

## Experiment 22 : Hashing – Separate Chaining

Code:

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

#define SIZE 10 // Hash table size

struct Node {
    int data;
    struct Node* next;
};

struct Node* hashTable[SIZE];

// Function to create a new node
struct Node* createNode(int data) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->next = NULL;
    return newNode;
}

// Hash function
int hashFunction(int key) {
    return key % SIZE;
}
```

```
// Insert an element

void insert(int key) {

    int index = hashFunction(key);

    struct Node* newNode = createNode(key);

    if (hashTable[index] == NULL) {

        hashTable[index] = newNode;

    } else {

        struct Node* temp = hashTable[index];

        while (temp->next != NULL)

            temp = temp->next;

        temp->next = newNode;

    }

    printf("%d inserted at index %d\n", key, index);

}
```

```
// Search for an element

void search(int key) {

    int index = hashFunction(key);

    struct Node* temp = hashTable[index];

    while (temp != NULL) {

        if (temp->data == key) {

            printf("%d found at index %d\n", key, index);

            return;

        }

        temp = temp->next;

    }

    printf("%d not found in hash table\n", key);

}
```

```
// Display hash table
```

```
void display() {  
    printf("\n--- Hash Table ---\n");  
    for (int i = 0; i < SIZE; i++) {  
        printf("[%d] -> ", i);  
        struct Node* temp = hashTable[i];  
        while (temp != NULL) {  
            printf("%d -> ", temp->data);  
            temp = temp->next;  
        }  
        printf("NULL\n");  
    }  
  
    // Main menu  
    int main() {  
        int choice, key;  
  
        while (1) {  
            printf("\n--- Hashing (Separate Chaining) Menu ---\n");  
            printf("1. Insert\n2. Search\n3. Display\n4. Exit\n");  
            printf("Enter your choice: ");  
            scanf("%d", &choice);  
  
            switch (choice) {  
                case 1:  
                    printf("Enter value to insert: ");  
                    scanf("%d", &key);  
                    insert(key);  
                    break;  
                case 2:  
                    printf("Enter value to search: ");  
            }  
        }  
    }  
}
```

```
    scanf("%d", &key);
    search(key);
    break;
case 3:
    display();
    break;
case 4:
    exit(0);
default:
    printf("Invalid choice!\n");
}
}

return 0;
}
```

Output:

```
--- Hashing (Separate Chaining) Menu ---
1. Insert
2. Search
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 49
49 inserted at index 9

--- Hashing (Separate Chaining) Menu ---
1. Insert
2. Search
3. Display
4. Exit
Enter your choice: 3

--- Hash Table ---
[0] -> NULL
[1] -> NULL
[2] -> 22 -> NULL
[3] -> NULL
[4] -> NULL
[5] -> 25 -> 15 -> NULL
[6] -> 66 -> NULL
[7] -> NULL
[8] -> NULL
[9] -> 49 -> NULL
```