COMP 3270 Assignment 5 20 Multiple Choice Questions No late submissions!

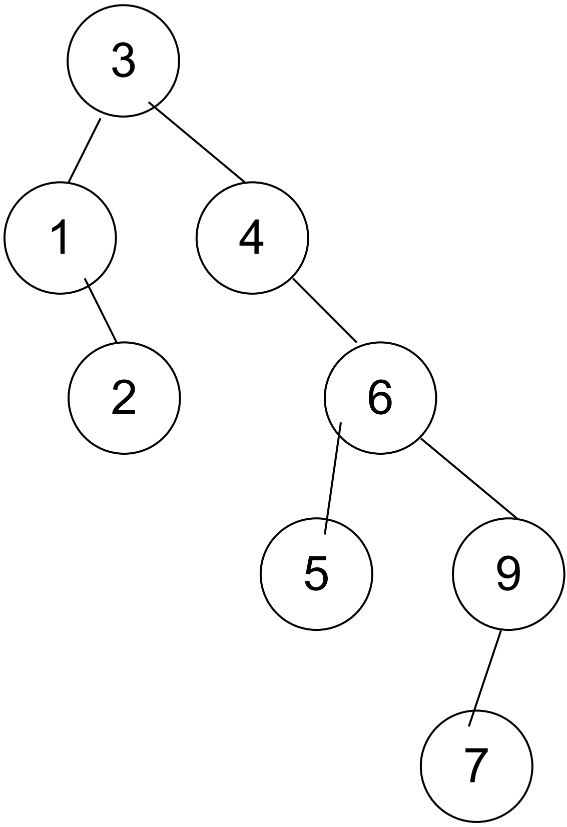
Due before 10:15 AM Tuesday November 16

Instructions:

1. This is an individual assignment. **You should do your own work.**
2. Late submissions will **not** be accepted.
3. Mark answers on a **scantron** sheet. **Fill in your name and Banner ID correctly on the sheet**.
4. Hand over the sheet to your course instructor or to a TA **before the due date & time**.
5. All questions carry 2 points.

Binary Search Trees (BST)

1. Draw the Binary Search Tree resulting from inserting 3,1,4,6,9,2,5,7 one after the other into an initially empty BST and then answer this question: **True or False?** The final BST will have a subtree rooted at 6 with a left child 5 and a right child 9, with 9 having a left child 7.



The BST should look like this

2. We can sort n numbers by first building a BST containing the n numbers (start with an empty BST and call Tree-Insert repeatedly n times) and then outputting the numbers from the BST using an Inorder-Tree-Walk. What is the complexity of this sorting algorithm?

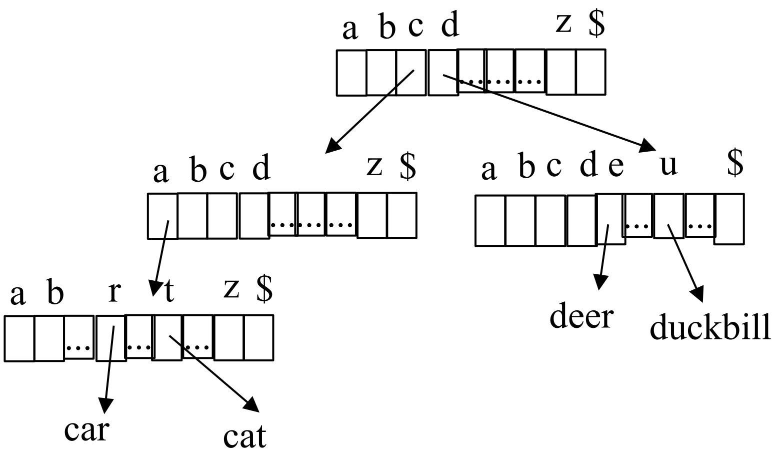
A. O(n) B. $O(n^2)$ C. O(logn) D. O(nlogn) E. none of these

Best case: n Tree-Inserts produce a balanced binary tree of height $\log_2 n$ so cost at most $n*O(\text{height})=n*O(\log_2 n)=O(n\log_2 n)$ followed by a $\Theta(n)$ Inorder-Tree-Walk so net complexity $O(n\log_2 n)+\Theta(n)=O(n\log_2 n)$. Worst case: n Tree-Inserts produce a linear tree of height n so cost at most n*O(height)=n*O(n)=O(n^2) followed by a Θ(n) Inorder-Tree-Walk so net complexity O(n^2)+Θ(n)= O(n^2). Since the question did not specify best/worst case both answers are acceptable.

