BACHELOR OF COMPUTER APPLICATIONS (BCA_NEW)

BCA_NEW /ASSIGN/SEMESTER-III

ASSIGNMENTS

(July, 2025 & January, 2026 Sessions)

MCS-208, MCSL-209, MCS-207, BCS-131, BCSL-135, BCS-040

SCHOOLOFCOMPUTERANDINFORMATIONSCIENCES
INDIRA GANDHI NATIONAL OPEN UNIVERSITY
MAIDAN GARHI, NEW DELHI – 110 068

CONTENTS

Important Notes

1. Submit your assignments to the Coordinator of your Study Centre on or before the due date.
2. Assignment submission before due dates is compulsory to become eligible for appearing in corresponding Term End Examinations. For further details, please refer to BCA Programme Guide.
3. To become eligible for appearing the Term End Practical Examination for the lab courses, it is essential to fulfill the minimum attendance requirements as well as submission of assignments (on or before the due date). For further details, please refer to the BCA Programme Guide.

Course Code	**:**	**BCSL-135**
Course Title	**:**	**DBMS and C++ Lab**
Assignment Number	**:**	**BCA_NEW(III)-135/Assignment/2025-26**
Maximum Marks	**:**	**100**
Weightage	**:**	**30%**
Last Date of Submission	**:**	**31st October,2025(For July 2025 Session)** **30th April, 2026 (For January 2026 Session)**

Note: This assignment has one question with two sections for a total of 40 marks. The rest 10 marks are for viva voce. You must complete both sections. For Section A, provide the database design and SQL queries. For Section B, write the C++ program and attach its output printout.

Q1. Project: Contact Management System

You are required to design and develop a simple Contact Management System. This project will be completed in two parts: first, you will design the database schema and write SQL queries (Section A), and second, you will develop a C++ console application to manage the contacts (Section B).

Section A:

Design a database schema in **3rd Normal Form (3NF)** to store contact information.

(a) Database Schema:
Create the necessary tables to store the following information:

- A unique Contact_ID for each contact.
- The First_Name and Last_Name of the contact.
- Multiple Phone_Numbers for each contact (e.g., 'Mobile', 'Home', 'Work'). Each phone number should be stored with its type.
- Multiple Email_Addresses for each contact (e.g., 'Personal', 'Work'). Each email should be stored with its type.
- A Category for each contact (e.g., 'Family', 'Friend', 'Work').

Your design should include: **(10 Marks)**

- An ER Diagram for the system.
- The final set of tables with all columns clearly defined. Underline the primary key and specify all foreign keys.

(b) SQL Queries:
Write SQL queries for the following operations on the database you designed: **(10 Marks)**

1. Retrieve the full name, all phone numbers, and all email addresses of a contact with a specific Contact_ID.
2. List all contacts (First Name and Last Name) belonging to the 'Work' category.
3. Find the Contact_ID and First_Name of all contacts who have a 'Mobile' phone number.
4. Count the total number of contacts in each category.
5. List the names of contacts who have more than one phone number registered.

Section B: C++ Application Development

Write a C++ program that acts as a console-based interface for the Contact Management System. Since this is a standalone C++ application, you will manage the data using classes and file handling. **(20 Marks)**

Your program must implement the following:

1. **A Contact Class:**
    - Create a class named Contact with private member variables to store contactID, firstName, lastName, and category.
    - Use a std::vector or a similar container within the class to store multiple phone numbers and email addresses.
    - Include a constructor and public member functions to add/edit details and display contact information.
2. **File Handling:**
    - On starting the application, it should load all existing contact records from a text file named contacts.dat.
    - When the user chooses to exit, the application must save all contact records (including any new or modified ones) back to the contacts.dat file.
3. **Menu-Driven Interface:**
   The program should display a menu with the following options:
    - **1. Add New Contact:** Prompt the user for all details and add the new contact. The contactID should be generated automatically (e.g., sequentially).
    - **2. Search for a Contact:** Allow searching by contactID or lastName. Display the full details if found.
    - **3. Delete a Contact:** Ask for a contactID and remove the corresponding contact record.
    - **4. Update a Contact:** Ask for a contactID and allow the user to modify the contact's details.
    - **5. Display All Contacts:** Display a summary (ID, Full Name, Category) of all contacts.
    - **6. Exit:** Save all data to the file and terminate the program.

The program should be robust and handle basic user input errors gracefully.