**Assignment for all Interns**

**C Programming Assignment**

|  | Write a menu driven program that depicts the working of a library. The menu  options should be:  1. Add book information  2. Display book information  3. List all books of given author  4. List the title of specified book  5. List the count of books in the library  6. List the books in the order of accession number  7. Exit  Create a structure called library to hold accession number, title of the book, author  name, price of the book, and flag indicating whether the book is issued or not. |
| --- | --- |
| 2. | Write a function that receives marks received by a student in 3 subjects  and returns the total and percentage of these marks. Call this function from  main( ) and print the results in main( ). |
| 3. | Search and Sequence  Consider an integer array ‘a’ of size 10, where the value of the array elements are  {1,5,4,8,9,2,0,6,11,7}. Write a program to  ▪ Find whether the given element exists in an array or not. If the element exists in an  array, display YES else NO.  ▪ To print a number following a sequence of elements in an array i.e., 15489206117. |

**Data Structure Assignment**

| 1. | In-place merge two sorted arrays. Given two sorted arrays, X[ ] and Y[ ] of size m and n each, merge elements of X[ ] with elements of array Y[ ] by maintaining the sorted order, i.e., fill X[ ] with the first m smallest elements and fill Y[ ] with remaining elements. For example,  **Input:**    X[] = { 1, 4, 7, 8, 10 }  Y[] = { 2, 3, 9 }    **Output:**    X[] = { 1, 2, 3, 4, 7 }  Y[] = { 8, 9, 10 } |
| --- | --- |
| 2. | Generate binary numbers between 1 to `n` using a queue. Given a positive number n, efficiently generate binary numbers between 1 and n using the queue data structure in linear time.  For example, for n = 16, the binary numbers are:  **Input :**  1 10 11 100 101 110 111 1000 1001 1010 1011 1100 1101 1110 1111 10000  **Output:**  1 10 11 100 101 110 111 1000 1001 1010 1011 1100 1101 1110 1111 10000 |
| 3. | Print all nodes of a perfect binary tree in a specific order.  Given a perfect binary tree, print the values of alternating left and right nodes for each level in a top-down and bottom-up manner.  For example, there are two ways to print the following tree:  Variation 1: Print Top-Down  (1, 2, 3, 4, 7, 5, 6, 8, 15, 9, 14, 10, 13, 11, 12)    Variation 2: Print Bottom-Up  (8, 15, 9, 14, 10, 13, 11, 12, 4, 7, 5, 6, 2, 3, 1) |

**C++ Assignment**

| 1. | Write a program by creating an 'Employee' class having the following functions and print the final salary.  1 - 'getInfo()' which takes the salary, number of hours of work per day of employee as parameters  2 - 'AddSal()' which adds $10 to the salary of the employee if it is less than $500.  3 - 'AddWork()' which adds $5 to the salary of the employee if the number of hours of work per day is more than 6 hours. |
| --- | --- |
| 2. | Create a class called 'Matrix' containing a constructor that initialises the number of rows and the number of columns of a new Matrix object. The Matrix class has the following information:  1 - number of rows of matrix  2 - number of columns of matrix  3 - elements of matrix (You can use 2D vector)  The Matrix class has functions for each of the following:  1 - get the number of rows  2 - get the number of columns  3 - set the elements of the matrix at a given position (i,j)  4 - adding two matrices.  5 - multiplying the two matrices  You can assume that the dimensions are correct for the multiplication and addition. |
| 3. | Suppose you have a Piggie Bank with an initial amount of Rs.50 and you have to add some more amount to it. Create a class 'AddAmount' with a data member named 'amount' with an initial value of Rs. 50. Now make two constructors of this class as follows:  1 - without any parameter - no amount will be added to the Piggie Bank  2 - having a parameter which is the amount that will be added to the Piggie Bank  Create an object of the 'AddAmount' class and display the final amount in the Piggie Bank. |

**Linux Assignment**

| 1. |  |
| --- | --- |
| 2. |  |
| 3. |  |