|  |  |  |
| --- | --- | --- |
| **PES University Logo.jpg** | **PES University, Bangalore**  (Established under Karnataka Act No. 16 of 2013) | UE18CS251 |
| **MAY 2020: END SEMESTER ASSESSMENT (ESA) B.TECH.**  UE18CS251 **– Design and Analysis of Algorithms** | | |
| Time: 3 Hrs Answer all questions in the same order Max Marks: 100 | | |
|  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | a) | Solve by backward substitution method: C(n) = C(n/2) + C(n/2) + 1 for n > 1, C(1) = 0.  Assume n=2k and the recurrence turns out to be 2C(2k-1) +1  By solving we get 2k -1=n-1 =O(n) | | 4 |
| b) | Answer the following Questions:   1. finds the minimum distance between the elements 2marks 2. Time efficiency is O(n^2) 1 mark 3. Change the inner loop to run faster: 3 marks   Algo what(a[]){  a ←∞  for i ← 0 to n − 2 do  for j ← i +1 to n − 1 do  temp ← |A[i] − A[j]|  if temp < a  a ← temp  return a  } | | 6 |
| c) | Indicate order of growth and prove your assertion Each carries 2 marks   1. 2n lg(n + 2)2 + (n + 2)2 lg n/2   Solution: 2n2 lg(n + 2) + (n + 2)2(lg n − 1) ∈  Θ(n lg n)+Θ(n2 lg n) = Θ(n2 lg n).   1. 3*n*2 log *n =* Θ(n2 lg n). 2. 5*n*+1 + 3*n*−1 = Θ(5n). | | 6 |
|  | d) | prove that if t1(n) ∈ Ω(g1(n)) and t2(n) ∈ Ω(g2(n)), then  t1(n) + t2(n) ∈ Ω(max{g1(n), g2(n)}).  Since t1(n) ∈ Ω(g1(n)), there exist some positive constant c1  and some nonnegative integer n1 such that  t1(n) ≥ c1g1(n) for all n ≥ n1.  Since t2(n) ∈ Ω(g2(n)), there exist some positive constant c2 and some  nonnegative integer n2 such that  t2(n) ≥ c2g2(n) for all n ≥ n2.  Let us denote c = min{c1, c2} and consider n ≥ max{n1, n2} so that we  can use both inequalities. Adding the two inequalities above yields the  following:  t1(n) + t2(n) ≥ c1g1(n) + c2g2(n)  ≥ cg1(n) + cg2(n) = c[g1(n) + g2(n)]  ≥ cmax{g1(n), g2(n)}.  Hence t1(n) + t2(n) ∈ Ω(max{g1(n), g2(n)}), with the constants c and  n0 required by the O definition being min{c1, c2} and max{n1, n2}, respectively. | | 4 |
|  | | | | |
| 2. | a) | The size of the array is 2n and hence  Initialize i=n and j=n+1  While(i>1 and j<2n)  Swap(a[i], a[j])  i=i-2, j=j+2 | | 6 |
| b) | int binarySearch(int arr[], int l, int r, int x)  {    while (l <= r)    {      int m = l + (r-l)/2;        if (arr[m] == x)          return m;        if (arr[m] < x)          l = m + 1;        else           r = m - 1;    }      return -1;  } | | 4 |
| c) | Merge Sort Time Analysis  Best case time efficiency T(n) = 2T(n/2)+n/2  And efficiency turns out to be O(nlogn)----------------(2 marks)  Worst case time efficiency T(n)=2T(n/2)+n  using master theorm or using backward substitution  Efficiency turns out to be O(nlogn)----------------------(3 marks) | | 5 |
| d) | int partition(int \*a, int l, int r)  {  int key, i, j,t;  key=a[l];  i=l;  j=r+1;  while(1)  {  i++;  while((a[i]<key)&&(i<=r))  i++;  j--;  while((a[j]>key)&&(j>=l))  j--;  if(i<j)  {  t=a[i];  a[i]=a[j];  a[j]=t;  }  else  break;  }  a[l]=a[j];  a[j]=key;  return j;  } | | 5 |
|  | | | | |
| 3. | a) | Listing the variants   1. Decrease by Constant 2. Decrease by constant factor 3. Variable size decrease-------------------2 marks   Examples : atleast one example each ----------3 marks | | 5 |
| b) | Topological-sort(G){  1.Call dfs all vertices on G to construct dfs forest  2. if G contains a back edge report error  3. else, as traversal of each vertex push in to the stack  4.return ordering  } ------- 3 marks for listing the steps    and topological ordering =A,C,D,B,E,H,F,G ---4 marks | | 7 |
| c) | 10,14,65,23,80,12,34   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 10 | 14 | 65 | 23 | 80 | 12 | 34 |   Using bottom up heap construction  Step 1: 65 last parent is heapyfied  14 is replaced with 80  10 is replaced with 80 and placed at 23rd position   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 80 | 23 | 65 | 10 | 14 | 12 | 34 |   80 is replaced with 34 as part of heap sorting   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 34 | 23 | 65 | 10 | 14 | 12 | 80 |     Step 2:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 65 | 23 | 34 | 10 | 14 | 12 | 80 |   65 is replace with 12   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 12 | 23 | 34 | 10 | 14 | 65 | 80 |   Step 3:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 34 | 23 | 14 | 10 | 12 | 65 | 80 |   34 is replaced with 12   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 12 | 23 | 14 | 10 | 34 | 65 | 80 |   Step 4:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 23 | 12 | 14 | 10 | 34 | 65 | 80 |   23 is replaced with 10   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 10 | 12 | 14 | 23 | 34 | 65 | 80 |     Step 5:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Array[] | 14 | 12 | 10 | 23 | 34 | 65 | 80 | |  | 10 | 12 | 14 | 23 | 34 | 65 | 80 |     Step 6:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  | 12 | 10 | 14 | 23 | 34 | 65 | 80 |      |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  | 10 | 12 | 14 | 23 | 34 | 65 | 80 |   Hence sorted-------6 marks  Heap sort is stable and worst case time complexity is O(nlogn) ----- 2 marks | | 8 |
|  | | | | |
| 4. | a) | Implement a function to sort integers using Distribution Counting  C:\Users\User\Desktop\d.png | | 4 |
| b) | Pattern: RECENT and Text: REGENTRELENTRESENTRODENTRECENTLY  Bad symbol table: Good Suffix Table  R -5 K=1 RECENT D2=6  E-2 K=2 D2=6  C-3 K=3 D2=6  N-1 K=4 D2=6  For all other characters 6 K=5 D2=6  REGENTRELENTRESENTRODENTRECENTLY  RECENT 3 characters matched G was a mismatch D1=(max(6-3,1))=3 D2=6 so Shift by 6  RECENT 3 characters matched L was a mismatch D1=3, D2=6 Shift by 6  RECENT 3 characters matched S was a mismatch D1=3,D2=6  RECENT D1=3, D2=6 Shift by 6  RECENT an exact match  Total number of shifts=4  Total number of comparisons = 22  2 marks for the table, 2 marks for comparison, two for the result | | 6 |
| c) |  | | 5 |
| d) | Encode the text DAD\_ADDED\_BED using Huffman coding and probabilities are  A=11, B=6,C=2,D=10,E=7,\_=10  Huffman tree Coding: A=10, B=1111, C=1110, D=01, E=110, \_=00  Coding for the above text: 011001001001011100100111111001  Tree 4 marks, coding 1 marks | | 5 |
|  | | | | |
| 5. | a) | 1. Trivial Lower Bounds  2. Information-Theoretic Arguments  3. Adversary Arguments  4. Problem Reduction  Any two with simple Examples | 8 | |
| b) | W=8 is the capacity of the knapsack, weights and values of items are w={2,3,4,5} V={1,2,5,6}.   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | | 0 | 0 | 1 | 2 | 5 | 5 | 6 | 7 | 7 | | 0 | 0 | 1 | 2 | 5 | 6 | 6 | 7 | **8** |   8 is the profit and selected objects are 2nd and 4th and hence profit is 8. | 6 | |
| c) | Define the following with an example: i) Class P ii) Class NP iii) NP-Complete.  Definition carries 3 marks  Examples carries 3 marks | 6 | |