COLLEGE DEPARTMENT MANAGEMENT

Project report submitted in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

R. Suhitha 24KB1A05FY

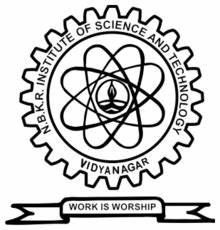
R. Asritha 24KB1A05FR

V. Nischitha 24KB1A05MX

S. Bavani 24KB1A05GJ

Under the Guidance of

P. Suneetha



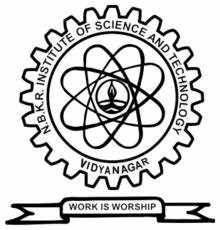
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

N.B.K.R. INSTITUTE OF SCIENCE AND TECHNOLOGY

N.B.K.R. INSTITUTE OF SCIENCE AND TECHNOLOGY

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



**CERTIFICATE**

This is to certify that that the project report entitled College Department Management being submitted by

R. Suhitha 24KB1A05FY

R. Asritha 24KB1A05FR

V. Nischitha 24KB1A05MX

S. Bavani 24KB1A05GJ

In partial fulfilment for the award of the degree of bachelor of technology in computer science and engineering to the N.B.K.R.IST is a record of Bonafede work carried out under my guidance and supervision

P. Suneetha

Designation

DECLARATON

I hereby declare that the dissertation entitled college department management submitted for the B. Tech Degree is my original work and the dissertation has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles.

Place: Vidyanagar

Date:7may2025

R. Suhitha 24KB1A05FY

R. Asritha 24KB1A05FR

V. Nischitha 24KB1A05MX

S. Bavani 24KB1A05GJ

Acknowledgment:

I would like to express my sincere gratitude to all those who supported and guided me throughout the development of this project titled “College Department Management System”.

First and foremost, I would like to thank my project guide, [Your Teacher’s/Professor’s Name], for their invaluable support, encouragement, and constructive feedback during the course of this work. Their guidance helped me in understanding the problem and implementing an efficient solution using C programming.

I would also like to extend my thanks to the [Department Name, e.g., Department of Computer Science], and all the faculty members who provided a helpful environment and resources that contributed to the successful completion of this project.

Abstract:

The College Department Management System is a C-based application designed to efficiently manage the academic structure of a college by organizing departments and their respective faculty members. The system uses an array of departments such as CSE, ECE, and MECH, with each department maintaining a linked list of faculty members. This structure enables dynamic storage and management of faculty data within each department.

The primary goal of this project is to demonstrate how fundamental data structures like arrays and linked lists can be utilized to model real-world systems in an organized and scalable way. Users can add and view faculty members under each department, making it a useful tool for academic administration and record

Keeping.

Introduction:

This C program is designed to manage faculty members across various college departments such as CSE, ECE, Civil, Mech, and EEE. Each department is represented by a linked list that stores the details of its faculty members, including their ID, name, and specialization. The program allows users to:

1. Add faculty to a selected department

2. View all departments with their faculty lists

3. Search for a faculty member by name

4. Free the memory before exiting

It provides a menu-driven interface to perform these operations easily

Literature Survey / Existing System:

Traditionally, faculty records in educational institutions are maintained manually through paper-based systems or spreadsheets, which can be inefficient and error-prone. While modern solutions like database systems and ERP software offer more features, they require significant resources, internet access, and technical expertise. This C-based program simplifies faculty management by using arrays and linked lists, offering a lightweight, easy-to-use solution for managing faculty data across departments without the need for complex setups or external tools.

Software Requirements Analysis:

Functional Requirements:

Add, display, and search for faculty members in different departments.

Free memory after use to avoid leaks.

Non-Functional Requirements:

Efficient, easy-to-use interface.

Portable and optimized for memory usage.

Hardware Requirements:

Basic computing system with keyboard and display.

Software Requirements:

C programming language and a C compiler (e.g., GCC).

Compatible with Windows, Linux, and macOS.

Limitations:

Fixed number of departments (5).

Memory constrained by available system RAM.

Software Design:

The program uses an array of departments and linked lists to manage faculty data. Each department points to a linked list storing faculty members dynamically.

Functions:

add\_faculty(), display\_all(), search\_faculty(), free\_memory()

Structure:

Array for departments

Linked list for faculty per department

Interface:Simple menu-driven command-line interface.

