COMP 7270 Assignment 4 50 Multiple Choice Questions No late submissions!

Due by 11 AM MONDAY 5.4.15

Instructions:

1. This is an individual assignment. **You should do your own work.**
2. Late submissions will **not** be accepted unless prior permission has been granted or there is a valid and verifiable excuse.
3. Mark answers in a **scantron** sheet.
4. Turn in the sheet to the instructor **in person**.
5. All questions carry 1 point each.

1. If you want to prove that an algorithm is correct, which of the following techniques **CANNOT** be used?
   A. Proof by induction
   B. Proof by contradiction
   C. Proof using loop invariants
   D. Proof by counterexample
   E. none of the above

2. If a recursive divide and conquer algorithm has the recurrence relation $T(n)=aT(n/b) + dn$, what can be said about the algorithm?
   A. Each non-base-case execution will create "a" recursive calls with input size "n/b"
   B. Each non-base-case execution will create "b" recursive calls with input size "n/a"
   C. Each non-base-case execution will create "d" recursive calls with input size "n/d"
   D. Each non-base-case execution will create one recursive call with input size "n/b"
   E. none of the above

3. Suppose a growth function of an algorithm is $T(n) = 1000\log n + 100\sqrt{n} + 3n^2 + n^2\log n + 5$. What is its asymptotic complexity order in the Big-Oh notation?
   A. $O(\sqrt{n})$
   B. $O(n^2)$
   C. $O(\log n)$
   D. $O(n^2\log n)$
   E. none of the above

4. If four algorithms to solve the same problem have the following complexities: $O(n^3)$, $\Omega(n^3)$, $o(n^3)$, $\Theta(n^3)$, what is a reasonable assumption about which is the most efficient (in the absence of any additional information)?
   A. $O(n^3)$ algorithm
   B. $\Omega(n^3)$ algorithm
   C. $o(n^3)$ algorithm
   D. $\Theta(n^3)$ algorithm
   E. all of the above four algorithms are equally efficient

5. Suppose you are asked to write a program to compute the recursive function $f(x) = f(x-1) + f(x-2) + f(x-3)$; $x \geq 3$; $f(0) = f(1) = f(2) = 1$. Which of the following statements is **most** true?
   A. One can write either a recursive or an iterative (looping) algorithm to compute this function
   B. One can only write an iterative algorithm to compute this function
   C. One can write either a recursive or an iterative algorithm to compute this function but the iterative algorithm will be faster
   D. One can only write a recursive algorithm to compute this function
   E. none of the above

6. What is the exact number of times the statement marked with a * in the following pseudocode fragment will be executed (n is a positive integer > 1):

```
i = n
repeat
*       i = i-1
until i = 0
```

A. n B. n-2 C. n+1 D. n-1 E. none of these

The next THREE questions refer to the following algorithm:

```
Mystery(S[i..j]: array of characters)
C: character
1  if S.length ≤ 1 then
2       return
  else
3       C = S[i]
4       Mystery(S[i+1..j])
5       for k=(i+1) to j
6               S[k-1] = S[k]
7       S[j] = C
8       return
```

7. What does this algorithm do?

A. Sorts the characters in the alphabetical order
B. Flips the first and last characters of the array while leaving the other characters where they are
C. Reverses the order of characters in the array
D. Creates a random permutation of the characters in the input array
E. none of the above

8. What are the most accurate recurrence relations of this algorithm (c1, c2, and c3 are constants)?

A. $T(n) = T(n+1) + c1(n-1) + c2$; $T(n=0 \text{ or } 1) = c3$
B. $T(n) = T(n-1) + c1(n^2) + c2$; $T(n=0 \text{ or } 1) = c3$
C. $T(n) = T(n+1) + c1(n+1) + c2$; $T(n=0 \text{ or } 1) = c3$
D. $T(n) = T(n-1) + c1(n) + c2$; $T(n=0 \text{ or } 1) = c3$
E. none of the above

9. What is a reasonable estimate of the order of complexity of this algorithm?

O(nlogn) B. O(1)
C. $O(n^2)$ D. $O(2^n)$
E. none of the above

The next FIVE questions refer to the following recurrences characterizing a recursive algorithm:

$T(n) = T(n-1) + 5$; $T(0) = 1$

10. What is the correct characterization of additional work (i.e., work other than, or excluding, any new recursive calls) done by each execution of this algorithm?

A. $\Theta(n)$ B. $\Theta(\log n)$ C. $\Theta(n \log n)$ D. $\Theta(1)$

E. none of these

11. What is the first step of solving these recurrences using the Backward Substitution method?

A. $T(n-1) = T(n-2) + 5$ so $T(n) = T(n-2) + 10$

B. Make a guess of the solution

C. $T(0)=1$ so $T(1) = T(0) + 5 = 6$

D. Make an inductive hypothesis

E. Apply the Master Theorem

12. Can the Master Theorem be used to solve these recurrences?

Yes (Mark True) No (Mark False)

13. If you were to solve these recurrences using the Substitution Method, with a guessed solution being O(n), what is the Inductive Hypothesis that you would make?

A. $T(n) \leq cn$ for some constant c

B. $T(n+1) \leq c(n+1)$ for some constant c

C. $T(n-1) \leq c(n-1)$ for some constant c

D. $T(n)=cn$ for some constant c

E. none of these

14. If you were to solve these recurrences using the Recursion Tree method, how many levels will the Recursion Tree have? Remember that the root is at level 0.

A. n B. n+1 C. n-1 D. log n E. none of these

15. Consider the problem of finding whether an array A[1…n] of non-negative integers that are already sorted in the ascending order, and which contains no duplicates, has any cell with index i that contains the integer i (i.e., A[i]= i). Which is a **correct** computational strategy to solve this problem?

A. Iterate through the array starting from the first cell going left to right until an index i such that i=A[i] is found (if so, return i and quit) or keep going till the end is reached (then report failure and quit).

B. Iterate through the array starting from the first cell going left to right until an index i such that i=A[i] is found (if so, return i and quit) or a cell with index k is reached where A[k]>k (then report failure and quit).

C. Iterate through the array starting from the last cell going right to left until an index i such that i=A[i] is found (if so, return i and quit) or a cell with index k is reached where A[k]<k (then report failure and quit).

D. Compute the index m of the middle cell of the array; If A[m]==m then return m and quit; If A[m]<m then recursively apply the same strategy to subarray A[m+1…n]; If A[m]>m then recursively apply the same strategy to subarray A[1…m-1]; Base case is when the subarray has only one cell – in that case, if the index of that cell ==its content then return the index and quit, else report failure and quit.

E. All of these strategies are correct.

16. Consider the same problem as in the previous question. Which is the **most efficient** computational strategy from among the four below (ignore whether they are correct or not) to solve this problem?

A. Iterate through the array starting from the first cell going left to right until an index i such that i=A[i] is found (if so, return i and quit) or keep going till the end is reached (then report failure and quit).

B. Iterate through the array starting from the first cell going left to right until an index i such that i=A[i] is found (if so, return i and quit) or a cell with index k is reached where A[k]>k (then report failure and quit).
C. Iterate through the array starting from the last cell going right to left until an index i such that i=A[i] is found (if so, return i and quit) or a cell with index k is reached where A[k]<k (then report failure and quit).
D. Compute the index m of the middle cell of the array; If A[m]==m then return m and quit; If A[m]<m then recursively apply the same strategy to subarray A[m+1…n]; If A[m]>m then recursively apply the same strategy to subarray A[1…m-1]; Base case is when the subarray has only one cell – in that case, if the index of that cell ==its content then return the index and quit, else report failure and quit.
E. All of these strategies are equally efficient.

17. True or False? A dynamic programming algorithm may produce incorrect solutions to some problem instances if the problem does not have optimal substructure.

18. True or False?
(i)A strategy always gives rise to exactly one algorithm
(ii)An algorithm generally represents one strategy to solve a problem
A. Both statements (i) and (ii) are true
B. Both statements (i) and (ii) are false
C. Statement (i) is true and statement (ii) is false
D. Statement (i) is false and statement (ii) is true
E. None of the above hold

19. Suppose q is a problem in NP but outside P. Suppose r is a NP-Complete problem. Explain why, if we show that any instance of r can be transformed to a corresponding instance of q in polynomial time, this means that EVERY problem in NP can then be transformed into q instance by instance.
A. Because r is NP-Complete, so it can be transformed in polynomial time to any and all problems in NP.
B. Because r is NP-Complete, any and all problems in NP can be transformed in polynomial time to it.
C. Because r is NP-Complete, any and all problems in NP can be transformed in polynomial time to it, and because of the polynomial time transformation from r to q, any and all problems in NP can be transformed in polynomial time to q.
D. Because if we show that any instance of r can be transformed to a corresponding instance of q in polynomial time, this means that r and q are one and the same.
E. none of the above.

