

DATA STRUCTURES

DAY-9

1.Hashing program

Program:

```
#include <stdio.h>

#include <stdlib.h>

#define SIZE 10

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

struct node* hashTable[SIZE];

void insert(int key) {
    int index = key % SIZE;

    struct node* newNode = (struct node*)malloc(sizeof(struct node));

    newNode->data = key;
    newNode->next = NULL;

    if (hashTable[index] == NULL) {
        hashTable[index] = newNode;
    } else {
        struct node* temp = hashTable[index];

        while (temp->next != NULL) {
            temp = temp->next;
        }

        temp->next = newNode;
    }
}
```

```

    }
}

void display() {
    for (int i = 0; i < SIZE; i++) {
        struct node* temp = hashTable[i];

        printf("Index %d:", i);

        while (temp != NULL) {
            printf(" %d", temp->data);

            temp = temp->next;
        }

        printf("\n");
    }
}

int main() {
    insert(5);
    insert(15);
    insert(25);
    insert(35);
    insert(6);
    insert(16);
    insert(26);
    insert(36);
    display();
    return 0;
}

```

Output:

Index 0:

Index 1:

Index 2:

Index 3:

Index 4:

Index 5: 5 15 25 35

Index 6: 6 16 26 36

Index 7:

Index 8:

Index 9:

2.Hashing separate program in c

Program:

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

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

```
#define SIZE 10
```

```
struct Node {
```

```
    int data;
```

```
    struct Node* next;
```

```
};
```

```
struct Node* hashTable[SIZE];
```

```
void insert(int key) {
```

```
    int index = key % SIZE;
```

```
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
```

```
    newNode->data = key;
```

```

newNode->next = NULL;

if (hashTable[index] == NULL) {
    hashTable[index] = newNode;
} else {
    struct Node* temp = hashTable[index];
    while (temp->next != NULL) {
        temp = temp->next;
    }
    temp->next = newNode;
}
}

void display() {
    for (int i = 0; i < SIZE; i++) {
        struct Node* temp = hashTable[i];
        printf("Index %d:", i);
        while (temp != NULL) {
            printf(" %d", temp->data);
            temp = temp->next;
        }
        printf("\n");
    }
}

int main() {
    insert(5);
    insert(15);
    insert(25);

```

```
    insert(6);  
    insert(16);  
    display();  
    return 0;  
}
```

Output:

Index 0:

Index 1:

Index 2:

Index 3:

Index 4:

Index 5: 5 15 25

Index 6: 6 16

Index 7:

Index 8:

Index 9: