JHARKHAND Rai University
RANCHI

LAB MANUAL

Master of Computer Application

MCA

(2024-2026) onwards

Vision

To develop the Department of Computer Science & Information Technology as a Center for Excellence to produce leading Professionals who can serve the society with innovative skills, Computer Experts, Researchers to meet the needs of the software industry in national /global scenario responding to the challenges of ever changing world.

Mission

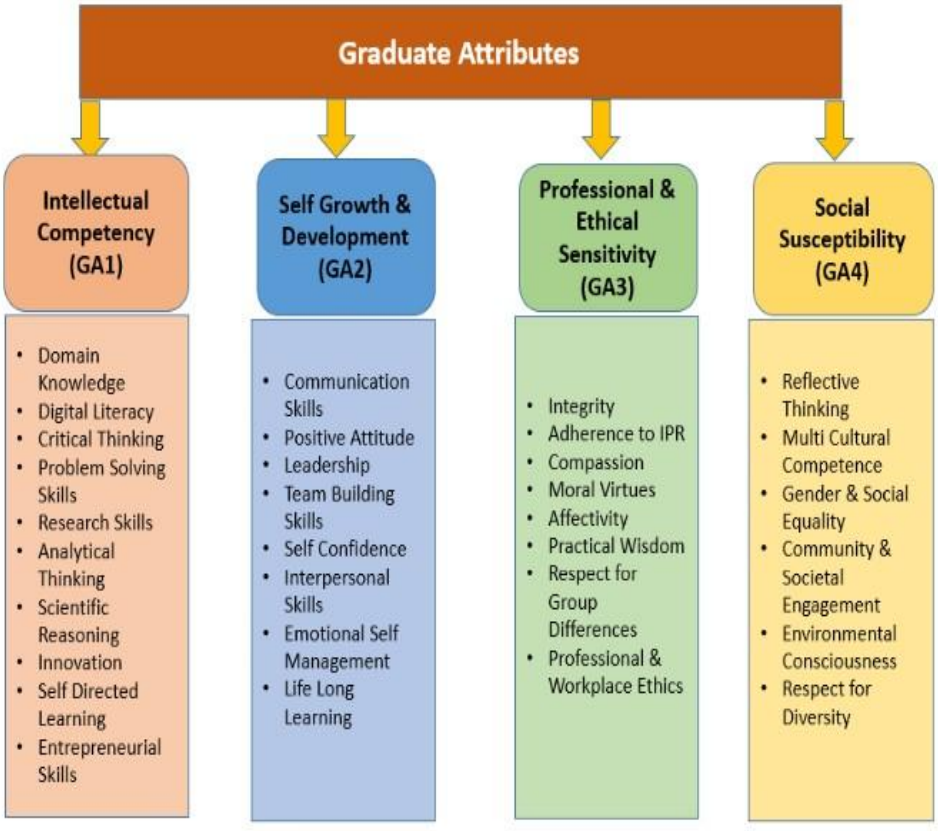
- We endeavor to provide the best possible learning environment to enhance innovations, research capabilities, problem solving skills, leadership qualities, team spirit and ethical responsibilities.
- To nurture the talent of the students to be successful, ethical and effective problem solvers who will contribute positively to the economic growth of the nation and prepare to respond to the challenges.

Graduate Attributes

Jharkhand Rai University is a mecca of transformative education which strongly believes in the holistic development of students. The university provides the cutting-edge of holistic learning to develop promising youngsters into leaders of tomorrow with globally relevant, future-ready and actionable intelligence. The objective of the Department is to make each student proficient in synthesizing/analysing information and be ethical, socially responsible, and just when making decisions. JRU ensures inclusive and equitable quality education and promote lifelong learning opportunities for all.

Every graduate of the Department will be developed to possess the following attributes:

1. Intellectual Competency
2. Self-Growth & Development
3. Professional & Ethical Sensitivity
4. Social Susceptibility



JHARKHAND Rai University RANCHI

L	T	P	C
0	0	4	2

Program: MCA
Semester: First
Course: Advance Database Management System Lab
Course Code: 3CIT101P

Course Objectives:

Students will be able to

1. To design and implement a database schema for a given problem domain
2. To create and manipulate tables using SQL queries
3. To prepare a Database for a given problem
4. To develop applications using PL/SQL

Course Outcomes:

After the successful completion of the course, the students will be able to:

1. Understand, analyze and apply common SQL statements including DDL, DML and DCL statements to perform different operations.
2. Design different views of tables for different users and to apply embedded and nested queries.
3. Design and implement a database for a given problem according to well-known design principles that balance data retrieval performance with data consistency.
4. Apply normalization techniques to avoid redundancy

Course Content:

For the Tables given below:
emp(empno, ename,job, mgr ,hiredate, sal, comm, deptno, gr),
dept(deptno, dname, loc)

Write the following queries:
1. List all information about all department from emp table.
2. List all employee names along with their salaries from emp table.
3. List all department numbers, employee numbers and their managers numbers in descending order of deptno from emp table.
4. List department names and locations from the dept table.
5. List the employees belonging to the department 20.
6. List the name and salary of the employees whose salary is more than 1000.
7. List the names of the clerks working in the department 20.
8. List the names of analysts and salesmen.
9. List the details of the employees who have joined before the end of September 81.
10. List the names of employees who are not managers.
11. List the names of employees whose employee number are 8569, 8521, 8839, 8934, 8788.
12. List the employee details not belonging to the department 10, 30, and 40.
13. List the employee name and salary, whose salary is between 5000 and 10,000.

14. List the employee names, who are not eligible for commission.(salary having >15,000 eligible for commission)
15 List the employees who are eligible for commission
16. List the details of employees, whose salary is greater than 2000 and commission is NULL.
17. List the employees whose names start with an “S” (not”s”).
18. List the name, salary and PF amount of all the employees(PF is calculated as 10% of salary).
19. List the empno, ename, sal in ascending order of salary.
20. List the employee name, salary, job and Department no descending order of Department No and salary.
21. List the employee details in ascending order of salary.
22. List the employee details in descending order of salary

23. Display name, and sal and commission of all employees whose monthly salary is greater than their commission.
24. Select SMITH HAS WORKED IN THE POSITION OF CLERK IN DEPT 20.Display result in this format.
25. Generate a statement which prompts the user at runtime. The intention is to display employees hired between 2 given dates.
26. Define a variable representing an expression used to calculate total annual remuneration. Use the variable in a statement which finds all employees who earn $30000 a year or more.
27. List all the employees name and salaries increased by 15% and expressed as a whole number of dollar
28. Produce the following

EMPLOYEE AND JOB

SMITH CLERK
ALLEN SALESMAN

29. Produce the following output: SMITH(Clerk) ALLEN(Salesman)
30. Do a case sensitive search for a list of employees with a job that the user enters.
31. It has been discovered that the sales people in dept. 30 are not all male. Please produce the following output.

ENAME	DEPTNO	JOB
ALLEN	30	Sales Person

32. Display each employees name and hire date of dept 20.
33. Display each employees name, hire date and salary review date. Assume salary review date is one year from hire date. Output should be in ascending review date.
34. Print list of employees displaying just salary, if more than 1500. If exactly 1500 display “ On Target”. If less than 1500 display “ Below 1500”.
35. Write a query which returns DAY of the week (i.e. MONDAY) for any date entered in the format DD/MM/YY.
36. Write a query to calculate length of service of each employee.
37. Find the minimum salary of all employees.
38. Find the maximum, minimum, and average salaries of all employees.
39. List the maximum and minimum salary of each job type.
40. Find how many managers are in each dept.

41. Write a query which returns DAY of the week (i.e. MONDAY) for any date entered in the format DD/MM/YY.
42. Write a query to calculate length of service of each employee.
43. Find the minimum salary of all employees.
44. Write a query which returns DAY of the week (i.e. MONDAY) for any date entered in the format DD/MM/YY.
45. Write a query to calculate length of service of each employee.
46. Find the minimum salary of all employees.
47. Find the maximum, minimum, and average salaries of all employees.
48. List the maximum and minimum salary of each job type.
49. Find how many managers are in each dept.
50. Find the average salary and average total remuneration of each job type. Remembers sales man earn commission.
51. Find out the difference between highest and lowest salary.
52. Find all department s which have more than three employees.
53. Check whether all employee nos are unique. (No Duplicate)
54. List lowest paid employee working for each Manager. Exclude any groups where the minimum salary is less than 1000. Sort the output by salary.

55. Produce a list showing employees 'salary grade'.(> 10000 A, >10000 &<20000 B,>20000 C)
56. Show only employee on Grade C.