20. The Vertex Cover Problem (VCP): determine whether a graph has a subset S with k nodes ($k \leq n$, n=total # of nodes in the graph) such that each of its edges is connected to at least one node in this subset. VCP is in NP. Why? Because though no polynomial time algorithm to find a VC of size k in a graph is known, if a graph and a possible VC (a set S of k nodes in the graph, $k \leq n$) are given, one can easily write a polynomial time algorithm to check if those k nodes form a vertex cover or not. Which of the following is a correct strategy for this algorithm? Assume that the graph G is undirected, unweighted and represented as an Adjacency Matrix A, and the possible solution S is provided as a set of nodes.
A. For each node s in the set S, scan the matrix row corresponding to s and when a 1 is encountered in cell A[s,i], replace that with a 0 and also replace the 1 in A[i,s] with a 0. When rows corresponding to all nodes in S have been processed, scan the entire matrix A for any remaining 1's. If there are any 1's left, S is not a correct VC; if there are no 1's left, S is a correct VC.
B. For each node s in the set S, scan the matrix row corresponding to s and when a 0 is encountered in cell A[s,i], replace that with a 1 and also replace the 0 in A[i,s] with a 1. When rows corresponding to all nodes in S have been processed, scan the entire matrix A for any remaining 0's. If there are any 0's left, S is not a correct VC; if there are no 0's left, S is a correct VC.
C. Check if matrix A is symmetric. If it is, then S is a correct VC. If it isn't, S is not a correct VC.

D. For each node s in the set S, scan the matrix row corresponding to s and when a 1 is encountered in cell A[s,i], check if node i is in set S. If i is in set S then S is not a correct VC: report that and quit. If i is not a node in S then continue scanning. If after all rows corresponding to nodes in S have been scanned and the algorithm did not quit, then S is a correct VC.
E. All of the above strategies are correct.

21. If Master Theorem is applied to solve the recurrence $T(n)=2T(n/4)+100n^{1/4}$, which of the three cases applies?
A. Case 1 B. Case 2. C. Case 3. D. Master Theorem does not apply
E. none of the above

22. Which of the following statements is **false** about this computational problem: Move all the zeroes in an array of n numbers to the left end of the array.
A. Every correct algorithm to solve it must execute at least n steps or operations
B. A correct $O(n^2)$ algorithm can be designed to solve it
C. The fastest algorithm to solve this problem has complexity O(n)
D. This problem can be solved by an algorithm in log(n) steps or operations.
E. none of these is false

23. What does the algorithm below compute?
Mystery (x: number, a: non-negative integer)
```
1       temp=1
2        while a>0
3               temp=temp*x
4               a = a ─ 1
5       return temp
```
A. x*a B. a^x C. x^a D. a! E. none of these

24. True or False? A correct and appropriate Loop Invariant to prove the correctness of the algorithm in the previous question is: Before the k-th execution of the while loop in step 2, $temp=x^{k-1}$.

25.
```
Function Mystery(n: non-negative integer)
1               if n ≤ 1 then return 0
2               else return (1+Mystery(n-2))
```
What does Mystery(2) return?
A. 0 B. 4 C. 1 D. 2 E. none of these

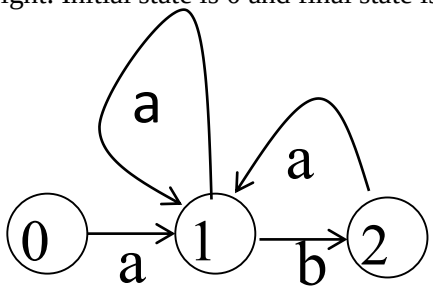
26. What steps of the algorithm in the previous question are executed when the input is not a base case?
A. only step 2 B. part of step 1 and step 2 C. only step 1 D. either step 1 or 2
E. none of these

27. If a recursive algorithm is characterized by the following recurrences, which of the following statements about it are true?
T(n)=T(n-1)+T(n-2)+c
T(0)=T(1)=c
A. Base cases are when input size = 0 and 1
B. When input is not a base case, each execution spawns two recursive executions.
C. These recursive executions reduce input size by 1 and 2 respectively.
D. In addition to spawning new recursive executions, the algorithm does a constant amount of work when the input is not a base case.

