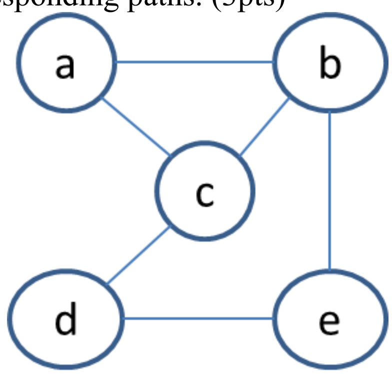
Homework #1

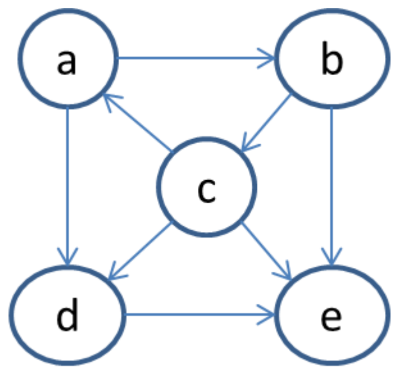
Due at 10:05 am EST, Thursday, Jan. 20

1. a. Assume you have an empty stack, show the stack after a sequence of operations: push(a), push(b), pop, push(c), push(d), pop, pop, push(e), pop. (10pts)
b. Assume you have an empty queue, show the queue after a sequence of operations: enqueue(a), enqueue(b), dequeue, enqueue(c), enqueue(d), dequeue, dequeue, enqueue(e), dequeue. (10pts)

2. For the graphs in Figure 1,
(a) For both figures, write the **adjacency matrix** and the **adjacency linked list**. (10pts)
(b) Is Figure (A) a complete graph? Why? (4pts)
(c) For both figures, are there any loops in each graph? If so, write down the corresponding edges. (3pts)
(d) For both figures, are there any cycles in each graph? If so, write down the corresponding paths. (3pts)



(A)



(B)

Figure 1

3. How would you implement a dictionary of a reasonably small size n of you knew that all its elements are distinct (e.g., names of the countries in the world)? Specify an implementation of each dictionary operation. (20 pts)

4. Consider the following algorithm for finding the distance between the two closest elements in an array of numbers.

Algorithm MinDistance(A[0..n − 1])
//Input: Array A[0..n − 1] of numbers
//Output: Minimum distance between two of its elements
dmin ←∞
for i ← 0 to n − 1 do
    for j ← 0 to n − 1 do

if $i \neq j$ and $\sqrt{(\mathrm{A[i]} - \mathrm{A[j]})^2} <$ dmin
dmin ← $\sqrt{(\mathrm{A[i]} - \mathrm{A[j]})^2}$

return dmin

(a) What is the basic operation of this algorithm? For the worst case, how many times is it performed as a function of the array size *n*? How about the best case? (10pts)

(b) Make as many improvements as you can in this algorithmic solution to the problem. You must write down the pseudocode for your new algorithm. What is the basic operation of your new algorithm? How many times is the basic operation performed as a function of the array size *n*? (If you need to, you may change the algorithm altogether.) (10pts)

5. We have introduced the Selection Sort as shown in the Algorithm below. Is Selection Sort stable? Is it in place? Explain your assertion. (20pts)

ALGORITHM *SelectionSort*($A[0..n-1]$)
//Sorts a given array by selection sort
//Input: An array $A[0..n-1]$ of orderable elements
//Output: Array $A[0..n-1]$ sorted in ascending order
for $i \leftarrow 0$ **to** $n-2$ **do**
  $min \leftarrow i$
  for $j \leftarrow i+1$ **to** $n-1$ **do**
    if $A[j] < A[min]$ $min \leftarrow j$
  swap $A[i]$ and $A[min]$

Bonus question: (10pts)

Prove that a complete binary tree has the height of $h = \lfloor \log_2 n \rfloor$, where n is the total number of nodes in the tree and $\lfloor \ \ \rfloor$ is the floor function.