48. .Show all employee in Dallas.
49. List the employees name, job, salary, grade and department for everyone in the company except clerks. Sort on salary, displaying the highest first.
50. List the following details of employees who earn $36000 a year or who are clerks.
Ename Job Annual Sal Dept no Dname Grade
51. Display all employees who earn less than their managers.
52. Display all employees by name and eno along with their managers name and number.

Program: MCA
Semester: First
Course: Advance Data structure Lab
Course Code: 3CIT104P

L	T	P	C
0	0	4	2

Course Objective:
Students will be able to

- To assess how the choice of data structures and algorithm design methods impact the performance of programs.
- To choose the appropriate data structure and algorithm design method for a specified application.
- To solve problems using data structures such as linear lists, stacks, queues, hash tables, binary trees, heaps, binary search trees, and graphs and writing programs for these solutions.
- Analyse and compare the different algorithms

Course Outcomes:

After the successful completion of the course, the students will be able to:

- At the end of this lab session, the student will
- Be able to design and analyze the time and space efficiency of the data structure
- Be capable to identity the appropriate data structure for given problem
- Have practical knowledge on the applications of data structures

Course Content:
Program List:

1. Program to Find the Number of Elements in an Array
2. Develop and Implement a menu driven program in C for the following Array operations
a. Creating Array of N Integer elements.
b. Display of Array elements with suitable headings.
c. Inserting an element (ELEM) at a given valid position (POS).
d. Deleting an element at a given valid position (POS).
e. Exit
3. Programs for Stack, Queues and Circular Queues using Arrays
4. Program to convert an Infix Expression into Postfix and Postfix Evaluation
5. Program to implement stack using arrays
6. Program to implement stack using linked list
7. Program to implement multiple stack in a single array
8. Program to convert infix notation to postfix notation using stacks
9. Program to implement queue using arrays
10. Program to implement queue using pointers
11. Program to reverse elements in a queue
12. Program to implement circular queue using arrays
13. Program to create add remove & display element from single linked list
14. Program to create add remove & display element from double linked list

15. Program to count number of nodes in linear linked list
16. Program to create add remove & display element from circular linked list
17. Programs to implement stack & queues using linked representation
18. Program to concatenate two linear linked lists
19. Program to accept a singly linked list of integers & sort the list in ascending order.
20. Program to reverse linked list
21. Program to represent polynomial using linked list
22. Program to add two polynomials using linked list
23. Program for the creation of binary tree, provide insertion & deletion in c
24. Program for pre-order, post-order & in-order traversals of a binary tree using non reccursive.
25. Program to count no, of leaves of binary tree
26. Program for implementation of B-tree (insertion & deletion)
27. Program for implementation of multi-way tree in c
28. Program for implementation of AVL tree
29. Program to implement bubble sort program using arrays
30. Program to implement merge sort using arrays
31. Program to implement selection sort program using arrays
32. Program to implement insertion sort program using arrays
33. Program to implement topological sort using arrays
34. Program to implement heap sort using arrays

35. Program to implement heap sort using pointers
36. Program to implement bubble sort program using pointers
37. Program to implement linear search using pointers
38. Program to implement binary search using pointers
39. Program to implement linear search using arrays
40. Program to implement binary search using arrays

Text books:

Baluja G S, “Data Structure through C”, Ganpat Rai Publication, New Delhi, 2015.
1. Pai G A V, “Data Structures and Algorithms: Concepts, Techniques and Applications”, 2ndEdn, Tata McGraw-Hill, 2008.
2. Horowitz E., Sahni S., Susan A., “Fundamentals of Data Structures in C”, 2nd Edition,

Program: MCA
Semester: Second
Course: Java Programming Lab
Course Code: 3CIT105P

L	T	P	C
0	0	4	2

Course Objectives:

Students will be able to

1. To write programs using abstract classes.
2. To write programs for solving real world problems using java collection frame work.
3. To write multithreaded programs.
4. To write GUI programs using swing controls in Java.
5. To introduce java compiler and eclipse platform.
6. To impart hands on experience with java programming.

Course Outcomes:

After the successful completion of the course, the students will be able to:

1. Write Java application programs using OOP principles and proper program structuring.
2. Develop Java program using packages, inheritance and interface.
3. Create multithreaded programs.
4. Write Java programs to implement error handling techniques using exception handling and develop programs using class and inputs from keyboard.
5. Develop graphical User Interface using AWT.
6. Demonstrate event handling mechanism.

Write a java:

1. Program to find square root of given number.
2. Program to enter principal, rate & time and find simple interest.
3. Program to create an object.
4. Program to enter a character from keyboard and find out the ASCII value of the character.
5. Program to enter a number from keyboard and find out Fibonacci series.
6. Program to enter a number from keyboard and find out factorial of the number.
7. Program to enter a number from keyboard and print the prime numbers present within it.
8. Program to swap two numbers without using third variable.
9. Program to show inheritance.
10. Program to demonstrate dynamic method dispatch.
11. Program to demonstrate abstraction.
12. Program to enter a number from keyboard and convert it into binary form and vice-versa.
13. Program to sort an array in an ascending order.

14. Program to find out the sum and average of the elements present in an array.
15. Program to find out the biggest and smallest number from a matrix.
16. Program to enter a string from keyboard and check how many vowels and consonants are present.
17. Program to reverse a specified string.
18. Program demonstrating the use of package.
19. Program to show the current date, date after one day and date before ten days.
20. Program to draw the shape.
21. Program to find out the pixel color.
22. Program to move the bubbles using thread.
23. Program to create an applet.
24. Program to create a form using applet.
25. Program to show the database connectivity.

Program: MCA
Semester: Second
Course: Computer Graphics lab
Course Code: 3CITE202P

L	T	P	C
0	0	2	1

Course Objectives:

Students will be able to

1. Understand the need of developing graphics application.
2. Learn algorithmic development of graphics primitives like: line, circle, polygon etc.
3. Learn the representation and transformation of graphical images and pictures.

Course Outcomes:

After the successful completion of the course, the students will be able to:

1. Understand the basic concepts of computer graphics.
2. Design scan conversion problems using C++ programming.
3. Apply clipping and filling techniques for modifying an object.
4. Understand the concepts of different type of geometric transformation of objects in 2D and 3D.
5. Understand the practical implementation of modeling, rendering, viewing of objects in 2D.

Course Content:

1. Write a program to draw a point.
2. Write a program to draw line
3. Write a program to draw a circle.
4. Write a program to draw a rectangle
5. Write a program to draw a sector
6. Write a program to draw an ellipse
7. Write a program to draw an arc.
8. Write a program to draw smiley face
9. Write a program to draw a polygon
10. Write a program to draw a filled polygon.
11. Write a program to draw a concentric circle.
12. Write a program to draw a kite
13. Write a program to draw pie slice
14. Write a program to draw a bar
15. Write a program to draw a 3D bar
16. Write a program to draw pie chart
17. Write a program to draw bar chart
18. Write a program to draw 3D Bar chart
19. Write a program to draw a digital clock

20. Write a program to draw a car.
21. Write a program to draw a moving car
22. Write a program to draw a half filled glass with blue water
23. Write a program to explain DDA Line Drawing algorithm.
24. Write a program to explain Bresenham's Line Drawing algorithm
25. Write a program to explain Bresenham's Circle Drawing algorithm.
26. Write a program to explain :
    - Text Animation
    - 2D Rotation, Translation, Scaling
    - 3D Rotation, Translation, Scaling

Line Clipping

Program: MCA
Semester: Third
Course: Artificial Intelligence and Applications lab
Course Code: 3CIT201P

Course Objectives:
Students will be able to

- The learning objectives of this course are:
- To learn how to design and program Python applications.
- To learn how to use lists, tuples, and dictionaries in Python programs.
- To learn how to identify Python object types.
- To define the structure and components of a Python program.
- To learn how to write loops and decision statements in Python.

Course Outcomes:
After the successful completion of the course, the students will be able to:

- At the end of the course, the student will be able to
- Explain basic principles of Python programming language
- Implement object oriented concepts,
- Implement database and GUI applications.
- Learn the structure and components of a Python program

Course Content:

List of experiments:

1. Python program to print your name.
2. Python program to find area of a circle.
3. Python program to check whether a number is even or not.
4. Python program to find factorial of a given number.
5. Python program for n-th Fibonacci series.
6. Python program to print all positive numbers in a range.
7. Python program to print ASCII value of a character.
8. Python program to print sum of n numbers.
9. Python program to find largest number in a list.
10. Python program to print even numbers in a list.
11. Python program to remove i'th character from string in Python.
12. Python program to implement graph.
13. Python program to implement BFS on graph.
14. Python Program to Find Shortest Path from a Vertex using BFS in an Unweight Graph.
15. Python Program to find if Directed Graph contains Cycle using DFS

Program: MCA
Semester: Third
Course: Python Programming lab
Course Code: 3CIT203P

L	T	P	C
0	0	2	1

Course Objectives:

Students will be able to

1. To acquire programming skills in core Python.
2. To acquire Object Oriented Skills in Python.
3. To develop the skill of designing Graphical user Interfaces in Python.
4. To develop the ability to write database applications in Python.

