|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | |  | |  | **Reg. No.:** | | |  | |
|  | |  | **Name :** | | |  | |
|  | See the source image | | | | | | | | | | |  | |
|  | **Continuous Assessment Test 2(CAT 2) – Oct 2022** | | | | | | | | | | |  | |
| Programme | | | | : | **B.Tech. CSE** | | Semester | | | : | **Fall Semester 2022-23** | | | |
| Course Code | | | | : | **CSE4001** | | Class Nbr(s) | | | : | **CH2022231000229**  **CH2022231000223**  **CH2022231000226**  **CH2022231000220**  **CH2022231000217** | | | |
| Course Title | | | | : | **Parallel & Distributed Computing** | |
| Faculty(s) | | | | : | **M.Sivagami, Harini.S, Kumar.R**  **Ayeesha S.K, VenkatRaman.S** | | Slot | | | : | **D2** | | | |
| Time | | | | : | **90 Minutes** | | Max. Marks | | | : | **50** | | | |
| **Answer all the Questions** | | | | | | | | | | | | | |
| Q. No. | | Sub- division | Question Text | | | | | | | | | | Marks | |
| 1. | | a.  b. | For the Task-dependency graph given in the below Figure 1, determine the following   1. Maximum degree of concurrency at each level (1M) 2. Critical path (1M) 3. Critical path length (2M) 4. Average degree of concurrency of a graph (2M)   Figure 1. Task Dependency Graph  **Solution:**  **a) Level 0 – 6 , Level 2 - 2, Level 3 – 1 (1 mark)**  **b) Task 6 -> Task 8 -> Task9 (1 mark)**  **c) 30 (2 marks)**  **d) 3.3 (2 marks)**  The distribution of work among processes as a result of the static mapping technique may be severely unbalanced. How can you alleviate this problem? In the context of parallel computing, suggest a strategy to deal with uneven work distribution. Detail the reasons for your response.  **Solution**:  **Distributed dynamic load balancing scheme, the set of executable tasks are distributed among processes that exchange task at run time to balance work.**  **\*\* Critical parameters\*\*\*\***  **How are the sending and receiving processes paired together?**  **How much work is transferred in each exchange ?**  **When is the work transfer performed ?** | | | | | | | | | | 6  4 | |
| 2. | | a.  b. | The file **description.txt** has results of Alzheimer’s Test along with serial numbers of 10 lakhs elderly people. The data format in the file as serial number, P or N ( Positive – P or Negative -N) of Alzheimer’s test results . Serial number acts a patient’s id. Now to use the memory efficiently the data entry operator would like to store the data as such (value, starting position, run length). For your better understanding the sample file content and the corresponding compressed data format are given below.  **Sample file content compressed data format (value, starting position, run length)**  **(Serial No., result) (result is represented as value)**  **1 P P 1 3**  **2 P N 4 3**  **3 P P 7 2**  **4 N N 9 2**  **5 N**  **6 N**  **7 P**  **8 P**  **9 N**  **10 N**  You are asked to compress the file content as mentioned above and display the same to the end user using distributed memory programming. (10 Marks)  Write the source code to display the second highest and lowest digit from a given number of ‘n’ digits. ‘n’ is varied and should not be passed as an argument from client to server. Assume that the client passes the number alone and the server should display the 2nd highest and lowest digit of the given number. Write the appropriate program for this scenario. (5 Marks)  Example:  **Given number: 53214**  **2nd Highest and lowest digits: 2 and 4**  **Solution:**   * **Client Code (1 mark)**   public class MyClient{  public static void main(String args[]){  try{  num=12345;  comp stub=(comp)Naming.lookup("rmi://localhost:5000/sonoo");  System.out.println(stub.cval(num));  }catch(Exception e){}  }  }   * **Server code (3 marks)**   import java.rmi.\*;  import java.rmi.server.