Institute of Computer Technology

B. Tech. Computer Science and Engineering

Sub: ESFP – II

Course Code: 2CSE203

Practical – 4

Name: Jaymin Gondaliya

Enrollment No: 23162171007

Sem - 2

Branch: CS

Class: B

Batch: 25

Objective:

To learn DMA (Dynamic memory allocation) and Double Linked-list

Problem Definition-1:

At Mumbai, Wellington college is situated at Bandra, where this college is offering so many varieties of courses like diploma, undergraduate, post graduate and technical education courses. At the time of admission college suffers a lot of problems due to manual work. So, for the betterment process of admission, college authority decides to adopt technical help from software technologies. For that, the college authority wants to make a software for performing all the tasks related to the admission process in a single desk. So that, the College authorities want, parents or students should not suffer for getting any types of information related to college from home before taking admission in different-different stream courses. So, whenever he/she wants to get information, they can check all information from the site and at the end just for the admission process he/she should come to college. For that, College gave one sample admission form to the developer to perform the specific task, the remaining part will see the latter as per the functionality workout. In the admission form, all the details are given for the admission purpose like college_ID, College_Name, course_stream, year, semester, subject group and their fee structure. So, as per the requirement, make a proper dynamic memory allocation program using “double link list concept”, where you have to perform all the above said requirements. Follow is the following instruction.

Code:

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct College
{
    int id;
    char name[50];
    char course[50];
    int year;
    int semester;
    char subject[50];
    float fee;
    struct College *next;
    struct College *prev;
};

struct College *head = NULL;
struct College *tail = NULL;

void addCollege()
{
    struct College *new_college = (struct College *)malloc(sizeof(struct
College));
    if (new_college == NULL)
    {
        printf("Memory allocation failed\n");
        exit(1);
    }
    printf("Enter College ID: ");
    scanf("%d", &new_college->id);
    printf("Enter College Name: ");
    scanf("%s", new_college->name);
    printf("Enter Course Stream: ");
    scanf("%s", new_college->course);
    printf("Enter Year: ");
    scanf("%d", &new_college->year);
    printf("Enter Semester: ");
    scanf("%d", &new_college->semester);
    printf("Enter Subject Group: ");
    scanf("%s", new_college->subject);
    printf("Enter Fee Structure: ");
    scanf("%f", &new_college->fee);

    new_college->next = NULL;
    new_college->prev = tail;

    if (head == NULL) {
```

```
        head = new_college;
    } else {
        tail->next = new_college;
    }
    tail = new_college;
}

void displayColleges()
{
    struct College *current = head;
    printf("College ID\tCollege Name\tCourse Stream\tYear\tSemester\tSubject
Group\tFee Structure\n");
    while (current != NULL)
    {
        printf("%d\t\t%s\t\t%s\t\t%d\t%d\t\t%s\t\t%.2f\n", current->id,
current->name,
               current->course, current->year, current->semester, current-
>subject, current->fee);
        current = current->next;
    }
}

void searchCollege()
{
    char search_Name[50];
    printf("Enter College ID or Name to search: ");
    scanf("%s", search_Name);

    struct College *current = head;
    while (current != NULL)
    {
        if ((atoi(search_Name) == current->id) || (strcmp(current->name,
search_Name) == 0))
        {
            printf("College ID\tCollege Name\tCourse
Stream\tYear\tSemester\tSubject Group\tFee Structure\n");
            printf("%d\t\t%s\t\t%s\t\t%d\t%d\t\t%s\t\t%.2f\n", current->id,
current->name,
                   current->course, current->year, current->semester, current-
>subject, current->fee);
            return;
        }
        current = current->next;
    }
    printf("College with ID/Name %s not found.\n", search_Name);
}


int main()
```

```c
{
    int num, choice;
    printf("Enter the number of colleges: ");
    scanf("%d", &num);

    for (int i = 0; i < num; i++)
    {
        printf("\nCollege %d:\n", i + 1);
        addCollege();
    }

    while (1)
    {
        printf("\n1. Display Colleges\n");
        printf("2. Search College\n");
        printf("3. Exit\n");
        printf("Enter Your Choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            displayColleges();
            break;
        case 2:
            searchCollege();
            break;
        case 3:
            exit(0);
        default:
            printf("Invalid Choice\n");
        }
    }
    return 0;
}
```

Output –

```
Enter Fee Structure: 49000

College 2:
Enter College ID: 7009
Enter College Name: Ganpat
Enter Course Stream: BE
Enter Year: 2023
Enter Semester: 3
Enter Subject Group: BDA
Enter Fee Structure: 40000

1. Display Colleges
2. Search College
3. Exit
Enter Your Choice: 1
College ID      College Name    Course Stream   Year    Semester        Subject Group   Fee Structure
4009            GUNI            Btech           2023    5               CS              49000.00
7009            Ganpat          BE              2023    3               BDA             40000.00

1. Display Colleges
2. Search College
3. Exit
Enter Your Choice: 2
Enter College ID or Name to search: 4009
College ID      College Name    Course Stream   Year    Semester        Subject Group   Fee Structure
4009            GUNI            Btech           2023    5               CS              49000.00

1. Display Colleges
2. Search College
3. Exit
Enter Your Choice: 2
Enter College ID or Name to search: Ganpat
College ID      College Name    Course Stream   Year    Semester        Subject Group   Fee Structure
7009            Ganpat          BE              2023    3               BDA             40000.00
```