Course Outcomes:

After the successful completion of the course, the students will be able to:

1. Understand and comprehend the basics of python programming.
2. Demonstrate the principles of structured programming and be able to describe, design, implement, and test structured programs using currently accepted methodology.
3. Explain the use of the built-in data structures list, sets, tuples and dictionary.
4. Make use of functions and its applications.
5. Identify real-world applications using oops, files and exception handling provided by python.

List of Programs:

1. Program to print “Hello” five times.

2. Write a program to convert fahrenheit to celcius

3. Write a Program for checking whether the given number is a even number or not.

4. Write a program add that takes 2 numbers as command line arguments and prints its sum.

5. Using a for loop, write a program that prints out the decimal equivalents of 1/2, 1/3, 1/4, . . . ,1/10.

6. Write a program to compute distance between two points taking input from the user (Pythagorean Theorem)

7. Write a function nearly equal to test whether two strings are nearly equal. Two strings a and b are nearly equal when a can be generated by a single mutation on b.

8. Write a program using for loop that loops over a sequence.

9. Write a program using a while loop that asks the user for a number, and prints a countdown from that number.

10. Write a program to count the numbers of characters in the string and store them in a dictionary data structure.

11. Write a program to perform addition of two square matrices
12. Write a program to perform multiplication of two square matrices.
13. Find mean, median, mode for the given set of numbers in a list.
14. Write a function unique to find all the unique elements of a list.

15. Write a function dups to find all duplicates in the list.
16. Write function to compute gcd, lcm of two numbers
17. Find mean, median, mode for the given set of numbers in a list.
18. Write a program to perform multiplication of two square matrices.
19. Write a program to print each line of a file in reverse order.
20. Write a program to compute the number of characters, words and lines in a file.
21. Write a function ball_collide that takes two balls as parameters and computes if they are colliding. Your function should return a Boolean representing whether or not the balls If (distance between two balls centers) <= (sum of their radii) then (they are colliding)

Reference Books:

1. Allen B. Downey , " Think Python: How to Think Like a Computer Scientist", Second Edition, Updated for Python 3, Shroff/O'Reilly Publishers, 2016.

2. Shroff "Learning Python: Powerful Object-Oriented Programming; Fifth edition, 2013.

3. David M.Baezly "Python Essential Reference". Addison-Wesley Professional; Fourth edition, 2009. 4. David M. Baezly "Python Cookbook" O'Reilly Media; Third edition, 2013.

Program: MCA
Semester: Third
Course: Compiler Design lab
Course Code: 3CIT202P

L	T	P	C
0	0	2	1

Course Objectives:

Students will be able to

1. To implement Lexical Analyzer using Lex tool & Syntax Analyzer or parser using YACC Tool
2. To implement NFA and DFA from a given regular expression
3. To implement front end of the compiler by means of generating Intermediate codes.
4. To implement code optimization techniques.

Course Outcomes:

After the successful completion of the course, the students will be able to:

1. Design Lexical analyzer for given language using C and LEX tools. CO
2. Design and convert BNF rules into YACC form to generate various parsers. CO
3. Generate machine code from the intermediate code forms CO
4. Implement Symbol table

List of experiments:

- C program to count white spaces, numbers, words in a file.
- C program to design Finite automata to identify different tokens (identifiers, constants, operators, etc.).
- C program to count number of a's in given string.
- C program to Identify different patterns like aa, ab, not containing a, etc. in given string.
- Lex program to identify all tokens of C programs.
- Design and Code individual programming code with all possible tokens in programming
- Starting and ending with 'a'.
- # a's divisible by 2 or b's divisible by 3.
- 4th Symbol 'a' from RHS.
- Output code after removing white spaces and comment.

Build parsers using yacc for L(G)={$anbn \mid n>=1$} over {a,b}

- Build Parser using yacc for L(G) where rule set of G is { S -> aSb, S-> bSa, S->c} over {a,b,c}.
- Build parser using yacc to convert the infix expression to postfix expression.
- Build a calculator in yacc which takes expression in postfix notation.
- Build parsers using yacc to convert the prefix expression into the postfix expression.