3. Simulate the Tree-Delete algorithm (slide # 24) on slide set 16 Searching Part I) on the BST on slide 23 of slide set 16 with z=5. Which of the statements below is true?

A. Condition check on step 9 of the algorithm is the one that succeeds.
B. Condition check on step 11 of the algorithm is the one that succeeds.
C. Neither will succeed and step 13 will be executed.
D. Execution will never reach the if statement in steps 9-13.
E. None of the above statements is true.

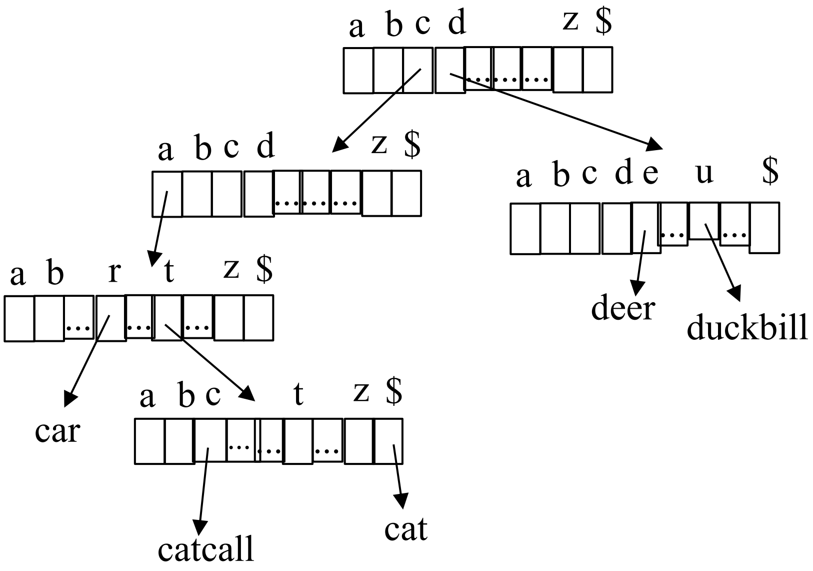
4. Now simulate the TREE-DELETE algorithm (p.298 of the text) on the same BST on slide 23 slide set 16 with z=5 and answer this question: **True or False?** The tree resulting after deletion in this case is different from the tree resulting after deletion using the Tree-Delete algorithm of Q. 3 above.



5. Suppose the word “catcall” is to be inserted into the above Trie. How many additional internal nodes (i.e., nodes with an array of 27 pointers) will get added?

A. 1 B. 2 C. 3 D. 4 E. 5

The new Trie will be:



String Searching

```
NAÏVE-STRING-MATCHER (T,P)
1 n = T.length
2 m = P.length
3 s = 0
4 while s<n–m+1 do
5       for i = 1 to m
6             if P[i] != T[s+i] then
7                   s = s+1
8                   exit the i-loop and go to step 4
9       print "pattern occurs with shift" s
10      s = s+1
```

6. If P=0001 and T=000010001010001, what are the shift values printed by step 9 of the algorithm above?

A. 0, 4 & 10 B. 2, 6 & 12 C. 5, 9 & 11 D. 1, 5 & 11 E. nothing will be printed

```
MODIFIED-NAÏVE-STRING-MATCHER (T,P)
1 n = T.length
2 m = P.length
3 s = 0
4 while s<n–m+1 do
5       for i = 1 to m
6             if P[i] != T[s+i] then
7                   s = s+i
8                   exit the i-loop and go to step 4
9       print "pattern occurs with shift" s
10      s = s+m
```

7. True or False? MODIFIED-NAÏVE-STRING-MATCHER is a correct algorithm.

The algorithm is incorrect due to steps 7 and 10.

