****

**Sample Question Format**

**KIIT Deemed to be University**

**Online Mid Semester Examination(Spring Semester-2021)**

**Subject Name & Code:** DAA/ CS2012 **Applicable to Courses: B. Tech**

**Full Marks=20** **Time:1 Hour**

**SECTION-A(Answer All Questions. All questions carry 2 Marks)**

**Time:20 Minutes (5×2=10 Marks)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Question No** | **Question Type(MCQ/SAT)** | **Question** | **Answer Key(if MCQ)** | **CO Mapping** |
| **Q.No:1(a)** |  | Let f(n) = 200n+7 and g(n) = 100n + 15 be asymptotically positive functions and h(n) be a third, unknown asymptotically positive functions function. Which of the following is not True.   1. h(n) is O(f(n)) and h(n) is also O(g(n)) 2. h(n) is O(f(n)) but h(n) is not O(g(n)) 3. h(n) is O(g(n)) but h(n) is not O(f(n)) 4. h(n) is not O(f(n)) and h(n) is also not O(g(n))   E. None | B |  |
|  |  | Let f(n) = 3^(2^n), g(n) =3^(n^2), h(n)=n^(n^2) be asymptotically positive functions.Which of the following is true? (^ represents to the power)  A. f(n)=(g(n))  B. f(n)=Θ(g(n))  C. h(n)=O(g(n))  D. g(n)=f(n))  E. None | A |  |
|  |  | Let f(n) = 2n2 + 4n + 5 and g(n) = 7n2 + 5n + 12 be asymptotically positive functions. Let h(n) be a third unknown asymptotically positive function. Which of the following is not possible.  A. h(n) is O(f(n)) and h(n) is also O(g(n))  B. h(n) is O(f(n)) but h(n) is not O(g(n))  C. h(n) is O(g(n)) but h(n) is not O(f(n))  D. h(n) is not O(f(n)) and h(n) is also not O(g(n))  E. None | C |  |
|  |  | Let f (n) be asymptotically positive function then choose the correct option by considering the following statements  I. f (n) + O(f (n)) = Θ(f (n))  II. f (n) + Ω(f (n)) =Θ(f (n))  A. Both I and II are always true.  B. I may not be true but II is always true  C. I is always true but II may not be true  D. Both I and II are false.  E. None | C |  |
| **Q.No:1(b)** |  | A image Processing application begins with an array A of size n. The first phase of preprocessing the inputs takes O(n) steps the array A. The second step involves a processing of A to yield a new array C in time O(n log n ). What is the most accurate and concise description of the complexity of the overall algorithm?     1. O(n2 log n) 2. O(n) 3. O(n log n+ n) 4. O(n log n) | D |  |
|  |  | What would be the time complexity of Randomized-Quick Sort  when applied on an sorted array of even numbers < 2, 4, 6, … n-1, n >  A. O(n)  B. O(n2)  C. O(nlogn)  D. O(logn)  E. None | C |  |
|  |  | Suppose maximum number of comparisons required to find any element in a given array of length n is 5. Each distinct element occurs equal number of times in that array. Determine the number of distinct elements present in that array.  A. 8  B. 16  C. 32  D. 64  E. None | C |  |
|  |  | Let the recurrence T(n) be defined as T(n) = a T(n/b) + 1, where a >=1 and b > 1.  Which of the following combinations of a & b will produce a complexity of Θ(logn)?  A. a=2, b = 2  B. a = 1, b = 4/5  C. a = 3, b = 3/4  D. a = 1, b = 7/6  E. None | D |  |
| **Q.No:1(c)** |  | Consider the following Code Segment  int fun ()  {  int i, C = 0;  for (i = 1; i ≤ 2048; i =4\*i)  C = C+i;  return C;  }  What is the time complexity of the above function fun?  A. Θ(nlog n)  B. Θ(log n)  C. Θ(n)  D. Θ(1) | D |  |
|  |  | Consider the following Code Segment  int fun(int n)  {  int i = 1, S = 1;  while (S ≤ n)  {  i++;  S = S + i;  }  return S;  }  What is the complexity of the above function fun?  A O(n)  B O(√n)  C Θ(log n)  D Θ(n2) | O(√n) |  |
|  |  | Consider the following Code Segment  int fun(int n)  {  int p, q, r = 0;  for (p = n; p ≥ 1; p=p/2)  for (q = 1; q ≤ p; q++)  r = r + 1/n;  return r;  }  What is the returned value of the above function fun?  A Θ(1)  B Θ(log n)  C Θ(n)  D Θ(nlog n) | A |  |
|  |  | Consider the following Code Segment  int fun(intn, int A[])  {  int i, j, s=0;  for(i = 1; i<= n; i++)  {  for(j=1; j<n; j += i)  {s = s + A[i];}  }  retun s;  }  What is the least upper bound time complexity of the function fun?  A. O(n)  B. O(nlog n)  C. O(n2)  D. O(2n)  E. None | B |  |
| **Q.No:1(d)** |  | What will be the final content of the Max-Heap(A) if key value at node index 7 is increased to 15, where A[1] represents root of the Max-Heap(A).  A={20, 11, 9, 6, 8, 5, 4, 3, 2}.  A. <20, 11, 9, 6, 8, 5, 15, 3, 2 >  B. < 20, 15, 9, 11, 8, 5, 4, 6, 2 >  C. <20, 11, 15, 6, 8, 5, 9, 3, 2 >  D. None | C |  |
|  |  | What will be the final content of the Max-Heap(A) if key value at node index 8 is increased to 15, where A[1] represents root of the Max-Heap(A).  A={18, 10, 9, 7, 8, 5, 4, 3, 2}.  A. <18, 10, 9, 15, 8, 5, 4, 7, 2 >  B. < 18, 15, 9, 10, 8, 5, 4, 7, 2 >  C. <18, 15, 9, 8, 10, 5, 9, 7, 2 >  D. None | B |  |
|  |  | What will be the final content of the Max-Heap(A) if key value at node index 9 is increased to 15, where A[1] represents root of the Max-Heap(A).  A={18, 10, 9, 7, 8, 5, 4, 3, 2}.  A. < 18, 15, 9, 10, 8, 5, 4, 7, 2 >  B < 18, 10, 9, 15, 8, 5, 4, 3, 7>  C. < 18, 15, 9, 10, 8, 5, 4, 3, 7>  D. None | C |  |
|  |  | What will be the final content of the Max-Heap(A) if key value at node index 5 is increased to 10, where A[1] represents root of the Max-Heap(A).  A={20, 11, 9, 6, 8, 5, 4, 3, 2}.  A. <20, 11, 9, 6, 10, 5, 4, 3, 2 >  B. < 20, 10, 9, 6,11, 5, 4, 3, 2 >  C. <20, 11, 9, 10, 6, 5, 4, 3, 2 >  D. None | A |  |
| **Q.No:1(e)** |  | Minimum number of comparison required to merge two sorted files of size p and q is MIN(p, q). How many minimum comparisons are required to merge 3 sorted files of size p, q, and r, where p>q>r?  A. Minimum number comparisons is q if q = p - r  B. Minimum number comparisons is p if p = q + r  C. Minimum number comparisons is r if r = p - q  D. None | B |  |
|  |  | Minimum number of comparison required to merge two sorted files of size p and q is MIN(p, q). How many minimum comparison are required to merge 3 sorted files of size p, q, and r, where p>q>r?  A. Minimum number comparisons is q if q = p - r  B. Minimum number comparisons is q or r if p = q + r  C. Minimum number comparisons is p if r = p - q  D. None | C |  |
|  |  | Minimum number of comparison required to merge two sorted files of size p and q is MIN(p, q). How many minimum comparison are required to merge 3 sorted files of size p, q, and r, where p>q>r?  A. Minimum number comparisons is p if q = p - r  B. Minimum number comparisons is r if p = q + r  C. Minimum number comparisons is q if r = p - q  D. None | A |  |
|  |  | Minimum number of comparison required to merge two sorted files of size p and q is MIN(p, q). How many minimum comparison are required to merge 3 sorted files of size p, q, and r, where p>q>r?  A. Minimum number comparisons is q if q = p - r  B. Minimum number comparisons is r if p = q + r  C. Minimum number comparisons is p if r = p - q  D. None | C |  |