E. all of the above statements are true.

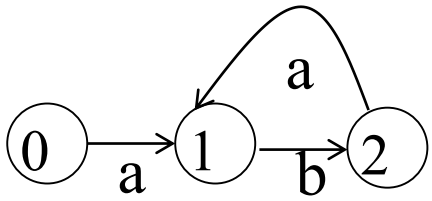
28. The precise meaning of algorithm correctness is:
A. A correct algorithm produces the correct outputs for all inputs.
B. A correct algorithm produces the correct output for any input.
C. A correct algorithm will not get into an infinite loop or infinite recursion for any input.
D. A correct algorithm produces the correct outputs for all valid inputs and halts.
E. None of the above.

29. Consider the String Matching Automaton shown to the right. Initial state is 0 and final state is 2. What is the pattern P it is matching for (or searching for)?
A. aba
B. aab
C. baa
D. ab
E. None of the above



30. True or False? The alphabet Σ for the above String Matching Automaton can contain any number of characters as long as it contains the characters "a" and "b"

31. Consider the Finite State Automaton shown to the right. Initial state is 0 and final state is 2. What is the **most accurate** statement about the patterns it accepts?
A. It accepts only the pattern ab.
B. It accepts the patterns ab and abab.
C. It accepts all patterns aba*****
D. It accepts all patterns (ab)*****
E. It accepts all patterns $(ab)^+$



32. Working modulo q=11, how many spurious hits does the Robin-Karp algorithm encounter in the text T=16152278 when looking for the pattern P=27?
A. 1 B. 2 C. 3 D. 4 E. 5

The following three questions are about the recursive characterization of a dynamic programming solution to some optimization problem, given below:

$$\text{Solution}[i,j],\ 1\le i,j\le n = \begin{cases} 0 \text{ if } i=j \\ \underset{1\le k\le n}{\text{Max}} \ [\text{Solution}[i,k]+1+\text{Solution}[k,j]] \end{cases}$$

33. How many choices does this characterization embody?
A. 1 B. 2 C. k D. n E. none of these

34. Once the algorithm makes a choice, how many subproblems need to be solved corresponding to that choice?
A. 1 B. 2 C. k D. n E. none of these

35. If you were to write a memoized recursive algorithm corresponding to this recursive characterization, what is the size of the memoization table (or array) you will need?
A. nXn B. 1Xn C. 2Xn D. kXn E. none of these

36. True or False? Making the choice of selecting the activity which overlaps with the fewest of the remaining activities is a correct greedy choice for the activity selection problem.

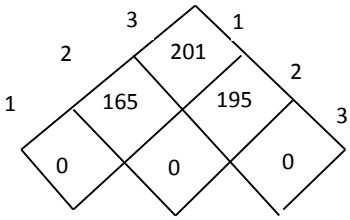
37. True or False? Every problem that a greedy algorithm correctly solves can also be correctly solved by a dynamic programming algorithm.

38. True or False? Every problem that a dynamic programming algorithm correctly solves can also be solved by a greedy algorithm.

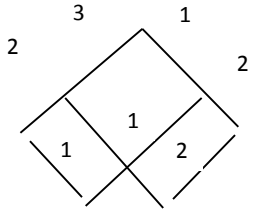
39. True or False? If there is a problem that belongs to both class P and class NP_complete, this implies that P=NP.

The m and s matrices resulting from running the matrix chain multiplication dynamic programming algorithm for a chain of three matrices $A_1A_2A_3$ are shown below.

Matrix m:



Matrix s:



40. What is the minimum number of scalar multiplications needed for computing $A_1A_2A_3$?
A. 165 B. 201 C. 195 D. 0 E. none of these

41. What is the optimal parenthesization?
A. $(A_1A_2A_3)$ B. $(A_1(A_2A_3))$ C. $((A_1A_2)A_3)$
D. $(A_1)(A_2)(A_3)$ E. none of these

The recurrence relations of a recursive algorithm are:
$T(n)=3T(n/4)+2T(n/8)+cn$, $n>8$
$T(n)=c$, $n\leq 8$

42. If you were to draw the recursion tree of this algorithm for an input size of n, how many children will the root node have?
A. 1 B. 2 C. 3 D. 4 E. 5

43. What can you say about the shape of this recursion tree?
A. Every leaf is at a depth of log(n) to the base 4.
B. Every leaf is at a depth of log(n) to the base 8.
C. Every leaf is at a depth of log(n) to the base 2.
D. The depths of the leaves vary between log(n) to the base 4 and log(n) to the base 8.
E. None of the above.

