জব নং-০৬

জবের নাম: লিংকড লিস্টে ডাটা সংযোজন এবং বিয়োজন করার জন্য প্রোগ্রাম লেখা ও এক্সিকিউট করা।

উদ্দেশ্য:

- লিংকড লিস্ট সম্পর্কে ধারণা লাভ করা।
- লিংকড লিস্টে ডাটা সংযোজন এবং বিয়োজন সম্পর্কে পূর্ণাঙ্গ জ্ঞান লাভ করা।
- বিভিন্ন সমস্যা সমাধানের জন্য কম্পাইলার ব্যবহার করে সি ভাষায় কোড লেখা।
- কোড কম্পাইল ও রান করা।
- প্রোগ্রামের আউটপুট পর্যবেক্ষণ করা।

কাজের ধারা:

- ধাপ-১: প্রদত্ত সমস্যা সমাধান করার জন্য অ্যালগরিদম তৈরি করতে হবে।
- ধাপ-২৪ অ্যালগরিদম অনুযায়ী ফ্লোচার্ট তৈরি করতে হবে।
- ধাপ-৩৪ প্রোগ্রামিং কোড লিখতে হবে।
- ধাপ-৪: প্রোগ্রামিং কোডকে কম্পাইল ও ডিবাগ করতে হবে।
- ধাপ-৫: লিখিত প্রোগ্রামটিকে এক্সিকিউট করতে হবে।
- ধাপ-৬: সর্বশেষ আউটপুট পর্যবেক্ষণ করতে হবে এবং কাজের রেকর্ড রাখতে হবে।

কাজের বিবরণী :

নির্দিষ্ট নোডের পরে নোড সংযোজন (Inserting after a given node): একটি নির্দিষ্ট লোকেশন LOC-এ Data সংযোজনের জন্য প্রথমে আমাদের খুঁজে দেখতে হবে List-টি পূর্ণ কি না। যদি LOC করতে হবে। NULL হয়, তবে ITEM-টি প্রথম Node হিসেবে যুক্ত।
মনে করি, LOC-এর জন্য আমরা একটি মান নির্ধারণ করেছি। Node NLOC-এ প্রদত্ত এই মান লিস্টে A নোজের অবস্থান নির্দেশ করে অথবা LOC NULL নির্দেশ করে। নিচের অ্যালগরিদমী LIST-এ ITEM সংযোজন করে, যাতে ITEM Node A-কে নির্দেশ করে অথবা LOC= NULL হয় তখন ITEM-ই হবে প্রথম নোড। মনে করি, NEW লোকেশনের নতুন নোডকে Node N দ্বারা নির্দেশ করা হচ্ছে। এখানে Node A-এর যে Link Pointer- Node B-কে নির্দেশ করত সেটি এখন Node N-কে নির্দেশ করছে এবং Node N-এর Link Pointer-টি এখন Node B-কে নির্দেশ করে। নিম্নোক্ত Statement-এর মাধ্যমে এটি সম্পাদিত হয়- LINK [NEW]: LINK [LOC] LINK (LOC): NEW

Algorithm-টি নিম্নে দেখানো হলো-
Algorithm: INSLOC (INFO, LINK, START, AVAIL, LOC, ITEM) এই অ্যালগরিদম ITEM insert করে যাতে করে 'ITEM' LOC লোকেশনের নোডকে নির্দেশ করে। অথবা LOC = NULL হলে, ITEM-কে প্রথম নোড হিসেবে List-এ সংযোজন করে।
1. [OVERFLOW?] If AVAIL = NULL, then: Write: OVERFLOW, and Exit.
2. [Remove first node from AVAIL list.]
Set NEW: AVAIL and AVAIL: LINK[AVAIL].
3. Set INFO[NEW]: ITEM. [Copies new data into new node.]
4. If LOCNULL, then: [Insert as first node.] Set LINK[NEW]: START and START: NEW.
Else: [Insert after node with location LOC]
Set LINK(NEW): LINK[LOC) and LINK(LOC): = NEW. [End of If structure.)
5. Exit.