**SECTION-B(Answer Any One Question. Each Question carries 10 Marks)**

**Time: 30 Minutes** **(3×10=30 Marks)**

|  |  |  |
| --- | --- | --- |
| **Question No** | **Question** | **CO Mapping** |
| **Q.No:2** | Write the procedure for partition of Quick-Sort, where Partition always splits the array into 2 : 8 ratio. Write the recurrence of the above Quick-Sort and solve the above equation using recursion tree method. |  |
| **Q.No:3** | Solve using Master theorem. Justify weather Master theorem is applicable.  (a) T(n) = 16T( n/4) + n2  (b) T(n) = 3T( n/2) + n log log n  (c) T(n) = 4T( n/2) + n3log n |  |
| **Q.No:4** | Consider the following instance to Construct a Max-Priority Queue from your own Roll No.  **For Example ::** if my roll no is 2028010, So my array elements A < 2, 0, 2, 8, 0, 1, 0 >. Now construct a Max-Priority Queue from this array step by step .  After constructing the Max-Priority Queue from your roll no as explained above, then replace **ZERO (0) with 7**, and **replace ONE (1) with 6** while maintaining properties of the Max-Priority Queue.  **For Example**, If My Max-Priority Queue obtained from my Roll no is < 8, 2, 2, 1, 0, 0, 0 > then replace 0 with 7 & 1 with 6 respectively, while maintaining properties of the Max-Priority Queue.  Write the procedure for Max-Priority Queue and find its complexity. |  |
| **Q.No:5** | (i) Prove that maximum height of a complete binary tree with ‘n’ number nodes is bounded by O(log2 n)  (ii) Given an unsorted array A[1..n], where odd indexed elements are sorted in ascending order and the even indexed elements are sorted in descending order. Design an algorithm to sort the array in O(n) worst case time in ascending order. |  |
| **Q.No:6** | Consider the following instance of fractional knapsack problem:  (**Note** :- ***Find out weights of items from your own roll number as mentioned below***)  **Weight**:- Each digit of your **roll number** is an individual weight of an item. If roll number contains 0 and 1, then replace **ZERO (0) with 3**, and **replace ONE (1) with 4**.  **For example**, if my roll is 2028010, then after replacing 0 & 1 with 3 & 4 respectively, it becomes 2328343.  So my weight vector of items are W < 2,3,2,8,3,4,3 >.  Knapsack capacity(K) = 16, and Profit(P) = < 10,9,8,32,12,15,10 >.  (Note:- *The knapsack capacity and profit is common for all as given in this question.* *For decimal value take up-to one decimal place*)  Solve the given knapsack problem by applying Greedy Strategy. |  |
| **Q. No:7** | Suppose we are given with an integer array A[1..n] and there exists an element A[i] such that a[1] ≥ a[2] ≥ .. ≥ A[i-1] ≥ A[i] ≤ A[i+1] ≤ .. ≤ A[n]. Write an efficient algorithm that finds an element x in a[1] ≥ a[2] ≥ .. ≥ A[i-1] ≥ A[i] and an element y in A[i] ≤ A[i+1] ≤ ... ≤ A[n] such that z = x + y, where z is a given integer. |  |

**Controller of Examinations**