1 = \Theta(n^2)$.

2. State Diagram. [12 points]

You are in the process of designing a simple programming language. You have decided that comment strings start with `(#` and continue until terminated with `#)`. For example, `(#hello#)`, `(##)` and `(#(##)` are valid comment strings. However, `(#hey#)(#huh#)` and `(#)` are not.

You should complete the state diagram below for a deterministic finite state acceptor that accepts only valid comment strings (fed left to right). For simplicity use an alphabet that consists of only four characters, `(`, `)`, `#` and `c`.

Add all the edges and clearly mark the labels on the edges. Remember to mark the start state and final state(s) using the standard convention in state diagrams.



Description of the states:

- entering: have seen one symbol that looks like the beginning of a comment.
- inside: have seen a comment's beginning sequence.
- leaving: have seen one symbol that looks like the ending of a comment.
- left: just finished seeing an entire comment.
- crash: have seen a string that cannot become a valid comment.

Be careful while designing the transition function. You may want to test your solution on a few inputs like `(#c)#c#)` (valid comment) and `(#c)#c)` (invalid comment).

CS 173 (B), Spring 2015, Examlet 7, Part A

NAME:	NETID:

Discussion Section: BDA:1PM BDB:2PM BDC:3PM BDD:4PM BDE:5PM

1. Consider the following sorting algorithm, which you already encountered in a homework problem (rewritten here more clearly):

```
1:  function BUBBLESORT($A$ : array of numbers, $n$: size of array)
2:      $t := n$                                  ▷ $t$ such that we need to sort $A[1]..A[t]$
3:      while $t > 1$   {
4:          $d := 0$
5:          for $i := 1$ to $t - 1$   {
6:              if $A[i] < A[i + 1]$   {
7:                  SWAP$(A, i, i + 1)$           ▷ swaps the entries $A[i]$ and $A[i + 1]$ in constant time
8:                  $d := i$
9:              }
10:         }
11:         $t := d$                  ▷ $d = \max i$ for which swap occurred. $A[d + 1]..A[n]$ finalized.
12:     }
13:     return $A$
```

(a) Execute this algorithm on input $(A = [1, 2, 0], n = 3)$. Write down all the values that get assigned to t in step 11. Write down the final array returned by the function. [3 points]

Solution: Step 11 is encountered only once, and t is set to 1 then. Finally, $A = [2, 1, 0]$.

(b) For each of the following arrays A, give a bound on the running time of the algorithm on input (A, n) (in the form $\Theta(f(n))$)? Briefly justify your answers. [10 points]

i. $A = [n, n - 1, \dots, 1]$

Solution: $\Theta(n)$. During the first iteration of the outerloop, no swaps take place and in Step 11 t is set to 0. Hence there is only one iteration of the outer-loop. During that iteration, the inner-loop iterates $n - 1$ times (each iteration taking $\Theta(1)$ time).

ii. $A = [1, n, n - 1, \dots, 2]$

Solution: $\Theta(n)$. During the first iteration of the outer-loop, the first entry n is bubbled all the way to the end of the array, to result in a sorted array; in Step 11, t is set to $n - 1$. In the second iteration of the outer-loop, no swaps take place, and in Step 11, t is set to 0. Hence there are only two iterations of the outer-loop, and the inner-loop iterates $n - 1$ and $n - 2$ times respectively in those two iterations.

iii. $A = [n - 1, n - 2, \dots, 1, n]$

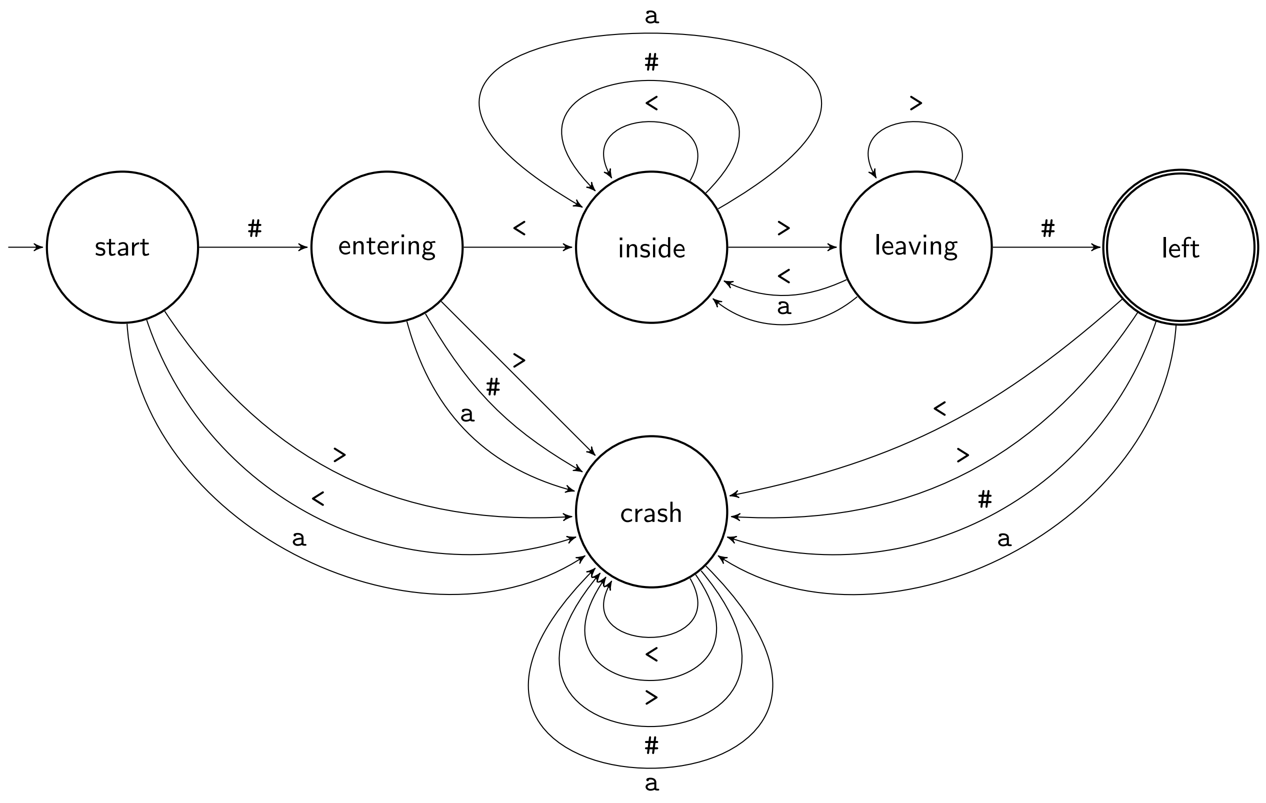
Solution: $\Theta(n^2)$. In the first iteration of the outer-loop, no swaps take place except at the last iteration of the inner-loop, when $A[n - 1] = 1$ and $A[n] = n$ are swapped; in Step 11, t is set to $n-1$. The array $A = [n-1, n-2, \dots, n-1, n, 1]$ at the end of this iteration. More generally, in every iteration of the outer-loop, t is decremented by 1 and the inner-loop is iterated $t-1$ times. Hence the total number of iterations of the inner-loop is $(n - 1) + (n - 2) + \cdots + 1 = \Theta(n^2)$.

2. State Diagram. [12 points]

You are in the process of designing a simple programming language. You have decided that comment strings start with `#<` and continue until terminated with `>#`. For example, `#<hello>#`, `#<>#` and `#<#<>#` are valid comment strings. However, `#<hey>##<huh>#` and `#<>` are not.

You should complete the state diagram below for a deterministic finite state acceptor that accepts only valid comment strings (fed left to right). For simplicity use an alphabet that consists of only four characters, `<`, `>`, `#` and `a`.

Add all the edges and clearly mark the labels on the edges. Remember to mark the start state and final state(s) using the standard convention in state diagrams.



Description of the states:

- entering: have seen one symbol that looks like the beginning of a comment.
- inside: have seen a comment's beginning sequence.
- leaving: have seen one symbol that looks like the ending of a comment.
- left: just finished seeing an entire comment.
- crash: have seen a string that cannot become a valid comment.

Be careful while designing the transition function. You may want to test your solution on a few inputs like `#<a#>a>#` (valid comment) and `#<a#>a#` (invalid comment).