ডিলিশন অ্যালগরিদম (Deletion algorithm) : বিভিন্ন অবস্থায় লিঙ্কড লিস্ট থেকে নোড ডিলিট (Delete) করা হয়ে থাকে। এর মধ্যে উল্লেখযোগ্য দুটি অবস্থা হলো প্রথমত, অ্যালগরিদম নির্দিষ্ট নোডের পূর্ববর্তী নোডকে ডিলিট করে। দ্বিতীয়ত, অ্যালগরিদম প্রদত্ত তথ্য ITEM-সমৃদ্ধ নোডকে ডিলিট করে।

প্রত্যেক অ্যালগরিদমের ক্ষেত্রে আমরা দেখেছি, লিঙ্কড লিস্ট LIST (INFO, LINK, START, AVAIL) আকারে মেমরিতে সজ্জিত আছে। এক্ষেত্রে প্রত্যেক Algorithm-ই AVAIL লিস্টের শুরুতে বিয়োজিত N নোডের স্পেসকে ফেরত পাঠায়। সে
অনুযায়ী উভয় ক্ষেত্রে নিচের অ্যাসাইনমেন্টগুলো সম্পাদনের মাধ্যমে বিয়োজিত N নোডের লোকেশন LOC নির্ধারণ করে।
LINK (LOC): AVAIL এবং AVAIL: LOC,
কোনো অ্যালগরিদম লিস্ট থেকে প্রথম কিংবা শেষ নোডটি ডিলিট করতে চায়। সেক্ষেত্রে প্রথমেই সে চেক করে দেখে লিস্টে কোন নোড আছে কি না। যদি না থাকে অর্থাৎ যদি START = NULL হয়, তাহলে UNDERFLOW প্রিন্ট ম্যাসেজ দেয়। লিস্টের শুরু নোড বিয়োজন । Linked list থেকে কোনো লোড বিয়োজন করার জন্য কিছু নিয়ম রয়েছে।

১। যে নোডকে ডিলিট করব তার আগের নোডটিকে খুঁজে বের করতে হবে।
২। আগের নোড এবং পরবর্তী নোডের মধ্যে Link করতে হবে।
৩। যে নোডকে Delete করতে চাই তাকে Memory থেকে মুছে ফেলতে হবে।

মনে করি, মেমরির List থেকে প্রথম নোডটি Delete কনাতে হবে। যে নোডটি Delete করতে চাই তার পরবর্তী নোডকে List C একে খুঁজে বের করতে হবে। যেমন এখানে ptr-কে Delete করতে চাই। সুতরাং, pu-এর পর নোড হলো B। ptr পূর্ববর্তী Head- এর সাথে Link করা আছে। ptr-এর সাথে Head-এর Link বন্ধ করে ptr-এর পরবর্তী node B-এর সাথে Head-এর Link করাতে এর এবং pir-কে Free করে দিতে হবে।

অ্যালগরিদম

Step-1:IF Head = null Write Underflow Go to step 5 (End of if)
Step-2 :Set ptr = Head
Step-3 : Set Head Head Next
Step 4: Free ptr
Step-5 : Exit

C Editor ব্যবহার করে প্রোগ্রামঃ

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

struct linked list
{
    int number,
        struct linked_list *next.
};
typedef struct linked_list node;
node *head = NULL, *last = NULL;

void create_linked_list();
void print_linked_list();
void insert_at_last(int value);
void insert_at_first(int value);
void insert_after(int key, int value);
void delete_item(int value);
void search_item(int value);

int main()
{
    int key, value;
    // Create a linked list
    printf("Create Linked List\n");
    create_linked_list();
    print_linked_list();
    // Insert value at last position to existing Linked List
    printf("\nInsert new item at lastin");
    scanf("%d", &value);
    insert_at_last(value);
    print_linked_list();
```

