II/IV B. TECH. SECOND SEMESTER
FILE STRUCTURES LAB(Required)

Course Code : CS 4L2 **Credits: 2**
Lab Hours: 3 periods/ week **Internal assessment: 25 Marks**
Tutorial:- **Semester end examination: 50 Marks**

Prerequisites: File Structures

Course Objectives:

1. Provide a solid introduction to the topic of file structure design.
2. Discuss, in detail, the data structures necessary for achieving its efficiency objectives.
3. Introducing the most important high-level file structures tools which include indexing, co sequential processing, B trees, Hashing.

Course Outcomes:
At the end of this course student will:

CO1) Implement various operations on files

CO2) Apply indexing techniques on files

CO3) Employ multiple lists merging concept for files.

CO4) Synthesize and implement the multilevel indexing concept(B Trees) on files

CO5) Apply the hashing technique to resolve collision of records.

Syllabus:

1. Implement the following programs using C++ language.
    a. Write a program to create a class Student. Each student object represents information about a single student. Members should be included for identifier, name, address, date of first enrollment, and number of credit hour completed. Methods should be included for initialization (constructors), assignment (overloaded '=' operator), and modifying field values, including a method to increment the number of credit hours.

    b. Write a program to create a class **CourseRegistration**. Each object represents the enrollment of a student in a course. Members should be included for a course identifier, student identifier, number of credits hours, and course grade. Method should be included as appropriate.

    c. Create a list of student and course registration information. This information will be used in subsequent exercises to test and evaluate the capabilities of the programming project.

2. Write a C++ program to read and write student objects and courseregistration objects with fixed-length records and the fields delimited by "|". Implement pack ( ), unpack ( ), modify ( ) and search ( ) methods.
3. Write a C++ program to read and write student objects and courseregistration objects with Variable - Length records using any suitable record structure. Implement pack ( ), unpack ( ), modify ( ) and search ( ) methods.
4. Write a C++ program to write student objects and courseregistration objects with Variable - Length records using any suitable record structure and to read from this file a student record using RRN.
5. Write a C++ program to implement simple index on primary key for a file of student objects and courseregistration objects. Implement add ( ), search ( ), delete ( ) using the index.
6. Write a C++ program to implement index on secondary key, the name, for a file of student objects and courseregistration objects . Implement add ( ), search ( ), delete ( ) using the secondary index.
7. Write a C++ program to read two lists of names and then match the names in the two lists using Cosequential Match based on a single loop. Output the names common to both the lists.
8. Write a C++ program to read k Lists of names and merge them using k-way merge algorithm with k = 8.
9. Write a C++ program to implement B-Tree for a given set of integers and its operations insert ( ) and search ( ). Display the tree.
10. Use class B+ Tree to create a B-tree index of a student record file with student identifier as key. Write a driver program to create a B-tree file from an existing student record file. Display the tree.
11. Write a C++ program to store and retrieve student data from file using hashing. Use any collision resolution technique.
12. Write a C++ program to reclaim the free space resulting from the deletion of records using linked lists.

Learning Resources:

1. File Structures: An Object-Oriented Approach with C++, Michael J. Folk, Greg Riccardi, Bill Zoellick, Third Edition, Pearson Education.
2. Data Management and File Structures, Mary E.S. Loomis, Second Edition, PHI.
3. File Organization and Processing, Alan L. Tharp, Wiley India Edition.