A counterexample for which the algorithm fails due to step 7 is P=0001 and T=1000010001010001. Condition check in step 6 will succeed when s=1 and i=4, so s will be updated to 1+4=5 so the algorithm will then try to match P with T starting with T[6] skipping the sliding window matches for T[3], T[4] and T[5]. So it will miss the pattern match starting at T[3] (i.e., shift s=2). Note that if the match fails because the next T character to be looked at does not match the very first character of P (i.e., i=1 and P[1] is compared with T[s+1]) then the algorithm correctly returns to the while loop to match the following T character T[s+2] with P[1], i.e., in this case step 7 does not make the algorithm incorrect! But if some of P's characters already matched with some of T's characters, i.e., the match fails when i>1 (say, i=4 and P[4] does not match with T[s+4] but P[1], P[2] and P[3] did match with T[s+1], T[s+2] and T[s+3] respectively), then when the algorithm returns to the while loop it should try to match T[s+2] again with P[1] (i.e., slide the window right by one character by incrementing s by 1) but step 7 will slide the window right by i characters instead of one character, resulting in possibly missing an earlier occurrence of P in T as this counterexample shows. For another counterexample, consider P=001, T=0001. When s=0, the loop will stop when i=3, so s will be incremented to 0+3=3. As a result, the algorithm will skip T[2] so it will not find any match even though there is a match with shift 1. Another counterexample is P=abc and T=dcababc.

A counterexample for which the algorithm fails due to step 10 is P=00 and T=0000. There are three matches, at shifts s=0, 1 & 2. Because of step 10, the above algorithm will only detect the matches at s=0 &2 and miss the match at s=1.

Do problem 32.2-1 in the text p. 994 before answering questions 8 and 9.

8. How many spurious hits will there be?
A. 1 B. 2 C. 3 D. 4 E. no spurious hits

9. How many correct matches will there be?
A. 1 B. 2 C. 3 D. 4 E. no correct matches

26 mod 11=4
314159265358979 15 mod 11=4, spurious hit
314159265358979 59 mod 11=4, spurious hit
314159265358979 92 mod 11=4, spurious hit
314159265358979 26 mod 11=4, matching
So there are 3 spurious hits and 1 correct match.

10. Working modulo q=13, how many spurious hits does the Rabin-Karp algorithm encounter in the text T=16152279 when looking for the pattern P=27?
A. 1 B. 2 C. 3 D. 4 E. 0

Disjoint Sets

11. If the array below implements disjoint sets as trees, what is the union being implemented?
The top row are cell indexes.

1	2	3	4	5	6	7	8
-2	6	8	8	1	3	1	-4

A. arbitrary B. by size C. by height

Design a recursive, and a non-recursive algorithm that employs a stack, for the operation **Find(e)** where e is a data element and Find is to be done with path compression. Assume the disjoint set is implemented using array P and union by size. Partial algorithms are given below.

```
        Find-recursive(e, P)
1       if P[e] > 0 then
2               P[e] = Find-recursive(P[e], P); return P[e]________Q 12_________
3       else return e
```

12. What is (are) the correct step (steps) to fill the blank above?

A. Find-recursive(e, P)
B. P[e] = Find-recursive(e, P)
C. return P[e]
D. P[e] = Find-recursive(e, P); return P[e]
E. none of these

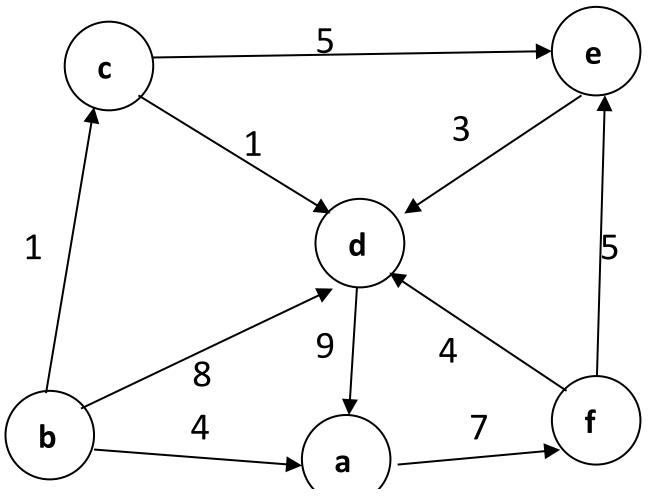
```
Find-iterative(e, P)
S: stack
1 while e is not negative
2       push(e, S)
3       e = P[e]________Q 13_________
4 v = pop(S)
5 while S is not empty
6       temp = pop(S)________Q 14_________
7       P[temp] = v ________Q 15_________
8 return v
```

13. Which step should go to line 3 of Find-iterative?
A. P[temp] = v B. temp = pop(S) C. e = P[e] D. push(e, S) E. none of these

14. Which step should go to line 6 of Find-iterative?
A. P[temp] = v B. temp = pop(S) C. e = P[e] D. push(e, S) E. none of these

15. Which step should go to line 7 of Find-iterative?
A. P[temp] = v B. temp = pop(S) C. e = P[e] D. push(e, S) E. none of these