```
    // Insert value at first position to existing Linked List
    printf("\nInsert new item at first'n");
    scanf("%d", &value);
    insert_at_first(value);
    print_linked_list();
    // Insert value after a defined value to existing Linked List
    printf("inEnter a KEY (existing item of List), after that you want to insert a valucin");
    scanf("%d", &key);
    printf("\nInsert new item after Ed KEY\n", key);
    scanf("%d", &value);
    insert_after(key, value);
    print_linked_list();
    // Search an item form Linked List
    printf("\nEnter an item to search it from List\n");
    scanf("%d", &value);
    search_item(value);
    // Delete value from List
    printf("\nEnter a value, which you want to delete from list\n");
    scanf("%d", &value);
    delete_item(value);
    print_linked_list();
    return 0;
}
// User Defined Functions
void create_linked_list()
{
    int val;
    while (1)
    {
        scanf("%d", &val);
        printf("Input a number. (Enter-1 to exit)\n");
        if (val == -1)
        {
            // Create a linked list
            printf("Create Linked List\n");
            create_linked_list();
            print_linked_list();
            // Insert value at last position to existing Linked List
            printf("\nInsert new item at lastin");
            scanf("%d", &val);
            insert_at_last(value);
            print_linked_list();
            // Insert value at first position to existing Linked List
            printf("\nInsert new item at first'n");
            scanf("%d", &value);
            insert_at_first(value);
            print_linked_list();
            // Insert value after a defined value to existing Linked List
            printf("inEnter a KEY (existing item of List), after that you want to insert a valucin");
            scanf("%d", &key);
            printf("\nInsert new item after Ed KEY\n", key);
            scanf("%d", &value);
            insert_after(key, value);
            print_linked_list();
            // Search an item form Linked List
            printf("\nEnter an item to search it from List\n");
            scanf("%d", &value);
            search_item(value); // Delete value from List
            printf("\nEnter a value, which you want to delete from list\n");
            scanf("%d", &value);
            delete_item(value);
            print_linked_list();
            return 0;
        }
    }
}
// User Defined Functions
void create_linked_list()
```

```
{
    int val;
    while (1)
    {
        scanf("%d", &val);
        printf("Input a number. (Enter-1 to exit)\n");
        if (val == -1)
        {
            break;
            insert_at_last(val);
        }
    }
}
void insert_at_last(int value)
{
    node *temp_node;
    temp_node = (node *)malloc(sizeof(node));
    temp_node->number = value;
    temp_node->next = NULL;
    // For the 1st element
    if (head == NULL)
    {
        head = temp_node;
        last = temp_node;
    }
    else
    {
        last->next = temp_node;
        last = temp_node;
    }
}
void insert_at_first(int value)
{
    node *temp_node = (node *)malloc(sizeof(node));
    temp_node->number = value;
    temp_node->next = head;
    head = temp_node;
}
void insert_after(int key, int value)
{
    node *myNode = head;
    int flag = 0;
    while (my Node != NULL)
    {
        if (myNode->number == key)
        {
            node *newNode = (node *)malloc(sizeof(node));
            newNode->number = value;
            newNode->next = myNode->next;
            myNode->next = newNode;
            printf("%d is inserted after %d\n", value, key);
            flag = 1;
            break;
            insert_at_last(val);
        }
    }
    myNode = myNode->next;
    if (flag == 0)
        printf("Key not found!\n");
}
void delete_item(int value)
{
    node *myNode = head, *previous = NULL;
    int flag = 0;
    while (myNode != NULL)
    {
        if (myNode->number == value)
        {
```

```
            if (previous == NULL)
                head = myNode->next;
        }
        else
        {
            previous->next = myNode->next;
            printf("%d is deleted from list\n", value);
            flag = 1;
            free(myNode); // need to free up the memory to prevent memory leak
            break;
        }
        previous = myNode;
        myNode + myNode->next;
    }
    if (flag == 0)
        printf("Key not found!\n");
}
void search_item(int value)
{
    node *searchNode = head;
    int flag = 0;
    while (searchNode != NULL)
    {
        if (searchNode->number == value)
        {
            printf("%d is present in this list. Memory address is %d\n", value, searchNode);
            flag = 1;
            break;
        }
        searchNode = searchNode->next;
        if (flag == 0)
        {
            printf("Item not found\n");
        }
    }
}
void print_linked_list()
{
    printf("\n Your full linked list is\n");
    node *myList;
    myList = head;
    while (myList = NULL)

    {
        printf("%d", myList->number);
        myList = myList->next;
    }
    puts("");
}
}
```

Output:

Create Linked List
Input a number. (Enter -1 to exit)
1
Input a number. (Enter -1 to exit)
2
Input a number. (Enter -1 to exit)

Input a number. (Enter -1 to exit)

4
Input, a number. (Enter -1 to exit)
5
Input a number. (Enter -1 to exit)
-1
Your full linked list is
12345

মন্তব্য (Summary) : লিংকড লিস্টে ডাটা সংযোজন এবং বিয়োজন করার জন্য প্রোগ্রাম লেখা ও এক্সিকিউট করার পদ্ধতি এবং প্রোগ্রাম কম্পাইল ও রান করা সম্পন্ন হয়েছে।