Proposed System:

The proposed system is a simple, menu-driven C program for managing faculty members across various college departments. It uses arrays to represent departments and linked lists to dynamically store faculty details like ID, name, and specialization. The system allows adding, viewing, and searching faculty efficiently, making it a lightweight and effective solution for academic use without the need for complex databases or external tools.

Coding:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_DEPTS 5

#define MAX\_NAME\_LEN 50

// Faculty structure for linked list

struct Faculty {

char name[MAX\_NAME\_LEN];

int id;

char specialization[50];

struct Faculty\* next;

};

// Department array with linked lists

struct Faculty\* departments[MAX\_DEPTS] = {NULL};

const char\* dept\_names[MAX\_DEPTS] = {"CSE", "ECE", "Civil", "Mech", "EEE"};

// Function prototypes

void add\_faculty();

void display\_all();

void search\_faculty();

void free\_memory();

int main() {

int choice;

do {

printf("\nCollege Department Management\n");

printf("1. Add Faculty Member\n");

printf("2. Display All Departments\n");

printf("3. Search Faculty\n");

printf("4. Exit\n");

printf("Enter choice: ");

scanf("%d", &choice);

switch(choice) {

case 1: add\_faculty(); break;

case 2: display\_all(); break;

case 3: search\_faculty(); break;

case 4: free\_memory(); break;

default: printf("Invalid choice!\n");

}

} while(choice != 4);

return 0;

}

void add\_faculty() {

int dept\_choice;

printf("\nAvailable Departments:\n");

for(int i = 0; i < MAX\_DEPTS; i++) {

printf("%d. %s\n", i+1, dept\_names[i]);

}

printf("Select department (1-5): ");

scanf("%d", &dept\_choice);

if(dept\_choice < 1 || dept\_choice > MAX\_DEPTS) {

printf("Invalid department selection!\n");

return;

}

struct Faculty\* new\_faculty = (struct Faculty\*)malloc(sizeof(struct Faculty));

if(!new\_faculty) {

printf("Memory allocation failed!\n");

return;

}

printf("Enter Faculty ID: ");

scanf("%d", &new\_faculty->id);

printf("Enter Faculty Name: ");

scanf("%s", new\_faculty->name);

printf("Enter Specialization: ");

scanf("%s", new\_faculty->specialization);

// Insert at beginning

new\_faculty->next = departments[dept\_choice-1];

departments[dept\_choice-1] = new\_faculty;

}

void display\_all() {

printf("\nDepartment Faculty List:\n");

for(int i = 0; i < MAX\_DEPTS; i++) {

printf("\n%s Department:\n", dept\_names[i]);

struct Faculty\* current = departments[i];

while(current) {

printf("ID: %d, Name: %s, Specialization: %s\n",

current->id, current->name, current->specialization);

current = current->next;

}

}

}

void search\_faculty() {

char search\_name[MAX\_NAME\_LEN];

printf("Enter faculty name to search: ");

scanf("%s", search\_name);

for(int i = 0; i < MAX\_DEPTS; i++) {

struct Faculty\* current = departments[i];

while(current) {

if(strcmp(current->name, search\_name) == 0) {

printf("Found in %s Department:\n", dept\_names[i]);

printf("ID: %d, Specialization: %s\n",

current->id, current->specialization);

return;

}

current = current->next;

}

}

printf("Faculty not found!\n");

}

void free\_memory() {

for(int i = 0; i < MAX\_DEPTS; i++) {

struct Faculty\* current = departments[i];

while(current) {

struct Faculty\* temp = current;

current = current->next;

free(temp);

}

departments[i] = NULL;

}

printf("Memory freed. Exiting...\n");

}

Output:

A computer screen shot of a program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

Conclusion:

The College Department Management System effectively demonstrates how arrays and linked lists can be used to manage faculty information in a structured and efficient way. The program provides essential features such as adding, displaying, and searching faculty members while ensuring memory is managed properly. It is a lightweight, easy-to-use solution suitable for small educational institutions or as a base for more advanced systems.

References:

1. E. Balagurusamy, Programming in ANSI C, McGraw Hill Education.

2. Yashavant Kanetkar, Let Us C, BPB Publications.

3. Online C documentation – https://www.cprogramming.com

4. TutorialsPoint – C Programming Language – https://www.tutorialspoint.com/cprogramming

5. GeeksforGeeks – Data Structures in C – <https://www.geeksforgeeks.org>

THANK YOU