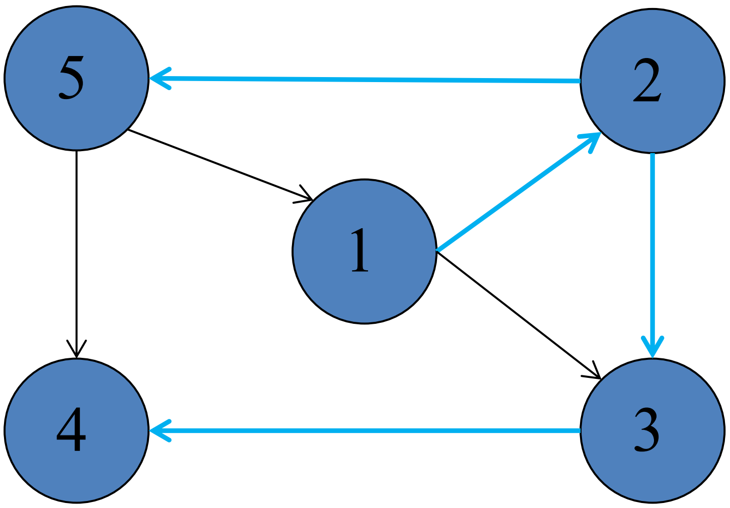
Graph representations and algorithms



16. Simulate algorithm Breadth-First Search (slide 7, slide set 19) on the graph above with “a” as the starting node and determine the d and ∏ values of each node. Assume that adjacent nodes are considered and inserted in the Queue in alphabetical order. What are the d and ∏ values of node “e”?

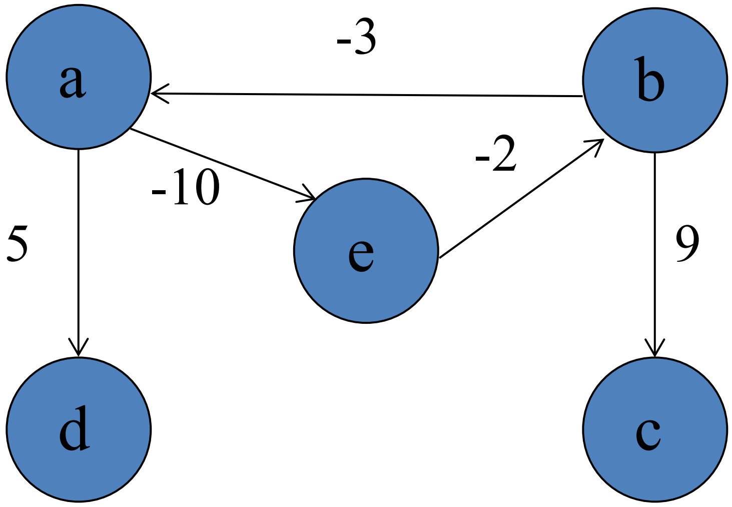
A. 2 & f B. 1 & a C. 0 & NIL D. ∞ & NIL E. none of these

17. If DFS-VISIT started at node 1 and the edges it traversed through recursive calls are shown in blue on the graph to the right, what are the correct classifications of the remaining three edges 5->1, 1->3, and 5->4 respectively?



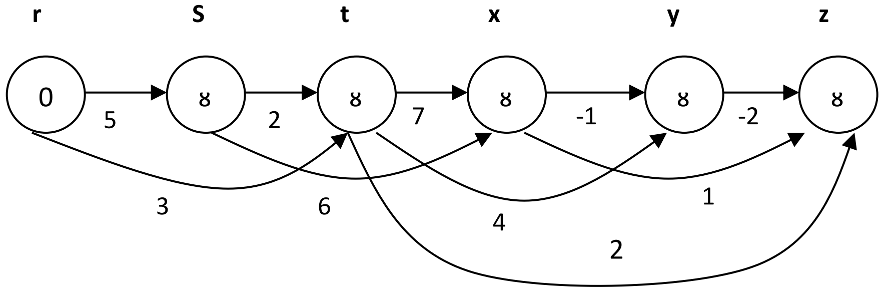
A. Cross, Forward and Back.
B. Forward, Back, and Cross.
C. Cross, Back and Forward.
D. Back, Forward and Cross.
E. none of the above

18. If the Bellman-Ford algorithm is run on the graph below with a negative cost cycle, with “a” as the start node, which edge when considered in step 6 of the algorithm will make the algorithm return false? Assume the for loops in steps 3 & 5 consider the edges in the following order: a->d, a->e, b->c, b->a, e->b.