\*;  public class elecRemote extends UnicastRemoteObject implements comp{compRemote()throws RemoteException{  super();  }  public int cval(int num)  {  Max=8;  Min=1;  while(num != 0){  int rightDigit = num % 10;  num /= 10;  if(rightDigit > max)  {secondmax=max  max=rightDigit;  }  if(rightDigit < min)  {secondmax=min  min=rightDigit;  }  }  Print secondmax and min  }}  import java.rmi.\*;  import java.rmi.registry.\*;  public class MyServer{  public static void main(String args[]){  try{  comp stub=new compRemote();  Naming.rebind("rmi://localhost:5000/sonoo",stub);  }catch(Exception e){System.out.println(e);}  }  }   * **Interface (1 mark)**   import java.rmi.\*;  public interface elec extends Remote  {  public int cval(double val, int i)throws RemoteException;  } | | | | | | | | | | 15 | |
| 3. | |  | Consider two data centers, DC1 and DC 2. Let DC1 and DC2 be located at the VIT campuses in Vellore and Chennai, respectively. For DC1 and DC2, answer the following questions and justify them in detail with proper illustration as required   1. Is it feasible to provide both DC1 and DC2 common global physical clock? (2 marks) 2. Distinguish the keywords distributed systems and distributed computing with respect to DC1 and DC2 (2 marks) 3. If a node failure occurs in DC1, can DC2 offer any support? (3 marks) 4. Assume the Capstone project will be reviewed for each student by both campus faculty. 50% marks will be updated by the Chennai campus faculty and the remaining by the Vellore counterpart. This mark is available as a single array of 1-dimension of size ‘n’ where ‘student’s student strength. How do faculty from both campuses, update the mark simultaneously? (3 marks)   **Solution:**   * **A) No (1 mark) Issues: skew, need for external synch (1 mark)** * **B)System includes network, network comp, processing components etc**   **Computing includes system software components, cluster softwares, RJMS, application softeare (1 mark each)**   * **C)Yes.(1 mark) Explanat of Fault tolerance (2 marks)** * **Here mark is a common entity that will be updated in distributed fashion. (1 mark)**   **Use of directory coherence / DSM to resolve the given issue (2 marks)** | | | | | | | | | | 10 | |
| 4. | | a)  b) | Considering two clocks C1 and C2 that are drifting at the same rate but in opposite direction. Let ‘d’ (any system differs no more than by d time units) and ‘2r’ (twice the round-trip time to the global time server) be 20 seconds and 5 seconds respectively. How frequently will the clocks C1 and C2, need to be synchronized (from a global time server), if they use the pulling server algorithm? Illustrate your answer with a neat table showing the synchronized values in the above system for five iterations. Assume the clocks starts at 10:00:00 AM.  Solution:   * In a normal system, it will be d/ᵨ). Since in opposite direction, it will be d/ d/ᵨ). (2 marks){this frequently systems has to be synchronized) * Synch. Value at client will be Time Stamp + roundtrip time /2 🡪 (5/4) * Here RTT=5/4 = 0.125  |  |  | | --- | --- | | Iteration | Synch. Time Value | | 1 | 10.00.0125 | | 2 | 10.00.0250 | | 3 | 10.00.0375 | | 4 | 10.00.0400 | | 5 | 10.00.0525 |     Figure 3. Events in a system with 6 distributed processes  List the vector clock time stamps for each event shown in the above Figure 3. (7 marks)  List the happened before relationship and concurrency events with justification (3 marks)  **Solution:**    **Happened B4 Relations:**  **a🡪b;c🡪d;e🡪f;g🡪h;x🡪l;q🡪r;p🡪0;k🡪s;i🡪h; u🡪v;t🡪w (minimum any five with proper vector comparison, 1.5 marks)**  **Concurrent: (minimum any three with proper justification 1.5 marks)**  **a||c||e**  **q||s||l** | | | | | | | | | | 5  10 | |
|  | |  | Total | | | | | | | | | | 50 | |
|  | ⇔⇔⇔ | | | | | | | | | | | | |