44. Given the matrix chain consisting of 4 matrices <A1, A2, A3, A4>, where the size of each matrix is provided in the table below, what is the optimal number of multiplications needed to multiply these together?

Matrix	Size
A1	5 X 10
A2	10 X 5
A3	5 X 20
A4	20 X 10

A. 1250 B. 1000 C. 750 D. 250 E. none of these

45. Given the following set of symbols and their frequency of appearance in a document, suppose we design an optimal variable length binary code for these symbols so that the file size of the document can be minimized.

Symbol	Λ	Γ	Π	Δ	Θ	Σ	Ω
Frequency (thousands)	2	5	15	3	7	1	8

What variable length code(s) might be produced by the Huffman Coding algorithm?

I.

Symbol	Λ	Γ	Π	Δ	Θ	Σ	Ω
Length of the code (in bits)	5	3	1	4	3	5	3

II.

Symbol	Λ	Γ	Π	Δ	Θ	Σ	Ω
Length of the code (in bits)	6	4	1	5	3	6	2

III.

Symbol	Λ	Γ	Π	Δ	Θ	Σ	Ω
Length of the code (in bits)	5	3	2	4	2	5	2

A. I or III
B. I or II
C. II or III
D. I or II or III
E. none of these

46. The l matrix computed by a DP algorithm to solve a 3 line Assembly Line Scheduling problem is given below. Suppose l*=3. What is the optimal path through the lines for stations 1-6?

j	2	3	4	5	6
$l_1[j]$	1	2	1	3	2
$l_2[j]$	1	2	1	3	2
$l_3[j]$	1	2	3	1	2

A. 1-2-1-3-2-3 B. 1-2-3-3-2-3 C. 1-2-3-1-2-3 D. 1-2-3-2-3-3
E. none of these

47. True or False? The DP bottom up table lookup algorithm for finding the LCS of two strings can be used as is, i.e., without any modification to the algorithm's steps, to solve the following computational problem: finding the longest palindrome that is a subsequence of the given input string. I.e., if the input string is "character" the answer is "carac."

Next two questions are about the following algorithm:

```
Mystery(A[p...r] of numbers)
1   x = A[r]
2   i  = p – 1
3   for j = p to r -1
4         if A[j] ≤ x
5                   i = i + 1
6                   swap A[i] and A[j]
7   swap A[i +1] and A[r]
8   return  i +1
```

48. If input to the algorithm is A= [10, 1, 9, 2, 8, 3], what will be the contents of A after Mystery finishes executing?

A. [10, 9, 8, 3, 2, 1]
B. [3, 8, 2, 9, 1, 10]
C. [1, 2, 3, 10, 8, 9]
D. [1, 2, 3, 8, 9, 10]
E. none of the above

49. This algorithm can be modified to solve a different problem - moving any and all zeroes in the array A to its left end. This can be accomplished by either deleting or changing some of its steps (i.e., no new steps need to be added). Which steps (lines) will have to be deleted or modified?

A. 1,3,4,7,8
B. 1,2,3,4,5,6,7,8
C. 2,5,6
D. 2,3,4,8
E. none of the above

Modified steps shown below in red

~~1 x = A[r]~~
2 i = p – 1
3 for j = p to r
4   if A[j] ==0
5   i = i + 1
6   swap A[i] and A[j]
~~7 swap A[i +1] and A[r]~~
8 ~~return i +1~~ OR return A

50. Suppose x is a suffix of y. This implies that $\sigma(x) \leq \sigma(y)$.
True or False? If |y|>|x| and $\sigma(y) > \sigma(x)$, then |x|≥|P|.
If |y|>|x| and $\sigma(y) > \sigma(x)$, this means that a suffix of y that is longer than x is a prefix of P. If P has a prefix that is longer than x then |x|<|P|.