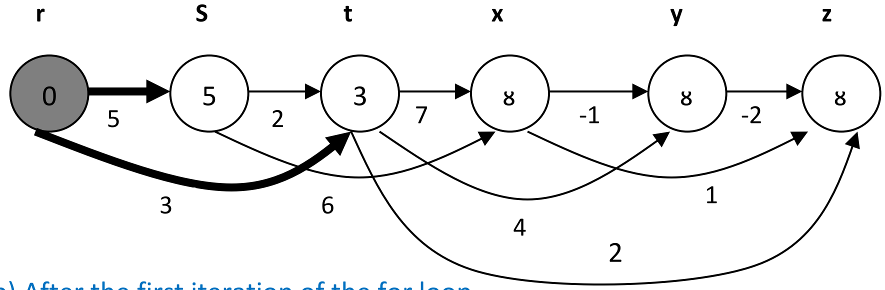


A. edge a->d edge B. a->e C. edge b->c D. edge b->a E. edge e->b

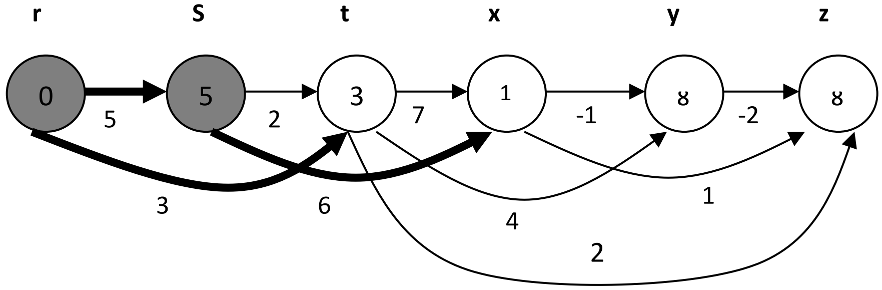
19. Do Problem 24.2-1 on p. 657 of the text. Then answer the following question. The for loop in step 3 of the DAG-SHORTEST-PATHS algorithm on text p. 655 will run six times. **True or False?** The "d" values of none of the six nodes will change during the fourth, fifth and sixth iterations of the for loop.



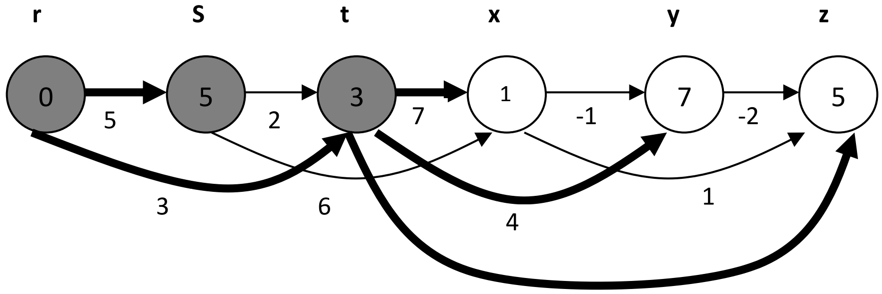
(a) Before the first iteration of the for loop. Source = r



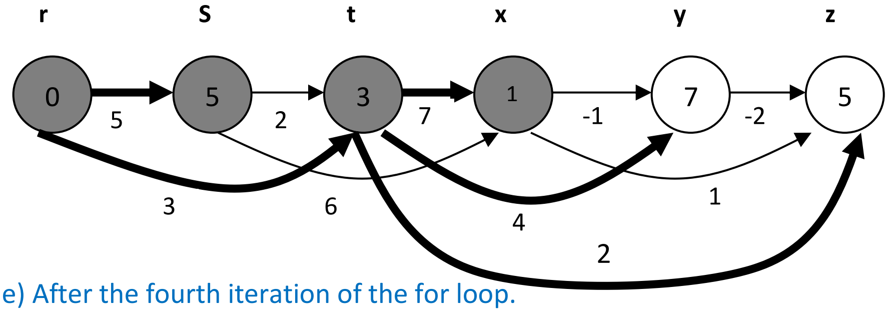
(b) After the first iteration of the for loop.



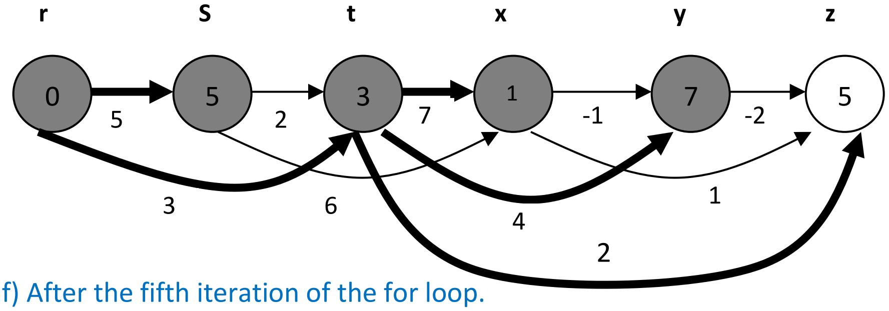
(c) After the second iteration of the for loop.



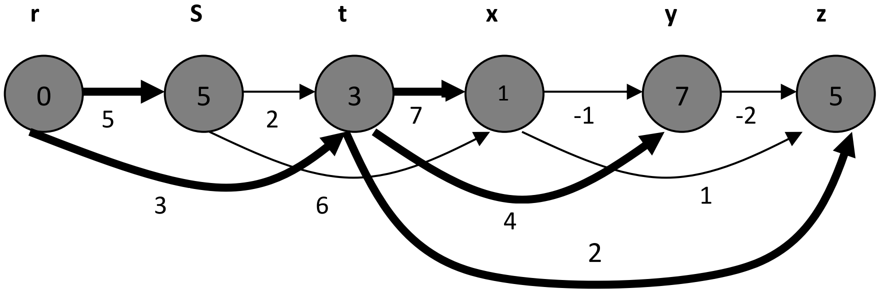
(d) After the third iteration of the for loop.



(e) After the fourth iteration of the for loop.

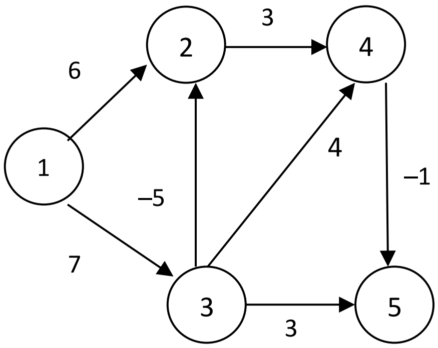


(f) After the fifth iteration of the for loop.



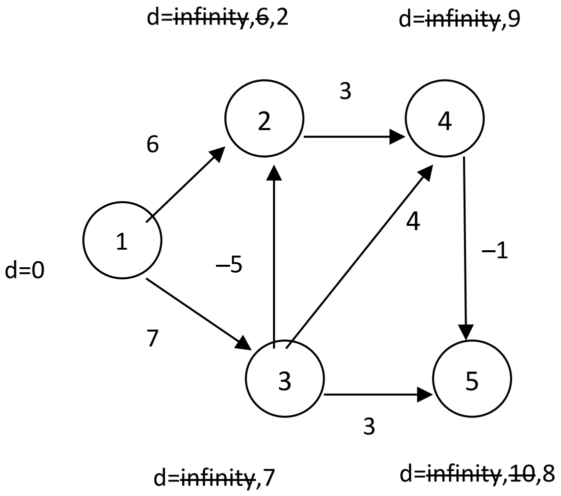
(g) After the sixth iteration of the for loop.

20. Consider the graph below with nodes numbered 1-5 (node numbers are inside the circles) with edge costs as shown. Simulate Dijkstra's algorithm on this graph with 1 as the start node, and answer the following question. The algorithm will fail to find the shortest paths because the graph has negative weight edges. That is, the final “d” values of nodes after the algorithm completes may or may not have the correct costs of the shortest paths to those nodes from the start node 1. The d-values of which nodes are incorrectly computed by the algorithm?



A. nodes 2 & 3 B. nodes 2 & 4 C. nodes 3 & 5 D. nodes 3 & 4 E. nodes 4 & 5

Shortest path to node 4 has a cost of 5 (1-3-2-4), not 9, and to node 5 has a cost of 4 (1-3-2-4-5), and not 8.


d=~~infinity~~,~~6~~,2
d=~~infinity~~,9
3
2
4
6
1
d=0
−5
4
−1
7
3
5
3
d=~~infinity~~,7
d=~~infinity~~,~~10~~,8